PRACTICAL ALGORITHMS FOR CLUSTERING AND MODELING LARGE DATA SETS

ANALYSIS AND IMPROVEMENTS

Dissertation
zur Erlangung des Grades
„Doktor der Naturwissenschaften”
(doctor rerum naturalium)
der Fakultät für Elektrotechnik, Informatik und Mathematik
der Universität Paderborn

vorgelegt von

DIPL.-INFORM.
DANIEL KUNTZE

Paderborn, den 29. August 2013
(überarbeitete Fassung vom 28. Dezember 2013)

angenommen auf Vorschlag von

PROF. DR. JOHANNES BLÖMER
UNIVERSITÄT PADERBORN

PROF. DR. CHRISTIAN SOHLER
TECHNISCHE UNIVERSITÄT DORTMUND

verteidigt am
7. November 2013

DEDICATED TO MY WIFE
EVA REBECCA

ACKNOWLEDGMENTS

This thesis would not have been written without a set of people supporting me. Clustering them leads to four groups of people I am deeply thankful.

First of all, I would like to thank my advisors. Prof. Dr. Johannes Blömer was never too busy to invest time in discussions that inspired new trains of thought and his guidance always advanced my work. Prof. Dr. Christian Sohler not only made me eat sushi but also fed my thoughts with lots of valuable ideas for my research. My colleagues Claudia Jahn, Kathrin Bujna, Dr. Stefanie Mense, Dr. Marcel Ackermann, Jonas Gefele, Peter Günther, Dr. Volker Krummel, Gennadij Liske, David Teusner, Paul Wolf, and of course the IT crowd (Ulrich Ahlers, Tobias Schultz-Friese, Thomas Thissen, and Michael Utermöhle) helped me out when I was stuck and provided welcome excuses from work. I learned a lot during my time with you.

The research presented in this thesis was supported by the Priority Programme 'Algorithm Engineering' of the German Research Foundation (DFG), grants BL 314/6-1 to BL 314/6-3, for which I am also really thankful.

I also want to thank my dear colleague Kathrin Bujna, who tamed R for me and made it draw beautiful pictures.

Last but not least, I want to express my appreciation for the family and friends cluster. Without Emily and Ronja, I might have finished more quickly but with less fun. You were the cutest to come into my office for chocolate! Furthermore, I am very grateful for my parents and parents-in-law. Their support, love, and prayers were a great encouragement. My friend Peter Richert never stopped to cheer me on — thanks for believing in me, mate.

Most of all, I want to thank my wife for accompanying me on my way, for sharing all the ups and downs, and for giving me unfailing support. Without you, I would never have succeeded.

ABSTRACT

In times of big data, large-scale database applications are of increasing importance. A central task in this field is to reduce the amount of data that has to be processed. One way to approach this task is to capture the structure of the given data. This thesis deals with two particular data structuring techniques.

The first part of this thesis is about cluster analysis. More precisely, we consider the diameter k-clustering problem. The subject of this problem is to partition a finite subset of $\mathbb{R}^d$ into k subsets called clusters such that the largest diameter of the clusters is minimized. One early clustering algorithm that computes a hierarchy of approximate solutions to this problem (for all values of k) is the agglomerative clustering algorithm with the complete linkage strategy. For decades, this algorithm has been widely used by practitioners. However, it is not well studied theoretically. We provide the first analysis of the agglomerative complete linkage clustering algorithm. Assuming that the dimension d is a constant, we show that for any k the solution computed by this algorithm is an $O(\log k)$-approximation to the diameter k-clustering problem. Our analysis does not only hold for the Euclidean distance but for any metric that is based on a norm. Furthermore, we analyze the closely related k-center and discrete k-center problem. For the corresponding agglomerative algorithms, we deduce an approximation factor of $O(\log k)$ as well.

The second part of this thesis deals with data modeling by training statistical models. More precisely, we focus on the parameter estimation problem for general and Gaussian mixture distributions. We analyze a probabilistic variant of the Expectation-Maximization (EM) algorithm, known as the Stochastic EM or SEM algorithm. Unlike the original work, we focus on the analysis of a single run of the algorithm. First, we discuss the SEM algorithm for general mixture distributions. Second, we consider Gaussian mixture models and show that with high probability the updates of the EM algorithm and its probabilistic variant are almost the same, given that the data set is sufficiently large. A series of experiments confirms that this still holds for a large number of successive update steps. Furthermore, we explain why the SEM algorithm is considerably faster than the classical EM algorithm. In particular, we show that the probabilistic variant establishes a constant factor speedup for Gaussian mixture models.

ZUSAMMENFASSUNG

In Zeiten von Big Data sind große Datenbankanwendungen von wachsender Bedeutung. Eine zentrale Aufgabe aus diesem Bereich ist die Reduzierung der zu verarbeitenden Datenmenge. Diese Aufgabe lässt sich beispielsweise lösen, indem die Struktur der gegebenen Daten erfasst wird. In dieser Dissertation beschäftigen wir uns mit zwei Techniken zur Strukturierung von großen Datenmengen.

Im ersten Teil der Arbeit geht es um Clusteranalyse, speziell um das Durchmesser-k-Clustering-Problem. Dabei wird eine endliche Teilmenge des $\mathbb{R}^d$ in k Cluster partitioniert, so dass der größte Clusterdurchmesser minimiert wird. Ein früher Clusteringalgorithmus, der eine Hierarchie von Näherungslösungen (für alle Werte von k) zu diesem Problem berechnet, ist der agglomerative Clusteringalgorithmus mit der Complete-Linkage-Strategie. Dieser Algorithmus erfährt in der Praxis seit Jahrzehnten eine breite Anwendung, ist aber theoretisch nicht gut untersucht. Wir liefern die erste Analyse des agglomerativen Complete-Linkage Clusteringalgorithmus. Wir zeigen, dass die vom Algorithmus berechnete Lösung für konstante Dimension d für jedes k eine $O(\log k)$-Approximation des Durchmesser-k-Clustering-Problems ist. Unsere Analyse gilt dabei nicht nur für den Euklidischen Abstand, sondern für jede Metrik, die auf einer Norm basiert. Außerdem analysieren wir das k-Center-Problem und das diskrete k-Center-Problem, welche eng mit dem Durchmesser-k-Clustering-Problem verwandt sind. Für die dazugehörigen agglomerativen Algorithmen leiten wir ebenfalls einen Approximationsfaktor von $O(\log k)$ her.

Der zweite Teil der Arbeit beschäftigt sich mit der Modellierung von Daten durch das Trainieren von statistischen Modellen. Hier betrachten wir die Parameterschätzung von allgemeinen und Gaußschen Mixturverteilungen. Wir analysieren eine probabilistische Variante des Expectation-Maximization (EM) Algorithmus, die als Stochastic EM oder SEM Algorithmus bekannt ist. Im Gegensatz zu den ursprünglichen Arbeiten zum SEM Algorithmus betrachten wir in unserer Analyse einen einzelnen Durchlauf des Algorithmus. Dabei beschreiben wir den Algorithmus zunächst für allgemeine Mixturverteilungen. Danach betrachten wir Gaußsche Mixturmodelle und zeigen für diese, dass die Updates des EM Algorithmus und seiner probabilistischen Variante mit hoher Wahrscheinlichkeit fast identisch sind, wenn die Datenmenge groß genug ist. Eine Reihe von Experimenten bestätigt, dass dies auch für eine große Anzahl aufeinanderfolgender Updateschritte noch gilt. Darüber hinaus erörtern wir, warum der SEM Algorithmus schneller arbeitet als der klassische EM Algorithmus. Insbesondere zeigen wir für Gaußsche Mixturmodelle, dass die probabilistische Variante eine Beschleunigung um einen konstanten Faktor erreicht.

CONTENTS

1 INTRODUCTION 1
- 1.1 Cluster analysis 2
- 1.2 Parameter estimation 5
- 1.3 Outline of the thesis 9

I ON CLUSTER ANALYSIS 11

2 CLUSTERING BASICS 13
- 2.1 The notion of similarity 14
- 2.2 Clustering quality 14
- 2.3 Problem Definitions 16

3 ANALYSIS OF AGGLOMERATIVE CLUSTERING 17
- 3.1 Preliminaries 18
- 3.2 Discrete k-center clustering 20
- 3.3 k-center clustering 24
- 3.4 Diameter k-clustering 34
- 3.5 The one-dimensional case 41
- 3.6 Lower bounds 45

II ON PARAMETER ESTIMATION 55

4 STATISTICAL MODELS 57
- 4.1 The parameter estimation problem 58
- 4.2 General mixture models 60
- 4.3 Gaussian mixture models 61
- 4.4 Handy notions from probability theory 67

5 THE CLASSICAL EM ALGORITHM 71
- 5.1 Convergence of the EM algorithm 73
- 5.2 The EM algorithm for mixture distributions 77
- 5.3 The EM algorithm for Gaussian mixtures 78

6 THE STOCHASTIC EM ALGORITHM 81
- 6.1 The generic SEM algorithm 82
- 6.2 The SEM algorithm for mixture distributions 83
- 6.3 The SEM algorithm for Gaussian mixtures 86

7 EXPERIMENTAL ANALYSIS 99
- 7.1 Implementation 99
- 7.2 Data sets 100
- 7.3 Experiments 102
- 7.4 Results 105

1 INTRODUCTION

Recently, as part of the *Human Brain Project* [Walker, 2012], researchers from Canada and Germany released *BigBrain*, a high-resolution 3D model of the human brain [Amunts et al., 2013]. The aim of this model is to provide "considerable neuroanatomical insight into the human brain, thereby allowing the extraction of microscopic data for modeling and simulation". The long-term objective of the Human Brain Project is the simulation of the human brain on supercomputers, thus realizing a recent dream of mankind. The chances of success for a venture like this were already discussed in the 1950s by great minds as Alan Turing[1] and John von Neumann[2].

Although science is not able to simulate the human brain yet, a lot of effort is put into the imitation of its capabilities. One of these capabilities is to handle large amounts of data. The human retina alone transmits approximately one megabyte of data to the brain every second [Koch et al., 2006]. Fortunately, we do not have to consciously process each bit of information that we see. Instead, our brain's abstraction capacity reduces the amount of information. This enables us to recognize a white area covered with black speckles as a page of printed text. Moreover, we are able to classify the speckles as Roman letters and recognize groups of letters as English words. Then, by processing one word after another while ignoring the remaining words on the page, we are able to read whole sentences.

For the sake of simplicity let's wrongly assume that the information of a sentence is simply the sum of information in the single words. Furthermore, we assume that the amount of information in a single word is simply the number of bits needed to identify one word of the English language. If we estimate the size of the English vocabulary by 500 000 words, then we need 19 bits per word. An average reader is able to read about 4 words per second which corresponds to 76 bits per second. This rough approximation shows that the brain reduces the information from the perception in the eye to the recognition of the words by several orders of magnitude.

1 In 1950, Turing wrote his well-known article "Computing Machinery and Intelligence" [Turing, 1950].

2 Shortly before his death in 1957, von Neumann began to write the unfinished and posthumously published book "The Computer and the Brain" [von Neumann, 1958].

Furthermore, the brain is also able to store large amounts of data. The number of neurons in the human brain is estimated to about 86 billion cells, 16 billion of which are located in the cerebral cortex [Azevedo et al., 2009]. We may assume that each neuron in the cerebral cortex is connected to other neurons by 50 000 synapses on average [Brewer et al., 2009]. This yields a total number of approximately 800 trillion synapses. It is not known how the brain stores information and which role the interaction between neurons and synapses plays. However, recent work suggests that a synapse switches between discrete states [Montgomery and Madison, 2004]. If we assume that each synapse can hold one bit of information, we get a total storage capacity of approximately 100 terabytes in the cerebral cortex.

The combination of the capability to store large amounts of data and the ability to process it resembles the requirements of large-scale database applications. With continuously growing databases, it becomes increasingly important to reduce the amount of data that has to be processed. That is, in order to work with very large data sets, we need to be able to extract the essential information. This is quite similar to the brain's ability to extract text from the electrochemical impulses received from the optic nerve. One step into this direction is to try and capture the structure of a given data set. The approaches to this task are as diverse as the underlying applications. This thesis deals with two particular data structuring techniques introduced in the following two sections.

1.1 CLUSTER ANALYSIS

One way of investigating the structure of a given data set is to divide it into subsets of similar data elements. This approach is called cluster analysis or simply clustering. There are many applications for clustering, including data compression [Pereira et al., 1993], structuring results of search engines [Broder et al., 1997], analysis of gene expression data [Eisen et al., 1998], anomaly detection [Lee et al., 2008], and community detection [Meyerhenke and Staudt, 2013]. For every application, a proper objective function is used to measure the quality of a clustering. One particular objective function that is based on a distance measure between data elements is the largest diameter of the clusters. If the desired number of clusters k is given, we call the problem of minimizing this objective function the *diameter k-clustering problem*. In this thesis, we focus on the diameter k-clustering problem and two closely related clustering problems.

One of the earliest and most widely used clustering algorithms is agglomerative clustering. The history of agglomerative clustering goes back to the 1950s at least [Florek et al., 1951, McQuitty, 1957]. Later, biological taxonomy became one of the driving forces of cluster analysis. In [Sneath and Sokal, 1973] the authors, who where the first biologists using computers to classify organisms, discuss several agglomerative clustering methods. Agglomerative clustering is a bottom-up clustering process. At the beginning, every data element forms its own cluster. In each subsequent step, the two 'closest' clusters will be merged until only one cluster remains.

In order to define the agglomerative algorithm properly, we have to specify what it means for two clusters to be 'close'. In case of the diameter clustering problem mentioned above, two clusters are defined to be close when the distance between their farthest pair of data elements is small. Then, the two 'closest' clusters are the two clusters that minimize the diameter of their union. Using this definition of closeness is called the *complete linkage strategy*. Alternate strategies are to minimize the distance between the closest pair of data elements or to minimize the average distance between data elements from the two clusters. These are called *single linkage strategy*[3] and *average linkage strategy*, respectively.

Agglomerative clustering algorithms create a hierarchy of clusters, such that for any two different clusters A and B from different levels of the hierarchy we either have $A \cap B = \emptyset$, $A \subset B$, or $B \subset A$. Such a hierarchy is useful in many applications, for example, when one is interested in hereditary properties of the clusters (as in some bioinformatics applications) or if the exact number of clusters is a priori unknown.

1.1.1 RELATED WORK

In this thesis, we study the agglomerative clustering algorithm using the complete linkage strategy. Since the complete linkage strategy minimizes the diameter in each step, this algorithm is a natural choice to find hierarchical diameter k-clusterings. The running time is obviously polynomial in the description length of the input data. Therefore, our only goal in this thesis is to give an approximation guarantee for the diameter k-clustering problem. The approximation guarantee is given by a factor α such that the largest diameter of the k-clustering computed by the algorithm is at most α times the largest diameter of an optimal k-clustering. Although the agglomerative complete linkage clustering algorithm is widely used, there are only few theoretical results considering the quality of the clustering computed by this algorithm. In [Dasgupta and Long, 2005] the authors provide a certain metric distance function such that this algorithm computes a k-clustering with an approximation factor of $\Omega(\log k)$.

The diameter k-clustering problem is closely related to the k-*center problem*. In this problem, we search for k centers and the objective is

3 Using the single linkage strategy is equivalent to computing a minimum spanning tree of the graph induced by the distance function using Kruskal's algorithm.

to minimize the maximum distance of any data element to the nearest center. When the centers are restricted to come from the input data set, the problem is called the *discrete k-center problem*. It can easily be verified that for metric distance functions the costs of optimal solutions to all three problems are within a factor of 2 from each other.

For the Euclidean case, we know that for fixed k, i. e., when we are not interested in a hierarchy of clusterings, the diameter k-clustering problem and the k-center problem are $\mathcal{NP}$-hard. In fact, it is already $\mathcal{NP}$-hard to approximate both problems with an approximation factor below 1.96 and 1.82, respectively [Feder and Greene, 1988]. These non-approximability results are transferable to the discrete k-center problem for approximation factors below 1.73 [Leder, 2013].

Furthermore, there exist provably good approximation algorithms in this case. For the k-center problem, a simple 2-approximation algorithm is known for metric spaces [Gonzalez, 1985]. Since this algorithm chooses the cluster centers from the input data set, it is also a 2-approximation algorithm for the discrete k-center problem. The analysis of this algorithm is immediately transferrable to the diameter k-clustering problem and yields a 2-approximation algorithm for this problem, too. For the k-center problem, a variety of results is known. For example, for the Euclidean metric in [Bādoiu et al., 2002] a $(1 + \epsilon)$-approximation algorithm with running time $2^{O(k \log k/\epsilon^2)} dn$ is shown. This implies a $(2 + \epsilon)$-approximation algorithm with the same running time for the diameter k-clustering problem. Moreover, for metric spaces a hierarchical clustering strategy is known with an approximation guarantee of 8 for the discrete k-center problem [Dasgupta and Long, 2005]. This implies an algorithm with an approximation guarantee of 16 for the diameter k-clustering problem.

Part I of this thesis as well as all of the above mentioned work is about static clustering, i. e., in the problem definition we are given the whole input data set at once. An alternative model of the input data is to consider sequences of data elements that are given one after another. In [Charikar et al., 1997], the authors discuss clustering in a so-called *incremental clustering* model. They give an algorithm with constant approximation factor that maintains a hierarchical clustering while new elements are added to the input data set. Furthermore, they show a lower bound of $\Omega(\log k)$ for the agglomerative complete linkage algorithm and the diameter k-clustering problem. However, since their model differs from ours, their results have no bearing on our results.

1.1.2 OUR CONTRIBUTION

One of the open problems discussed in [Dasgupta and Long, 2005] is to derive a non-trivial upper bound for the approximation guarantee of the classical agglomerative complete linkage clustering algorithm. In this thesis, we present such an analysis for the agglomerative complete linkage and related clustering algorithms. Our results hold for metrics on $\mathbb{R}^d$ that are based on a norm as for example the Euclidean metric. We prove that the agglomerative solution to the diameter k-clustering problem is an $O(\log k)$-approximation. Here, the O-notation hides a constant that is doubly exponential in d. This approximation guarantee holds for every level of the hierarchy computed by the algorithm. That is, we compare each computed k-clustering with an optimal solution for that particular value of k. These optimal k-clusterings do not necessarily form a hierarchy. In fact, there are simple examples where optimal solutions have no hierarchical structure.

Our analysis also yields that if we allow $2k$ instead of k clusters and compare the cost of the computed $2k$-clustering to an optimal solution with k clusters, the approximation factor is independent of k and depends only on d. Moreover, the techniques of our analysis can be applied to prove stronger results for the k-center problem and the discrete k-center problem. For the k-center problem, we derive an approximation guarantee that is logarithmic in k and only singly exponential in d. For the discrete k-center problem, we derive an approximation guarantee that is logarithmic in k and the dependence on d is only linear and additive.

Furthermore, we give almost matching upper and lower bounds for the one-dimensional case. These bounds are independent of k. For $d \geqslant 2$ and the metric based on the L_∞-norm, we provide a lower bound that exceeds the upper bound for $d = 1$. For $d \geqslant 3$, we give a lower bound for the Euclidean case which is larger than the lower bound for $d = 1$. Finally, we construct data sets which provide lower bounds for any metric based on an L_p-norm with $1 \leqslant p \leqslant \infty$. However, the construction of these data sets needs the dimension to depend on k.

Our analysis, presented in Chapter 3, was previously published in [Ackermann et al., 2011, Ackermann et al., 2012].

1.2 PARAMETER ESTIMATION

Apart from the clustering approach outlined in Section 1.1, another way of revealing the structure of large data sets is to train the parameters of statistical models. This approach is discussed in Part II of this thesis. To describe a given data set by the parameters of a statistical model is a central task in the field of data mining and machine learning. Many applications in this field deal with real-world data that is suitable to be modeled by stochastic processes. That is, the data is assumed to be generated by a stochastic process which is governed by some model parameters. Then, the goal is to optimize the parameters such that the stochastic process is a likely source of the data. Presumably, one of the first studies of this problem was done by Karl Pearson for a data set consisting of measurements of the frontal breadth of the carapace of 1 000 female shore crabs [Pearson, 1894]. Pearson showed that the data was most likely a mixture of two Gaussian distributions. This suggests that the crabs belonged to two different species. To compute the parameters of the mixture, Pearson used the method of moments to derive a ninth-degree polynomial by hand which gave rise to a set of candidate parameters.

Today, we are able to work with much larger data sets and we do not have to do any calculations by hand. Instead, we are equipped with powerful algorithms for the parameter optimization. One way to optimize the parameters is to maximize the probability of the given data set over the choice of the parameters. As a function of the model parameters, this probability is called the likelihood function. The problem of maximizing the likelihood is called the parameter estimation problem. The Expectation-Maximization (EM) algorithm introduced in [Dempster et al., 1977] is a general scheme for finding maximum likelihood solutions for this parameter estimation problem. It is used when the observed data X can be seen as incomplete and when the parameter estimation problem given the corresponding complete data (X, Z) is easy to solve. Starting with an initial set of model parameters, the algorithm repeatedly performs two steps to compute a new set of parameters. The first step derives an optimal distribution for the hidden values. The second step computes the new set of parameters by maximizing the expectation (over the hidden values) of the complete-data likelihood.

Applications of the EM algorithm to special cases were already known before the method was formulated in its general form in [Dempster et al., 1977]. For example, the well known Baum-Welch algorithm (see [Baum et al., 1970]) for the computation of parameters for hidden Markov models is an instantiation of a generalized variant of the EM algorithm. For exponential families, the method was studied in [Sundberg, 1974]. More examples can be found in [Dempster et al., 1977].

1.2.1 RELATED WORK

During the decades since its first presentation, a lot of work has been done to improve the EM algorithm. Most of these improvements deal with one or more of three major drawbacks of the algorithm. First, the convergence of the EM algorithm can be very slow. For a discussion of several speed-up techniques see Chapter 4 in [McLachlan and Krishnan, 2008]. Second, the EM algorithm is prone to converge only to a local maximum, or even worse, to a saddle point of the likelihood function [Wu, 1983, Dias and Wedel, 2004]. One way to deal with this problem is to compute 'good' initial estimates [Biernacki et al., 2003] or to restart the algorithm with several different initializations in the hope that one iteration leads to a satisfying solution. Finally, in some cases the EM algorithm is not applicable at all since the maximization step is computationally intractable. To overcome this problem, the maximization step has to be substantially simplified.

There exist a number of stochastic variants of the EM algorithm that try to deal with these drawbacks, including the Stochastic EM or SEM algorithm [Celeux and Diebolt, 1985], the Stochastic Approximation EM or SAEM algorithm [Celeux and Diebolt, 1992], and the Monte Carlo EM or MCEM algorithm [Wei and Tanner, 1990]. The MCEM algorithm replaces the distribution over the hidden values from the first step of the EM algorithm by an empirical Monte Carlo approximation. Thus, for continuous random variables Z, it replaces analytic by numerical computation. Furthermore, due to the inherent randomness, the algorithm is capable of escaping from a saddle point or from a local maximum of the likelihood function.

The SEM algorithm is even simpler than the MCEM algorithm. Instead of maximizing the expectation as done in the second step of the EM algorithm, it uses the distribution from the first step to simply guess the missing values. That is, the algorithm samples an assignment for the hidden values Z. Afterwards, it maximizes the complete-data likelihood only for that fixed assignment. This considerably reduces the computational complexity and may lead to the applicability of the EM scheme in the first place. The SEM algorithm can be seen as a special case of the MCEM algorithm where only one sample is drawn to approximate the distribution over the hidden values. It is known that for particular classes of statistical models, the sequence of models generated by the SEM iterations is an ergodic Markov chain converging weakly to a stationary distribution over models. Furthermore, it is known that under appropriate assumptions and for the number of input points going to infinity, the mean of this stationary distribution converges to the maximum likelihood estimate [Ip, 1994].

A suitable application of the SEM algorithm is the parameter estimation problem for mixture distributions. In this model, the data set consists of independent draws from a mixture distribution and the hidden values are usually assumed to be the assignments of each data element to the component distribution it was generated by. That is, for each data element, the SEM algorithm guesses which component distribution it was generated by. Obviously, if the number of data elements is small, then each wrong assignment has a large impact. And indeed, a limitation of the SEM algorithm is that it does not work for very small data sets. Therefore, the authors of the original SEM work introduced the SAEM algorithm later. The SAEM algorithm is a hybrid algorithm that mixes the computations of the SEM algorithm with the computations of the classical EM algorithm. At the beginning it works like a pure SEM algorithm and then it gradually transforms until it finally does pure EM computations. For a comparison of all three mentioned stochastic EM variants see [Celeux et al., 1995].

1.2.2 OUR CONTRIBUTION

As already mentioned in the previous section, the original line of work on the SEM algorithm focused on stochastic properties of the sequence of models generated by the SEM algorithm. However, to approximate the above mentioned mean that converges to the maximum likelihood estimate, we would need to average over a large number of runs of the SEM algorithm. In this thesis, we present a new analysis of the SEM algorithm for Gaussian mixture models. Unlike the previous work, our analysis studies only a single run of the algorithm. Furthermore, in addition to our theoretical results, we also provide experimental results for various artificial and real world data sets.

Since practitioners are often satisfied with the solutions computed by the classical EM algorithm, we analyze the SEM algorithm in relation to the EM algorithm. More precisely, we examine the parameter updates computed in an arbitrary iteration of the SEM algorithm and compare them to the updates computed by the EM algorithm for the same previous parameter estimate. We show that with high probability the updates of the EM and SEM algorithm are almost the same, given that the input data set is sufficiently large. Our results are stated as three separate theorems. The first theorem gives a high probability bound on the difference of the weight updates. This result does not only hold for Gaussian mixture models, but also for arbitrary mixture models. The second theorem gives a high probability bound on the distance between the means computed by the EM and SEM iteration. Finally, the third theorem does the same for the covariance updates.

Since the computed parameters generally differ from each other after a single round of the two algorithms, our analysis does not work for a sequence of several EM and SEM iterations. However, our experiments confirm that the similarity of the parameters mostly persists even after a large number of successive update steps. Moreover, we explain why the SEM algorithm is considerably faster than the classical EM algorithm. In particular, we show that the probabilistic variant establishes a constant factor speedup for Gaussian mixture models.

Our theoretical and experimental analysis, presented in Chapter 6 and Chapter 7, was also published in [Blömer et al., 2013].

1.3 OUTLINE OF THE THESIS

This thesis is organized into two parts. In Part I we turn towards cluster analysis. After giving a short introduction to clustering in Chapter 2, we present the first analysis of a classical clustering algorithm in Chapter 3. Part II is about statistical models and the parameter estimation problem. The basic concepts needed later are introduced in Chapter 4. In Chapter 5, we discuss the classical Expectation-Maximization algorithm. In Chapter 6, we analyze a probabilistic variant of the EM algorithm. Finally, in Chapter 7, we present an experimental comparison of both algorithms.

On some pages you will find cross-references and additional remarks written in the margins that may be useful to understand the main text. Furthermore, some bibliographic references are only mentioned in these marginal notes.

Part I

ON CLUSTER ANALYSIS

The first part of this thesis is about cluster analysis. We focus on so-called distance-based hard clustering. Using this approach, the distances between the elements of a given data set are used to compute a partition of the data into subsets of elements that are close to each other.

The main result of Part I is the first analysis of the quality of agglomerative complete linkage clustering for the diameter k-clustering problem.

2

CLUSTERING BASICS

Simply put, *clustering* or *cluster analysis* is the process of organizing a given set of items into groups (called clusters) of similar items. The term is used for the entire number of techniques to solve this task as well as for the task itself. Furthermore, the term *clustering* is also used for a resulting set of clusters. If a clustering consists of k clusters, it is called a k-*clustering*.

Ideally, all elements of a particular cluster are similar while elements from different clusters are dissimilar. It is evident that the notion of similarity or dissimilarity is a crucial ingredient for the formulation of clustering problems. As a consequence, the different ways to model a cluster are as numerous as the number of clustering applications. There is a broad range of (possibly overlapping) approaches, including graph-based clustering [Schaeffer, 2007, Fortunato, 2010], density-based clustering [Kriegel et al., 2011a], distribution-based clustering (cf. Chapter 4), and distance-based clustering (see Section 2.1).

Another key differentiator between common clustering applications is the certainty of membership in a cluster. The two main categories are *hard* and *soft clustering*. In case of soft clustering, an element of the input data set may be assigned to multiple clusters at the same time and in addition this assignment can be weighted. For example, using a distribution-based clustering approach, we may want to model the input data by a mixture distribution (cf. Chapter 4.2). Then, we can identify the mixture components with the clusters and assign the elements of the input data set to each cluster with a weight proportional to the probability of the element in the corresponding mixture component. In case of hard clustering however, each element is assigned to exactly one cluster. That is, a hard clustering is actually a partition of the given data set.

An important special case of hard clustering is the so-called *hierarchical clustering*. We call a collection of k-clusterings of the same finite data set X, but for different values of k, hierarchical, if it fulfills the following two properties. First, for any k, the collection contains at most one k-clustering. Second, for any two of its clusterings $\mathcal{C}_i, \mathcal{C}_j$ with $|\mathcal{C}_i| = i < j = |\mathcal{C}_j|$, every cluster in $\mathcal{C}_i$ is the union of one or more clusters from $\mathcal{C}_j$. A hierarchical collection of clusterings is called a hierarchical clustering.

In the following, we introduce the concepts of clustering that we need to apply our analysis of agglomerative clustering in Chapter 3. In particular, we consider distance-based hard clustering only. For a thorough introduction to clustering see for example [Tan et al., 2005].

2.1 THE NOTION OF SIMILARITY

The distance-based clustering approach establishes a notion of dissimilarity based on the distances between the elements of the input data. Therefore, we start our formal introduction to cluster analysis with the definition of *distance measures*. We use the term distance measure to differentiate from the distance function of a metric since we neither require symmetry nor the triangle inequality.

Definition 2.1. *Let* $\mathbb{X}$ *be an arbitrary domain. A function* $D : \mathbb{X} \times \mathbb{X} \to \mathbb{R}$ *is called a* distance measure, *if the following conditions are satisfied for all* $x, y \in \mathbb{X}$*:*

1. $D(x, y) \geqslant 0$ *(non-negativity)*
2. $D(x, y) = 0 \Leftrightarrow x = y$ *(identity of indiscernibles)*

Cluster analysis is established for a wide variety of distance measures. Although it is very common to embed the input data into a metric space like the d-dimensional Euclidean vector space $\mathbb{R}^d$, some widely used distance measures do not form a metric. For example, the *squared Euclidean distance* and the *Mahalanobis distance* both are no metrics since they do not fulfill the triangle inequality. Other distance measures as for example the *Kullback-Leibler divergence* or the *Itakura-Saito divergence* (both defined for $\mathbb{X} = \mathbb{R}^d_{\geqslant 0}$) are not even symmetric. For a discussion of clustering algorithms for non-metric distance measures see [Ackermann, 2009]. For the remainder of this chapter, we consider metric distance measures only.

Definition 2.2. *Let* $\mathbb{X}$ *be an arbitrary domain. A function* $D : \mathbb{X} \times \mathbb{X} \to \mathbb{R}$ *is called a* metric, *if* D *is a distance measure and in addition, the following conditions are satisfied for all* $x, y, z \in \mathbb{X}$*:*

3. $D(x, y) = D(y, x)$ *(symmetry)*
4. $D(x, y) \leqslant D(x, z) + D(z, y)$ *(triangle inequality)*

2.2 CLUSTERING QUALITY

To distinguish good clusterings from bad clusterings we need to introduce a measure of quality. It is obvious that the choice of the right measure highly depends on the particular clustering application. Generally, the desired properties of the clustering are transformed to a cost function that maps clusterings to real numbers. Then, the goal

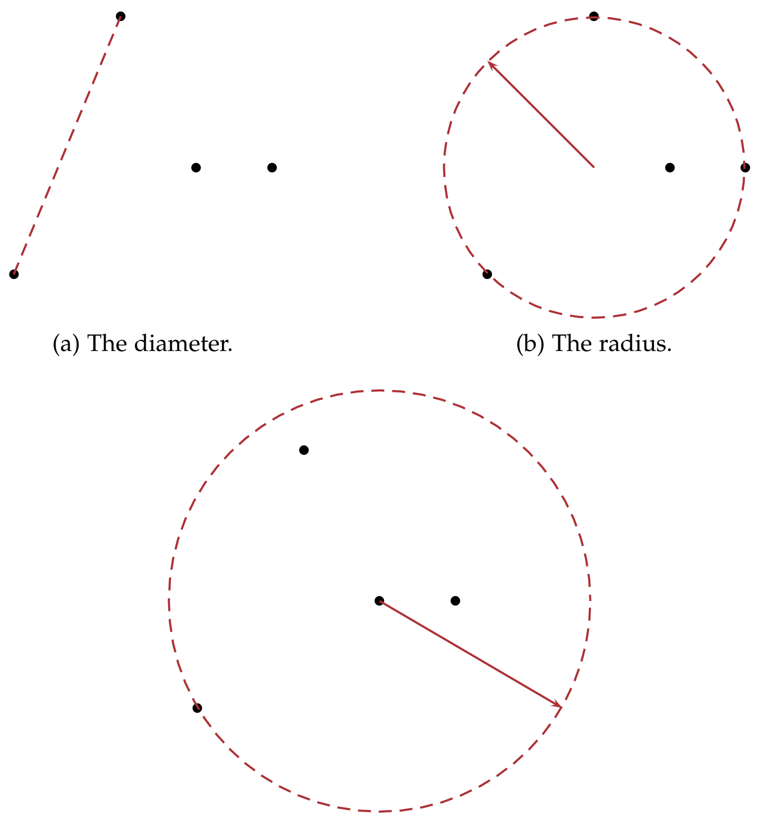

(a) The diameter. (b) The radius. (c) The discrete radius.

Figure 1: Different measurements for the same cluster.

of the clustering process is to minimize the cost of the resulting clustering.

Before introducing three popular hard clustering cost functions, we give a formal definition of a k-clustering.

Definition 2.3. *Let $\mathbb{X}$ be an arbitrary domain, $k \in \mathbb{N}$ and $X \subset \mathbb{X}$ a finite set with $|X| \geqslant k$. Then, a k-clustering of X is a partition $\mathcal{C}_k = \{C_1, \ldots, C_k\}$ of X into k non-empty subsets. The subsets $C_1, \ldots, C_k$ are called clusters.*

The first cost function we consider is the diameter cost of a clustering which is simply the largest diameter among its clusters.

Definition 2.4. *Let $\mathbb{X}$ be an arbitrary domain with a distance measure D. Furthermore, let $X \subseteq \mathbb{X}$ be a finite input data set and $\mathcal{C} = \{C_1, \ldots, C_k\}$ a k-clustering of X.*

Then, the diameter cost *of $\mathcal{C}$ is defined as*

$$\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}) := \max_{C \in \mathcal{C}} \mathrm{diam}(C),$$

where $\mathrm{diam}(C) := \max_{x,y \in C} D(x, y)$ denotes the diameter of C. cf. Figure 1a

The diameter cost only depends on the distances between elements of the input data set. For the definition of the radius cost, we additionally introduce so-called cluster centers. These centers come from the same domain that the input data set is embedded into.

Definition 2.5. *Let $\mathbb{X}$ be an arbitrary domain with a distance measure D. Furthermore, let $X \subseteq \mathbb{X}$ be a finite input data set and $\mathcal{C} = \{C_1, \ldots, C_k\}$ a k-clustering of X.*

Then, the radius cost *of $\mathcal{C}$ is defined as*

$$\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}) := \max_{C \in \mathcal{C}} \mathrm{rad}(C),$$

cf. Figure 1b

where $\mathrm{rad}(C) := \min_{y \in \mathbb{X}} \max_{x \in C} D(x, y)$ denotes the radius of C.

The discrete radius cost is a special case of the radius cost where the centers have to be elements of the input data set.

Definition 2.6. *Let $\mathbb{X}$ be an arbitrary domain with a distance measure D. Furthermore, let $X \subseteq \mathbb{X}$ be a finite input data set and $\mathcal{C} = \{C_1, \ldots, C_k\}$ a k-clustering of X.*

Then, the discrete radius cost *of $\mathcal{C}$ is defined as*

$$\mathrm{cost}_{\mathrm{drad}}(\mathcal{C}) := \max_{C \in \mathcal{C}} \mathrm{drad}(C),$$

cf. Figure 1c

where $\mathrm{drad}(C) := \min_{y \in C} \max_{x \in C} D(x, y)$ denotes the discrete radius of C.

That is, if we choose $\mathbb{X} = X$ the radius cost coincides with the discrete radius cost.

2.3 PROBLEM DEFINITIONS

Based on the three cost functions introduced in Section 2.2 we define the corresponding clustering problems. All three problems are discussed in Chapter 3.

Problem 2.7 (diameter k-clustering)**.** *Given $k \in \mathbb{N}$ and a finite set $X \subset \mathbb{R}^d$ with $|X| \geqslant k$, find a k-clustering $\mathcal{C}_k$ of X with minimal diameter cost.*

Problem 2.8 (k-center)**.** *Given $k \in \mathbb{N}$ and a finite set $X \subset \mathbb{R}^d$ with $|X| \geqslant k$, find a k-clustering $\mathcal{C}_k$ of X with minimal radius cost.*

Problem 2.9 (discrete k-center)**.** *Given $k \in \mathbb{N}$ and a finite set $X \subset \mathbb{R}^d$ with $|X| \geqslant k$, find a k-clustering $\mathcal{C}_k$ of X with minimal discrete radius cost.*

3

ANALYSIS OF AGGLOMERATIVE CLUSTERING

In this chapter we analyze the agglomerative clustering algorithms for the (discrete) k-center problem and the diameter k-clustering problem. As mentioned in the introduction, an agglomerative algorithm takes a bottom-up approach. It starts with the $|X|$-clustering that contains one cluster for each element of a given data set and then successively merges two of the remaining clusters such that the cost of the resulting clustering is minimized. That is, in each merge step the agglomerative algorithms for Problem 2.7, Problem 2.8 and Problem 2.9 minimize the diameter, the radius and the discrete radius of the resulting cluster, respectively.

Our main objective is the agglomerative complete linkage clustering algorithm, which minimizes the diameter in every step. Nevertheless, we start with the analysis of the agglomerative algorithm for the discrete k-center problem since it is the simplest one of the three. Then we adapt our analysis to the k-center problem and finally to the diameter k-clustering problem. In each case we need to introduce further techniques to deal with the increased complexity of the given problem.

We show that all three algorithms compute an $O(\log k)$-approximation for the particular corresponding clustering problem. However, the dependency on the dimension which is hidden in the O-notation ranges from only linear and additive in case of the discrete k-center problem to a factor that is doubly exponential in case of the diameter k-clustering problem. As mentioned in the introduction, the cost of optimal solutions to the three problems are within a factor of 2 from each other. That is, each algorithm computes an $O(\log k)$-approximation for all three problems. However, we will analyze the proper agglomerative algorithm for each problem.

Strictly speaking, the clusterings computed by the agglomerative algorithms stated as Algorithm 3.1, 3.2 and 3.3 are not uniquely determined, since the minimization step may be ambiguous. However, all our results hold for any particular tie-breaking strategy (even nondeterministic ones). To keep the discussion simple, we ignore such subtleties unless they are crucial.

3.1 PRELIMINARIES

Throughout this chapter, we confine ourselves to metric distance measures that are defined on $\mathbb{R}^d$ for some $d \in \mathbb{N}$. That is, we consider input data sets that are finite subsets of $\mathbb{R}^d$. Our results hold for arbitrary metrics that are based on a norm, i. e., we consider only distance measures of the form

$$D(x, y) = \|x - y\|$$

The figures in this chapter assume the Euclidean metric.

for an arbitrary norm $\|\cdot\|$. Readers who are not familiar with arbitrary metrics or are only interested in the Euclidean case, may assume that the Euclidean norm $\|\cdot\|_2$ is used, i. e.,

$$\|x - y\| = \sqrt{\sum_{i=1}^{d} (x_i - y_i)^2}.$$

Furthermore, for $r \in \mathbb{R}$ and $y \in \mathbb{R}^d$, we denote the closed d-dimensional ball of radius r centered at y by

$$B_r^d(y) := \{x \mid \|x - y\| \leqslant r\}.$$

For our analysis of agglomerative clustering, we repeatedly use the volume argument stated in Lemma 3.2. This argument provides an upper bound on the minimum distance between two points from a finite set of points lying inside the union of finitely many balls. For the application of this argument, the following definition is crucial.

Definition 3.1. *Let* $k \in \mathbb{N}$ *and* $r \in \mathbb{R}$. *Then, a set* $X \subset \mathbb{R}^d$ *is called* (k, r)-coverable, *if there exist* $y_1, \dots, y_k \in \mathbb{R}^d$ *with*

$$X \subseteq \bigcup_{i=1}^{k} B_r^d(y_i).$$

Lemma 3.2. *Let* $k \in \mathbb{N}$, $r \in \mathbb{R}$ *and* $P \subset \mathbb{R}^d$ *be finite and* (k, r)*-coverable with* $|P| > k$. *Then, there exist distinct* $p, q \in P$ *such that*

$$\|p - q\| \leqslant 4r \sqrt[d]{\frac{k}{|P|}}.$$

Proof. Let $Z \subset \mathbb{R}^d$ with $|Z| = k$ and $P \subset \bigcup_{z \in Z} B_r^d(z)$. We define δ to be the minimum distance between two points of P, i. e.,

$$\delta := \min_{\substack{p,q \in P \\ p \neq q}} \|p - q\|.$$

That is, we have to show that $\delta \leqslant 4r \sqrt[d]{\frac{k}{|P|}}$. We assume for contradiction that $u := 4r \sqrt[d]{\frac{k}{|P|}} < \delta$. Since $|P| > k$ there exists $z \in Z$ with

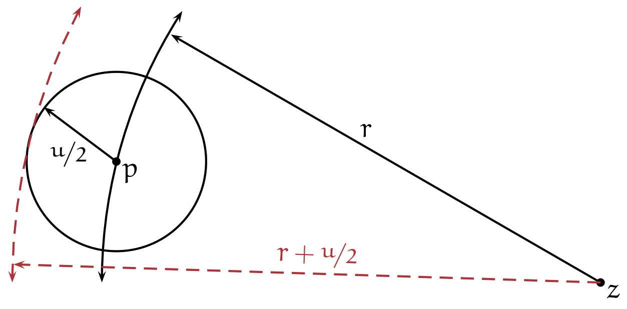


Figure 2: The volume argument.

$\left|B_r^d(z) \cap P\right| \geqslant 2$. It follows that $\delta \leqslant 2r$ and hence, $\frac{u}{2} < r$. Note that for any $y \in \mathbb{R}^d$, $\tau \in \mathbb{R}$, and any norm $\|\cdot\|$,

$$\mathrm{Vol}\left(B_\tau^d(y)\right) = \tau^d \cdot V_d,$$

where V_d is the volume of the d-dimensional unit ball $B_1^d(0)$ (see Corollary 6.2.15 in [Webster, 1994]). Therefore,

$$\mathrm{Vol}\left(\bigcup_{z \in Z} B_{r+u/2}^d(z)\right) < \sum_{z \in Z} \mathrm{Vol}\left(B_{2r}^d(z)\right) = k(2r)^d \cdot V_d.$$

Furthermore, since any $p \in P$ is contained in a ball $B_r^d(z)$ for some $z \in Z$, we conclude that any ball $B_{u/2}^d(p)$ for $p \in P$ is contained in a ball $B_{r+u/2}^d(z)$ for some $z \in Z$ (cf. Figure 2). Thus,

$$\mathrm{Vol}\left(\bigcup_{p \in P} B_{u/2}^d(p)\right) \leqslant \mathrm{Vol}\left(\bigcup_{z \in Z} B_{r+u/2}^d(z)\right) < k(2r)^d \cdot V_d. \tag{1}$$

Since $u < \delta$, for any distinct $p, q \in P$, we conclude that

$$B_{u/2}^d(p) \cap B_{u/2}^d(q) = \varnothing.$$

Therefore, using the definition of u, the total volume of the $|P|$ balls $B_{u/2}^d(p)$ is given by

$$\mathrm{Vol}\left(\bigcup_{p \in P} B_{u/2}^d(p)\right) = |P| \left(\frac{u}{2}\right)^d V_d = k(2r)^d \cdot V_d.$$

This however contradicts Inequality (1) and we obtain $\delta \leqslant u$, which proves the lemma. □

Furthermore, we need the following covering lemma that is a corollary of Theorem 1.2 in [Naszódi, 2010].

Lemma 3.3. *Let $\lambda, \tau \in \mathbb{R}$ with $\lambda, \tau > 0$. Then, for arbitrary metrics that are based on a norm, a ball of radius τ can be covered by $\left\lceil \left(\frac{3}{\lambda}\right)^d \right\rceil$ balls of radius $\lambda\tau$.*

Algorithm 3.1: AgglomerativeDiscreteRadius(X)

```
Input: finite set of input points $X \subset \mathbb{R}^d$

$\mathcal{C}_{|X|} \longleftarrow \{\{x\} \mid x \in X\}$
for $i = |X| - 1, \dots, 1$ do
    find distinct clusters $A, B \in \mathcal{C}_{i+1}$ minimizing $\mathrm{drad}(A \cup B)$
    $\mathcal{C}_i \longleftarrow (\mathcal{C}_{i+1} \setminus \{A, B\}) \cup \{A \cup B\}$
end
return $\mathcal{C}_{|X|}, \dots, \mathcal{C}_1$
```

3.2 DISCRETE k-CENTER CLUSTERING

The agglomerative algorithm for the discrete k-center problem is stated as Algorithm 3.1. Given a finite set of input points $X \subset \mathbb{R}^d$, the algorithm computes hierarchical k-clusterings for all values of k between 1 and $|X|$. We denote them by $\mathcal{C}_{|X|}, \dots, \mathcal{C}_1$. Throughout this section, *cost* always means discrete radius cost and opt_k refers to the cost of an optimal discrete k-center clustering of $X \subset \mathbb{R}^d$ where $k \in \mathbb{N}$ with $k \leqslant |X|$. That is, opt_k denotes the cost of an optimal solution to Problem 2.9. Since any cluster C is contained in a ball of radius $\mathrm{drad}(C)$, we have that X is $(k, \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_k))$-coverable for any k-clustering $\mathcal{C}_k$ of X. It follows that X is (k, opt_k)-coverable. This fact, as well as the following observation about the greedy strategy of Algorithm 3.1, will be used frequently in our analysis.

Observation 3.4. *The cost of all computed clusterings is equal to the discrete radius of the cluster created last. Furthermore, the discrete radius of the union of any two clusters is always an upper bound for the cost of the clustering to be computed next.*

The following theorem states our result for the discrete k-center problem.

Theorem 3.5. *Let $X \subset \mathbb{R}^d$ be a finite set of points. Then, for arbitrary metrics that are based on a norm and for all $k \in \mathbb{N}$ with $k \leqslant |X|$, the partition $\mathcal{C}_k$ of X into k clusters as computed by Algorithm 3.1 satisfies*

$$\mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_k) < (20d + 2\log_2(k) + 2) \cdot \mathrm{opt}_k,$$

where opt_k denotes the cost of an optimal solution to Problem 2.9.

We prove Theorem 3.5 in two steps. First, Proposition 3.6 in Section 3.2.1 provides an upper bound to the cost of the intermediate $2k$-clustering. This upper bound is independent of k and $|X|$, only linear in d and may be of independent interest. In its proof, we use the volume argument from Lemma 3.2 to bound the distance between the centers of pairs of remaining clusters.

Second, in Section 3.2.2, we analyze the remaining k merge steps of Algorithm 3.1 down to the computation of the k-clustering. There, we no longer need to apply Lemma 3.2 to bound the distance between two cluster centers. Instead, the volume argument is replaced by a very simple bound that is already sufficient to obtain our result.

3.2.1 ANALYSIS OF THE $2k$-CLUSTERING

Proposition 3.6. *Let $X \subset \mathbb{R}^d$ be finite. Then, for all $k \in \mathbb{N}$ with $2k \leqslant |X|$, the partition $\mathcal{C}_{2k}$ of X into $2k$ clusters as computed by Algorithm 3.1 satisfies*

$$\mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_{2k}) < 20d \cdot \mathrm{opt}_k,$$

where opt_k denotes the cost of an optimal solution to Problem 2.9.

To prove Proposition 3.6, we divide the merge steps of Algorithm 3.1 into phases, each reducing the number of remaining clusters by one fourth. The following lemma bounds the increase of the cost during a single phase by an additive term.

Lemma 3.7. *Let $m \in \mathbb{N}$ with $2k < m \leqslant |X|$. Then,*

$$\mathrm{cost}_{\mathrm{drad}}\left(\mathcal{C}_{\lfloor \frac{3m}{4} \rfloor}\right) \leqslant \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_m) + 4\sqrt[d]{\frac{2k}{m}} \cdot \mathrm{opt}_k \,.$$

Proof. Let $t := \lfloor \frac{3m}{4} \rfloor$. Then, $\mathcal{C}_m \cap \mathcal{C}_{t+1}$ is the set of clusters from $\mathcal{C}_m$ that still exist $\lceil \frac{m}{4} \rceil - 1 < \frac{m}{4}$ merge steps after the computation of $\mathcal{C}_m$. In each iteration of its loop, the algorithm can merge at most two clusters from $\mathcal{C}_m$. Thus,

$$|\mathcal{C}_m \cap \mathcal{C}_{t+1}| > m - 2\frac{m}{4} = \frac{m}{2}.$$

Let $\tau := \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_m)$. From every cluster $C \in \mathcal{C}_m$, we fix a center $p_C \in C$ with $C \subset B^d_\tau(p_C)$ and define $P := \{p_C \,|\, C \in \mathcal{C}_m \cap \mathcal{C}_{t+1}\}$. Then,

$$|P| = |\mathcal{C}_m \cap \mathcal{C}_{t+1}| > \frac{m}{2} > k.$$

Since X is (k, opt_k)-coverable, so is $P \subset X$. Then, by Lemma 3.2, there exist distinct $A, B \in \mathcal{C}_m \cap \mathcal{C}_{t+1}$ such that

$$\|p_A - p_B\| \leqslant 4\sqrt[d]{\frac{2k}{m}} \cdot \mathrm{opt}_k \,.$$

Therefore, the distance from p_A to any $q \in B$ is at most $4\sqrt[d]{\frac{2k}{m}} \cdot \mathrm{opt}_k + \tau$. We conclude that merging A and B would result in a cluster whose discrete radius can be upper bounded by

$$\mathrm{drad}(A \cup B) \leqslant \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_m) + 4\sqrt[d]{\frac{2k}{m}} \cdot \mathrm{opt}_k \,.$$

cf. Figure 3

The result follows from $A, B \in \mathcal{C}_{t+1}$ and Observation 3.4. □

To prove Proposition 3.6, we apply Lemma 3.7 for $\left\lceil \log_{\frac{4}{3}} \frac{|X|}{2k} \right\rceil$ consecutive phases.

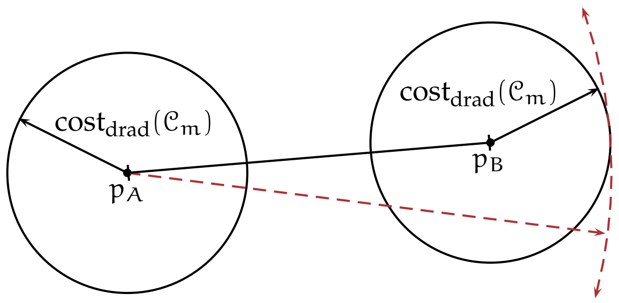

$\mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_m)$
$\mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_m)$
p_A
p_B

Figure 3: $\mathrm{drad}(A \cup B) < \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_m) + \|p_A - p_B\|$.

Proof of Proposition 3.6. Let $u := \left\lceil \log_{\frac{4}{3}} \frac{|X|}{2k} \right\rceil$ and for $i = 0, \dots, u$, define

$$m_i := \left\lceil \left(\tfrac{3}{4}\right)^i |X| \right\rceil .$$

Then, $m_u \leqslant 2k$ and $m_i > 2k$ for $i = 0, \dots, u-1$. Since

$$\left\lfloor \tfrac{3m_i}{4} \right\rfloor = \left\lfloor \tfrac{3}{4} \left\lceil \left(\tfrac{3}{4}\right)^i |X| \right\rceil \right\rfloor \leqslant \left\lfloor \left(\tfrac{3}{4}\right)^{i+1} |X| + \tfrac{3}{4} \right\rfloor \leqslant \left\lceil \left(\tfrac{3}{4}\right)^{i+1} |X| \right\rceil = m_{i+1}$$

and Algorithm 3.1 uses a greedy strategy,

$$\mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_{m_{i+1}}) \leqslant \mathrm{cost}_{\mathrm{drad}}\left(\mathcal{C}_{\left\lfloor \frac{3m_i}{4} \right\rfloor}\right)$$

for $i = 0, \dots, u-1$. Combining this inequality with Lemma 3.7 (applied to $m = m_i$), we obtain

$$\mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_{m_{i+1}}) \leqslant \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_{m_i}) + 4\sqrt[d]{\frac{2k}{m_i}} \cdot \mathrm{opt}_k .$$

By repeatedly applying this inequality for $i = 0, \dots, u-1$ and using $\mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_{2k}) \leqslant \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_{m_u})$ and $\mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_{m_0}) = 0$, we deduce

$$\begin{aligned} \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_{2k}) &\leqslant \sum_{i=0}^{u-1} \left(4\sqrt[d]{\frac{2k}{m_i}} \cdot \mathrm{opt}_k \right) \\ &\leqslant 4\sqrt[d]{\frac{2k}{|X|}} \cdot \sum_{i=0}^{u-1} \sqrt[d]{\left(\frac{4}{3}\right)^i} \cdot \mathrm{opt}_k . \end{aligned}$$

Solving the geometric series and using $u - 1 < \log_{\frac{4}{3}} \frac{|X|}{2k}$ leads to

$$\mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_{2k}) \leqslant 4\sqrt[d]{\frac{2k}{|X|}} \cdot \frac{\sqrt[d]{\left(\frac{4}{3}\right)^u} - 1}{\sqrt[d]{\frac{4}{3}} - 1} \cdot \mathrm{opt}_k < \frac{4\sqrt[d]{\frac{4}{3}}}{\sqrt[d]{\frac{4}{3}} - 1} \cdot \mathrm{opt}_k . \tag{2}$$

By taking only the first two terms of the series expansion of the exponential function, we get $\sqrt[d]{\frac{4}{3}} = e^{\frac{\ln \frac{4}{3}}{d}} > 1 + \frac{\ln \frac{4}{3}}{d}$. Substituting this bound into Inequality (2) yields

$$\mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_{2k}) < \frac{4\sqrt[d]{\frac{4}{3}}}{\ln \frac{4}{3}} d \cdot \mathrm{opt}_k < 20d \cdot \mathrm{opt}_k .$$

□

3.2.2 ANALYSIS OF THE REMAINING MERGE STEPS

The analysis of the remaining merge steps introduces the $O(\log k)$ term to the approximation factor of our result. It is similar to the analysis used in the proof of Proposition 3.6. Again, we divide the merge steps into phases. This time however, one phase consists of one half of the remaining merge steps. Furthermore, we replace the volume argument from Lemma 3.2 with a simpler bound. This bound results from the observation that, as long as there are more than k clusters left, we are able to find a pair of clusters whose centers lie in the same cluster of an optimal k-clustering. That is, the distance between these centers is at most two times the discrete radius of the common cluster. The following lemma bounds the increase of the cost during a single phase.

Lemma 3.8. *Let* $m \in \mathbb{N}$ *with* $k < m \leqslant |X|$. *Then,*

$$\mathrm{cost}_{\mathrm{drad}}\left(\mathcal{C}_{\lfloor \frac{m+k}{2} \rfloor}\right) \leqslant \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_m) + 2\,\mathrm{opt}_k\,.$$

Proof. Let $t := \lfloor \frac{m+k}{2} \rfloor$. Then, $\mathcal{C}_m \cap \mathcal{C}_{t+1}$ is the set of clusters from $\mathcal{C}_m$ that still exist $\lceil \frac{m-k}{2} \rceil - 1 < \frac{m-k}{2}$ merge steps after the computation of $\mathcal{C}_m$. In each iteration of its loop, the algorithm can merge at most two clusters from $\mathcal{C}_m$. Thus,

$$|\mathcal{C}_m \cap \mathcal{C}_{t+1}| > m - 2\frac{m-k}{2} = k.$$

Let $\tau := \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_m)$. From every cluster $C \in \mathcal{C}_m$, we fix a center $p_C \in C$ with $C \subset B^d_\tau(p_C)$ and define $P := \{p_C \mid C \in \mathcal{C}_m \cap \mathcal{C}_{t+1}\}$. Then,

$$|P| = |\mathcal{C}_m \cap \mathcal{C}_{t+1}| > k.$$

Since X is (k, opt_k)-coverable, so is $P \subset X$. It follows that there exist distinct $A, B \in \mathcal{C}_m \cap \mathcal{C}_{t+1}$ such that p_A and p_B are contained in the same ball of radius opt_k, i. e.,

$$\|p_A - p_B\| \leqslant 2\,\mathrm{opt}_k\,.$$

Then, the distance from p_A to any $q \in B$ is at most $2\,\mathrm{opt}_k + \tau$. We conclude that merging A and B would result in a cluster whose discrete radius can be upper bounded by

$$\mathrm{drad}(A \cup B) \leqslant \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_m) + 2\,\mathrm{opt}_k\,.$$

cf. Figure 3

The result follows using $A, B \in \mathcal{C}_{t+1}$ and Observation 3.4. □

To prove Theorem 3.5, we apply Lemma 3.8 for about $\log k$ consecutive phases.

Algorithm 3.2: AgglomerativeRadius(X)

Input: finite set of input points $X \subset \mathbb{R}^d$

```
$\mathcal{C}_{|X|} \longleftarrow \{\{x\} \mid x \in X\}$
for $i = |X|-1, \ldots, 1$ do
    find distinct clusters $A, B \in \mathcal{C}_{i+1}$ minimizing $\mathrm{rad}(A \cup B)$
    $\mathcal{C}_i \longleftarrow (\mathcal{C}_{i+1} \setminus \{A, B\}) \cup \{A \cup B\}$
end
return $\mathcal{C}_{|X|}, \ldots, \mathcal{C}_1$
```

Proof of Theorem 3.5. Let $u := \lfloor \log_2(k) \rfloor + 1$. Then,

$$\log_2 k < u \leqslant \log_2(k) + 1.$$

Furthermore, for $i = 0, \ldots, u$ we define

$$m_i := k + \left\lfloor \left(\tfrac{1}{2}\right)^i k \right\rfloor.$$

Then, $m_u = k$ and $m_i > k$ for $i = 0, \ldots, u-1$. Since

$$\begin{aligned}\left\lfloor \tfrac{m_i+k}{2} \right\rfloor &= \left\lfloor \tfrac{1}{2}\left(2k + \left\lfloor \left(\tfrac{1}{2}\right)^i k \right\rfloor\right) \right\rfloor = k + \left\lfloor \tfrac{1}{2} \left\lfloor \left(\tfrac{1}{2}\right)^i k \right\rfloor \right\rfloor \\ &\leqslant k + \left\lfloor \left(\tfrac{1}{2}\right)^{i+1} k \right\rfloor = m_{i+1}\end{aligned}$$

and Algorithm 3.1 uses a greedy strategy,

$$\mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_{m_{i+1}}) \leqslant \mathrm{cost}_{\mathrm{drad}}\left(\mathcal{C}_{\left\lfloor \frac{m_i+k}{2} \right\rfloor}\right)$$

for $i = 0, \ldots, u-1$. Combining this inequality with Lemma 3.8 (applied to $m = m_i$), we obtain

$$\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{m_{i+1}}) \leqslant \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_{m_i}) + 2\,\mathrm{opt}_k\,.$$

By repeatedly applying this inequality for $i = 0, \ldots, u-1$ and using $m_0 = 2k$ and $m_u = k$, we deduce

$$\begin{aligned}\mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_k) &\leqslant \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_{2k}) + \sum_{i=0}^{u-1} 2\,\mathrm{opt}_k \\ &\leqslant \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_{2k}) + (2\log_2(k) + 2) \cdot \mathrm{opt}_k\,.\end{aligned}$$

Hence, the result follows using Proposition 3.6. □

3.3 k-center clustering

The agglomerative algorithm for the k-center problem is stated as Algorithm 3.2. The only difference to Algorithm 3.1 is the minimization of the radius instead of the discrete radius in the minimization step. In the following, *cost* always means radius cost and opt_k refers to the cost of an optimal k-center clustering of $X \subset \mathbb{R}^d$ where $k \in \mathbb{N}$ with $k \leqslant |X|$. That is, opt_k denotes the cost of an optimal solution to Problem 2.8. Analogously to the discrete case, any cluster C is contained in a ball of radius $\text{drad}(C)$ and thus, the set X is (k, opt_k)-coverable.

Observation 3.9. *The cost of all computed clusterings is equal to the radius of the cluster created last. Furthermore, the radius of the union of any two clusters is always an upper bound for the cost of the clustering to be computed next.*

This is analogous to Observation 3.4.

The following theorem states our result for the k-center problem.

Theorem 3.10. *Let $X \subset \mathbb{R}^d$ be a finite set of points. Then, for arbitrary metrics that are based on a norm and for all $k \in \mathbb{N}$ with $k \leqslant |X|$, the partition $\mathcal{C}_k$ of X into k clusters as computed by Algorithm 3.2 satisfies*

$$\text{cost}_{\text{rad}}(\mathcal{C}_k) = O(\log k) \cdot \text{opt}_k,$$

where opt_k denotes the cost of an optimal solution to Problem 2.8, and the constant hidden in the O-notation is singly exponential in the dimension d.

As in the proof of Theorem 3.5, we first bound the cost of the intermediate $2k$-clustering. However, we have to apply a different analysis. As a consequence, the dependency on the dimension increases from linear and additive to a singly exponential factor.

3.3.1 Analysis of the $2k$-clustering

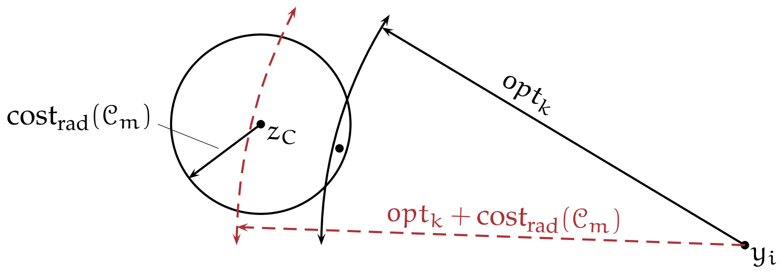
Proposition 3.11. *Let $X \subset \mathbb{R}^d$ be finite. Then, for all $k \in \mathbb{N}$ with $2k \leqslant |X|$, the partition $\mathcal{C}_{2k}$ of X into $2k$ clusters as computed by Algorithm 3.2 satisfies*

$$\text{cost}_{\text{rad}}(\mathcal{C}_{2k}) < 24d \cdot e^{24d} \cdot \text{opt}_k,$$

where opt_k denotes the cost of an optimal solution to Problem 2.8.

Just as in the analysis of Algorithm 3.1, we divide the merge steps of Algorithm 3.2 into phases, such that in each phase the number of remaining clusters is reduced by one fourth. Like in the discrete case, the input points are (k, opt_k)-coverable. However, centers corresponding to an intermediate solution computed by Algorithm 3.2 need not be covered by the k balls induced by an optimal solution. As a consequence, we are no longer able to apply Lemma 3.2 on the centers as in the discrete case.

To bound the increase of the cost during a single phase, we cover the remaining clusters at the beginning of a phase by a set of overlapping balls. Each of the clusters is completely contained in one of these balls that all have the same radius. Furthermore, the number of remaining clusters will be at least twice the number of these balls. It follows that there are many pairs of clusters that are contained in the



$\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_m)$
z_C
opt_k
$\mathrm{opt}_k + \mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_m)$
y_i

Figure 4: Intermediate centers.

same ball. Then, as long as the existence of at least one such pair can be guaranteed, the radius of the cluster created next can be bounded by the radius of the covering balls. The following lemma will be used to bound the increase of the cost during a single phase.

Lemma 3.12. *Let* $m \in \mathbb{N}$ *with* $2k < m \leqslant |X|$. *Then,*

$$\mathrm{cost}_{\mathrm{rad}}\left(\mathcal{C}_{\lfloor \frac{3m}{4} \rfloor}\right) < \left(1 + 6\sqrt[d]{\frac{2k}{m}}\right) \mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_m) + 6\sqrt[d]{\frac{2k}{m}}\,\mathrm{opt}_k\,.$$

Proof. Let $\mathcal{P} = \{P_1, \dots, P_k\}$ be an optimal k-center clustering of X. For each $i \in \{1, \dots, k\}$ we fix a point $y_i \in \mathbb{R}^d$ such that

$$P_i \subset B_{\mathrm{opt}_k}(y_i).$$

Furthermore, for any $C \in \mathcal{C}_m$ let $z_C \in \mathbb{R}^d$ such that

$$C \subset B_{\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_m)}(z_C).$$

With $\tau := \mathrm{opt}_k + \mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_m)$ it follows that each z_C is contained in at least one of the balls $B_\tau(y_i)$ for $i \in \{1, \dots, k\}$ as illustrated in Figure 4.

Applying Lemma 3.3 with $\lambda = \frac{3}{\sqrt[d]{\lfloor \frac{m}{2k} \rfloor}}$ yields that each of the balls $B_\tau(y_i)$ for $i = 1, \dots, k$ can be covered by

$$\ell := \left\lfloor \frac{m}{2k} \right\rfloor \leqslant \frac{m}{2k}$$

balls of radius $\varepsilon := \frac{3\tau}{\sqrt[d]{\lfloor \frac{m}{2k} \rfloor}}$. Therefore, there exist

$$k\ell \leqslant \frac{m}{2}$$

centers $v_1, \dots, v_{k\ell} \in \mathbb{R}^d$ such that each z_C for $C \in \mathcal{C}_m$ is contained in at least one of the balls $B_\varepsilon(v_1), \dots, B_\varepsilon(v_{k\ell})$. For $i = 1, \dots, k\ell$, we define

$$V_i := B_{\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_m)+\varepsilon}(v_i).$$

Then, any cluster $C \in \mathcal{C}_m$ is contained in at least one of the balls $V_1, \dots, V_{k\ell}$ (cf. Figure 5).

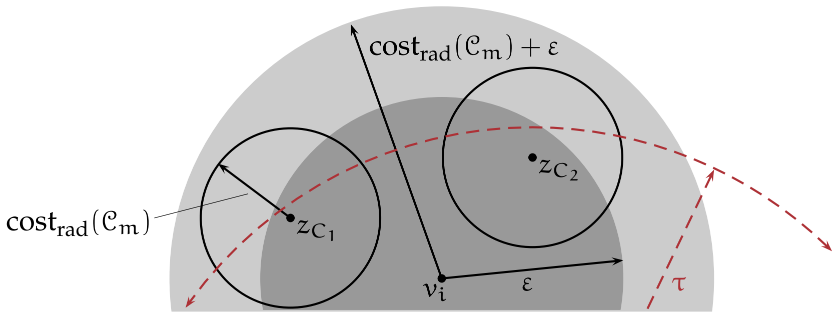

$\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_m) + \varepsilon$
$\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_m)$
z_{C_1}
z_{C_2}
v_i
ε
τ

Figure 5: Covering centers and clusters.

As in the proof of Lemma 3.7 we define $t := \lfloor \frac{3m}{4} \rfloor$ and deduce $|\mathcal{C}_m \cap \mathcal{C}_{t+1}| > \frac{m}{2}$. Since $k\ell \leqslant \frac{m}{2}$, there exist two clusters $A, B \in \mathcal{C}_m \cap \mathcal{C}_{t+1}$ that are contained in the same ball V_i with $i \in \{1, \ldots, k\ell\}$. Therefore, merging clusters A and B would result in a cluster whose radius can be upper bounded by

$$\mathrm{rad}(A \cup B) \leqslant \mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_m) + \varepsilon.$$

cf. Figure 5

Using Observation 3.9 and the fact that A and B are part of the clustering $\mathcal{C}_{t+1}$, we can upper bound the cost of $\mathcal{C}_t$ by

$$\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_t) \leqslant \mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_m) + \varepsilon.$$

It remains to show

$$\varepsilon < 6 \sqrt[d]{\frac{2k}{m}} (\mathrm{opt}_k + \mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_m)).$$

However, since $\frac{m}{2k} > 1$, it follows $\frac{m}{2k} < 2 \lfloor \frac{m}{2k} \rfloor$. Thus,

$$\sqrt[d]{\frac{m}{2k}} < \sqrt[d]{2 \left\lfloor \frac{m}{2k} \right\rfloor} \leqslant 2 \sqrt[d]{\left\lfloor \frac{m}{2k} \right\rfloor}$$

and we conclude

$$\frac{3}{\sqrt[d]{\lfloor \frac{m}{2k} \rfloor}} < 6 \sqrt[d]{\frac{2k}{m}}.$$

□

To prove Proposition 3.11, we apply Lemma 3.12 for $\left\lceil \log_{\frac{4}{3}} \frac{|X|}{2k} \right\rceil$ consecutive phases.

Proof of Proposition 3.11. Analogously to the proof of Proposition 3.6, we define $u := \left\lceil \log_{\frac{4}{3}} \frac{|X|}{2k} \right\rceil$ and $m_i := \left\lceil \left(\frac{3}{4}\right)^i |X| \right\rceil$ for $i = 0, \ldots, u$. Then, $m_u \leqslant 2k$ and for $i = 0, \ldots, u-1$, $m_i > 2k$ and $\left\lfloor \frac{3m_i}{4} \right\rfloor \leqslant m_{i+1}$. Using Lemma 3.12, for $i = 0, \ldots, u-1$ we deduce

$$\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_{m_{i+1}}) < \left(1 + 6 \sqrt[d]{\frac{2k}{m_i}}\right) \mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_{m_i}) + 6 \sqrt[d]{\frac{2k}{m_i}} \,\mathrm{opt}_k .$$

By repeatedly applying this inequality for $i = 0, \dots, u-1$ and using $\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_{2k}) \leqslant \mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_{m_u})$ and $\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_{m_0}) = 0$, we get

$$\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_{2k}) < \sum_{i=0}^{u-1} \left(6 \sqrt[d]{\frac{2k}{m_i}} \prod_{j=i+1}^{u-1} \left(1 + 6 \sqrt[d]{\frac{2k}{m_j}} \right) \right) \mathrm{opt}_k \, .$$

Using $m_i \geqslant \left(\frac{3}{4}\right)^i |X|$, we obtain

$$\begin{aligned} &\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_{2k}) \\ &\leqslant 6\,\mathrm{opt}_k \sqrt[d]{\frac{2k}{|X|}} \cdot \sum_{i=0}^{u-1} \left(\left(\frac{4}{3}\right)^{\frac{i}{d}} \prod_{j=i+1}^{u-1} \left(1 + 6 \sqrt[d]{\frac{2k}{|X|}} \left(\frac{4}{3}\right)^{\frac{j}{d}} \right) \right) . \end{aligned}$$

Substituting $u-1-i$ for i yields

$$\begin{aligned} &\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_{2k}) \\ &= 6\,\mathrm{opt}_k \sqrt[d]{\frac{2k}{|X|}} \left(\frac{4}{3}\right)^{\frac{u-1}{d}} \\ &\quad \cdot \sum_{i=0}^{u-1} \left(\left(\frac{3}{4}\right)^{\frac{i}{d}} \prod_{j=u-i}^{u-1} \left(1 + 6 \sqrt[d]{\frac{2k}{|X|}} \left(\frac{4}{3}\right)^{\frac{u-1}{d}} \left(\frac{3}{4}\right)^{\frac{u-1-j}{d}} \right) \right) . \end{aligned}$$

Using $u - 1 < \log_{\frac{4}{3}} \frac{|X|}{2k}$, we deduce

$$\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_{2k}) < 6\,\mathrm{opt}_k \cdot \sum_{i=0}^{u-1} \left(\left(\frac{3}{4}\right)^{\frac{i}{d}} \prod_{j=0}^{i-1} \left(1 + 6 \left(\frac{3}{4}\right)^{\frac{j}{d}} \right) \right) . \qquad (3)$$

By taking only the first two terms of the series expansion of the exponential function, we get $1 + 6\left(\frac{3}{4}\right)^{\frac{j}{d}} < e^{6\left(\frac{3}{4}\right)^{\frac{j}{d}}}$ and therefore,

$$\prod_{j=0}^{i-1} \left(1 + 6 \left(\frac{3}{4}\right)^{\frac{j}{d}} \right) < \prod_{j=0}^{i-1} \left(e^{6\left(\frac{3}{4}\right)^{\frac{j}{d}}} \right) = e^{6 \sum_{j=0}^{i-1} \left(\frac{3}{4}\right)^{\frac{j}{d}}} . \qquad (4)$$

The sum in the exponent can be bounded by the infinite geometric series

$$\sum_{j=0}^{\infty} \left(\frac{3}{4}\right)^{\frac{j}{d}} < \frac{1}{1 - \left(\frac{3}{4}\right)^{\frac{1}{d}}} \leqslant 4d. \qquad (5)$$

Here, the last inequality follows by upper bounding the convex function $f(x) = \left(\frac{3}{4}\right)^x$ in the interval $[0, 1]$ by the line through $f(0) = 1$ and $f(1) = \frac{3}{4}$, i. e., $\left(\frac{3}{4}\right)^x \leqslant 1 - \frac{x}{4}$. Putting Inequalities (3), (4) and (5) together then gives

$$\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_{2k}) < 6\,\mathrm{opt}_k \cdot \sum_{i=0}^{u-1} \left(\left(\frac{3}{4}\right)^{\frac{i}{d}} e^{24d} \right) < 24d \cdot e^{24d} \cdot \mathrm{opt}_k ,$$

where the last inequality follows by using Inequality (5) again. □

3.3.2 CONNECTED SUBSETS

The analysis of the remaining merge steps from the discrete k-center case (cf. Section 3.2.2) is not transferable to the k-center case. Again, as in the proof of Proposition 3.11, we are no longer able to derive a simple additive bound on the increase of the cost when merging two clusters. In order to preserve the logarithmic dependency of the approximation factor on k, we show that it is sufficient to analyze Algorithm 3.2 on a subset $Y \subseteq X$ satisfying a certain connectivity property. Using this property, we are able to apply a combinatorial approach that relies on the number of merge steps left.

We start by defining the connectivity property that will be used to relate clusters to an optimal k-clustering.

Definition 3.13. *Let $Z \subseteq \mathbb{R}^d$ and $r \in \mathbb{R}$. Two sets $A, B \subseteq \mathbb{R}^d$ are called* (Z, r)-connected *if there exists a point $z \in Z$ with*

$$B_r^d(z) \cap A \neq \varnothing \quad \textit{and} \quad B_r^d(z) \cap B \neq \varnothing.$$

Note that for any two (Z, r)-connected clusters A, B, we have

$$\operatorname{rad}(A \cup B) \leqslant \operatorname{rad}(A) + 2\operatorname{rad}(B) + 2r. \tag{6}$$

To see this, let $y \mathbb{R}^d$ with $A \subset B^d_{\operatorname{rad}(A)}(y)$, $x_a \in B_r^d(z) \cap A$ and $x_b \in B_r^d(z) \cap B$. Then, the distance between y and x_a is at most $\operatorname{rad} A$, the distance between x_a and x_b is at most $2r$ and the distance between x_b and any point $x \in B$ is at most $2\operatorname{rad}(B)$.

Next, we show that for any data set X we can bound the cost of the k-clustering computed by Algorithm 3.2 by the cost of the ℓ-clustering computed on a connected subset $Y \subseteq X$ for a proper $\ell \leqslant k$.

Lemma 3.14. *Let $X \subset \mathbb{R}^d$ be finite and $k \in \mathbb{N}$ with $k \leqslant |X|$. Then, there exists a subset $Y \subseteq X$, a number $\ell \in \mathbb{N}$ with $\ell \leqslant k$ and $\ell \leqslant |Y|$, and a set $Z \subset \mathbb{R}^d$ with $|Z| = \ell$ such that:*

1. *Y is (ℓ, opt_k)-coverable;*
2. *$\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_k) \leqslant \mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_\ell)$;*
3. *For all $n \in \mathbb{N}$ with $\ell < n \leqslant |Y|$, every cluster in $\mathcal{P}_n$ is (Z, opt_k)-connected to another cluster in $\mathcal{P}_n$.*

Here, the collection $\mathcal{P}_1, \dots, \mathcal{P}_{|Y|}$ denotes the hierarchical clustering computed by Algorithm 3.2 on input Y.

Proof. To define Y, Z, and ℓ we consider the $(k+1)$-clustering computed by Algorithm 3.2 on input X. We know that $X = \bigcup_{A \in \mathcal{C}_{k+1}} A$ is (k, opt_k)-coverable. Let $E \subseteq \mathcal{C}_{k+1}$ be a minimal subset such that $\bigcup_{A \in E} A$ is $(|E| - 1, \mathrm{opt}_k)$-coverable, i.e., for all sets $F \subseteq \mathcal{C}_{k+1}$ with $|F| < |E|$ the union $\bigcup_{A \in F} A$ is not $(|F| - 1, \mathrm{opt}_k)$-coverable. Since a set F of size 1 cannot be $(|F| - 1, \mathrm{opt}_k)$-coverable, we deduce $|E| \geqslant 2$.

Let $Y := \bigcup_{A \in E} A$ and $\ell := |E| - 1$. Then, $\ell \leqslant k$ and Y is (ℓ, opt_k)-coverable. This establishes the first property of the lemma. It follows that there exists a set $Z \subset \mathbb{R}^d$ with $|Z| = \ell$ and $Y \subset \bigcup_{z \in Z} B^d_{\mathrm{opt}_k}(z)$.

Since Y is the union of the clusters from $E \subseteq \mathcal{C}_{k+1}$, each merge step between the computation of $\mathcal{C}_{|X|}$ and $\mathcal{C}_{k+1}$ merges either two clusters $A, B \subset Y$ or two clusters $A, B \subset X \setminus Y$. In the following, we consider the restriction of the clusterings $\mathcal{C}_{k+1}, \ldots, \mathcal{C}_{|X|}$ to the input data set $Y \subset X$. We need to deal with the resulting clusterings as if they were computed by Algorithm 3.2 on input Y. This is the part of our analysis where we have to be more precise on the tie-breaking strategy as briefly mentioned at the beginning of this chapter. If on input Y, the minimization step of Algorithm 3.2 is not always unambiguous, the computed clusterings may differ from the restriction of $\mathcal{C}_{k+1}, \ldots, \mathcal{C}_{|X|}$ to Y. Thus, for the analysis of any fixed run on input X we simply assume that on input Y Algorithm 3.2 performs the same sequence of merge steps that ignoring the merge steps inside $X \setminus Y$ lead to $\mathcal{C}_{k+1}, \ldots, \mathcal{C}_{|X|}$ in the fixed run on input X.

2^Y denotes the power set of Y.

We let $\mathcal{P}_1, \ldots, \mathcal{P}_{|Y|}$ be the hierarchical clustering computed by Algorithm 3.2 on input Y. Then, we have $\mathcal{P}_{\ell+1} = E = \mathcal{C}_{k+1} \cap 2^Y$. Thus, $\mathcal{P}_{\ell+1} \subseteq \mathcal{C}_{k+1}$. To compute $\mathcal{P}_\ell$, on input Y, Algorithm 3.2 merges two clusters from $\mathcal{P}_{\ell+1}$ that minimize the radius of the resulting cluster. Analogously, on input X, Algorithm 3.2 merges two clusters from $\mathcal{C}_{k+1}$ to compute $\mathcal{C}_k$. Since $\mathcal{P}_{\ell+1} \subseteq \mathcal{C}_{k+1}$, Observation 3.9 implies $\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_k) \leqslant \mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_\ell)$, thus proving the second property of the lemma.

It remains to show that for all $n \in \mathbb{N}$ with $\ell < n \leqslant |Y|$ it holds that every cluster in $\mathcal{P}_n$ is (Z, opt_k)-connected to another cluster in $\mathcal{P}_n$ (the third property of the lemma). By the definition of Z, every cluster in $\mathcal{P}_n$ intersects at least one ball $B^d_{\mathrm{opt}_k}(z)$ for $z \in Z$. Therefore, it is enough to show that each ball $B^d_{\mathrm{opt}_k}(z)$ intersects at least two clusters from $\mathcal{P}_n$. We first show this property for $n = \ell + 1$. For $\ell = 1$, this follows from the fact that $B^d_{\mathrm{opt}_k}(z)$ with $Z = \{z\}$ has to contain both clusters from $\mathcal{P}_2$. For $\ell > 1$, we would otherwise be able to remove one cluster from $\mathcal{P}_{\ell+1}$ and get ℓ clusters whose union is $(\ell - 1, \mathrm{opt}_k)$-coverable. However, this contradicts the definition of $E = \mathcal{P}_{\ell+1}$ as a minimal subset with this property.

To show the third property of the lemma for general n, let $A \in \mathcal{P}_n$ and $z \in Z$ with $B^d_{\mathrm{opt}_k}(z) \cap A \neq \emptyset$. There exists a unique cluster $\tilde{A} \in \mathcal{P}_{\ell+1}$ with $A \subseteq \tilde{A}$. Then, we have $B^d_{\mathrm{opt}_k}(z) \cap \tilde{A} \neq \emptyset$. However, we have just shown that $B^d_{\mathrm{opt}_k}(z)$ has to intersect at least two clusters from $\mathcal{P}_{\ell+1}$. Thus, there exists another cluster $\tilde{B} \in \mathcal{P}_{\ell+1}$ with $B^d_{\mathrm{opt}_k}(z) \cap \tilde{B} \neq \emptyset$. Since every cluster from $\mathcal{P}_{\ell+1}$ is a union of clusters from $\mathcal{P}_n$, there exists at least one cluster $B \in \mathcal{P}_n$ with $B \subseteq \tilde{B}$ and $B^d_{\mathrm{opt}_k}(z) \cap B \neq \emptyset$. $\square$

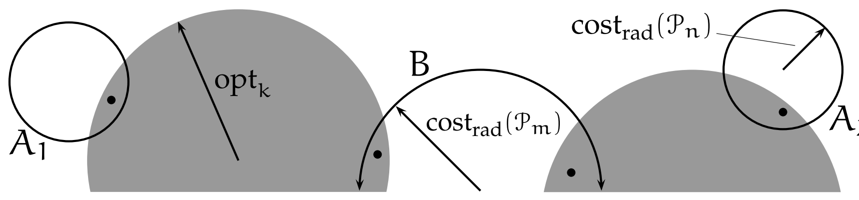

$\text{cost}_{\text{rad}}(\mathcal{P}_n)$
opt_k
B
$\text{cost}_{\text{rad}}(\mathcal{P}_m)$
A_1
A_2

Figure 6: Merging (Z, opt_k)-connected clusters.

3.3.3 Analysis of the remaining merge steps

Let Y, Z, ℓ, and $\mathcal{P}_1, \ldots, \mathcal{P}_{|Y|}$ be as given by Lemma 3.14. Then, Proposition 3.11 can be used to obtain an upper bound for the cost of $\mathcal{P}_{2\ell}$. In the following, we analyze the merge steps leading from $\mathcal{P}_{2\ell}$ to $\mathcal{P}_{\ell+1}$ and show how to obtain an upper bound for the cost of $\mathcal{P}_{\ell+1}$. As in Section 3.3.1, we analyze the merge steps in phases. The following lemma is used to bound the increase of the cost during a single phase. Note that opt_k still refers to the cost of an optimal solution on input X, not Y.

Lemma 3.15. *Let $m, n \in \mathbb{N}$ with $n \leqslant 2\ell$ and $\ell < m \leqslant n \leqslant |Y|$. If there are no two (Z, opt_k)-connected clusters in $\mathcal{P}_m \cap \mathcal{P}_n$,*

$$\text{cost}_{\text{rad}}\left(\mathcal{P}_{\left\lfloor \frac{m+\ell}{2} \right\rfloor}\right) \leqslant \text{cost}_{\text{rad}}(\mathcal{P}_m) + 2\,\text{cost}_{\text{rad}}(\mathcal{P}_n) + 2\,\text{opt}_k\,.$$

Proof. Analogously to the proof of Lemma 3.8 we define $t := \left\lfloor \frac{m+\ell}{2} \right\rfloor$ and deduce that there are strictly less than $\frac{m-\ell}{2}$ merge steps between the computations of $\mathcal{P}_m$ and $\mathcal{P}_{t+1}$. We show that there exist at least $m - \ell$ disjoint pairs of clusters from $\mathcal{P}_m$ such that the radius of their union can be upper bounded by $\text{cost}_{\text{rad}}(\mathcal{P}_m) + 2\,\text{cost}_{\text{rad}}(\mathcal{P}_n) + 2\,\text{opt}_k$. Then, at least one of these pairs still has to exist in $\mathcal{P}_{t+1}$ since in each iteration of its loop the algorithm can destroy at most two of the pairs. The lemma follows using Observation 3.9 and from the fact that there is only one merge step left.

To bound the number of the above mentioned cluster pairs, we start with a structural observation. $\mathcal{P}_m \cap \mathcal{P}_n$ is the set of clusters from $\mathcal{P}_n$ that still exist in $\mathcal{P}_m$. By our definition of Y, Z, and ℓ, we conclude that any cluster $A \in \mathcal{P}_m \cap \mathcal{P}_n$ is (Z, opt_k)-connected to another cluster $B \in \mathcal{P}_m$. If we assume that there are no two (Z, opt_k)-connected clusters in $\mathcal{P}_m \cap \mathcal{P}_n$, this implies $B \in \mathcal{P}_m \setminus \mathcal{P}_n$ (cf. Figure 6). Thus, using $A \in \mathcal{P}_n$, $B \in \mathcal{P}_m$, and Inequality (6), the radius of $A \cup B$ can be bounded by

$$\text{rad}(A \cup B) \leqslant \text{cost}_{\text{rad}}(\mathcal{P}_m) + 2\,\text{cost}_{\text{rad}}(\mathcal{P}_n) + 2\,\text{opt}_k\,. \tag{7}$$

Moreover, using a similar argument, we derive the same bound for two clusters $A_1, A_2 \in \mathcal{P}_m \cap \mathcal{P}_n$ that are (Z, opt_k)-connected to the same cluster $B \in \mathcal{P}_m \setminus \mathcal{P}_n$. That is,

$$\text{rad}(A_1 \cup A_2) \leqslant \text{cost}_{\text{rad}}(\mathcal{P}_m) + 2\,\text{cost}_{\text{rad}}(\mathcal{P}_n) + 2\,\text{opt}_k\,. \tag{8}$$

Next, we show that there exist at least $\left\lceil\frac{|\mathcal{P}_m\cap\mathcal{P}_n|}{2}\right\rceil$ disjoint pairs of clusters from $\mathcal{P}_m$ such that the radius of their union can be bounded either by Inequality (7) or by Inequality (8). To do so, we first consider the pairs of clusters from $\mathcal{P}_m\cap\mathcal{P}_n$ that are (Z,opt_k)-connected to the same cluster from $\mathcal{P}_m\setminus\mathcal{P}_n$ until no candidates are left. For these pairs, we can bound the radius of their union by Inequality (8). Then, each cluster from $\mathcal{P}_m\setminus\mathcal{P}_n$ is (Z,opt_k)-connected to at most one of the remaining clusters from $\mathcal{P}_m\cap\mathcal{P}_n$. Thus, each remaining cluster $A\in\mathcal{P}_m\cap\mathcal{P}_n$ can be paired with a different cluster $B\in\mathcal{P}_m\setminus\mathcal{P}_n$ such that A and B are (Z,opt_k)-connected. For these pairs, we can bound the radius of their union by Inequality (7). Since for all pairs either one or both of the clusters come from the set $\mathcal{P}_m\cap\mathcal{P}_n$, we can lower bound the number of pairs by $\left\lceil\frac{|\mathcal{P}_m\cap\mathcal{P}_n|}{2}\right\rceil$.

To complete the proof, we show

$$m-\ell\leqslant\left\lceil\frac{|\mathcal{P}_m\cap\mathcal{P}_n|}{2}\right\rceil.$$

In each iteration of its loop, the algorithm can merge at most two clusters from $\mathcal{P}_n$. Therefore, there are at least $\left\lceil\frac{n-|\mathcal{P}_m\cap\mathcal{P}_n|}{2}\right\rceil$ merge steps between the computations of $\mathcal{P}_n$ and $\mathcal{P}_m$. Hence,

$$m\leqslant n-\left\lceil\frac{n-|\mathcal{P}_m\cap\mathcal{P}_n|}{2}\right\rceil\leqslant\frac{n}{2}+\frac{|\mathcal{P}_m\cap\mathcal{P}_n|}{2}.$$

Using $n\leqslant 2\ell$, we deduce

$$m-\ell\leqslant\frac{|\mathcal{P}_m\cap\mathcal{P}_n|}{2}.$$

□

Lemma 3.16. *Let* $n\in\mathbb{N}$ *with* $n\leqslant 2\ell$ *and* $\ell<n\leqslant|Y|$. *Then,*

$$\mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_{\ell+1})<2\left(\log_2(\ell)+2\right)\left(\mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_n)+\mathrm{opt}_k\right).$$

Proof. For $n=\ell+1$ there is nothing to show. Hence, we assume $n>\ell+1$. By the definition of Z, there exist two (Z,opt_k)-connected clusters in $\mathcal{P}_n$. Let $\tilde{n}\in\mathbb{N}$ with $\tilde{n}<n$ be maximal such that no two (Z,opt_k)-connected clusters exist in $\mathcal{P}_{\tilde{n}}\cap\mathcal{P}_n$. The number $\tilde{n}$ is well-defined since $|\mathcal{P}_1|=1$ implies $\tilde{n}\geqslant 1$. It follows that the same holds for all $m\in\mathbb{N}$ with $m\leqslant\tilde{n}$. We conclude that Lemma 3.15 is applicable for all $m\in\mathbb{N}$ with $\ell<m\leqslant\tilde{n}$.

By the definition of $\tilde{n}$ there still exist at least two (Z,opt_k)-connected clusters in $\mathcal{P}_{\tilde{n}+1}\cap\mathcal{P}_n$. Then, Observation 3.9 implies

$$\mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_{\tilde{n}})\leqslant 2\,\mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_n)+\mathrm{opt}_k. \tag{9}$$

If $\tilde{n}\leqslant\ell+1$, Inequality (9) proves the lemma. For $\tilde{n}>\ell+1$, define $u=\lceil\log_2(\tilde{n}-\ell)\rceil$ and

$$m_i=\left\lceil\left(\frac{1}{2}\right)^i(\tilde{n}-\ell)+\ell\right\rceil>\ell$$

for $i = 0, \dots, u$. Then, $m_0 = \tilde{n}$ and $m_u = \ell + 1$. Furthermore,

$$\begin{aligned}\left\lfloor \frac{m_i+\ell}{2} \right\rfloor &= \left\lfloor \frac{1}{2} \left\lceil \left(\tfrac{1}{2}\right)^i (\tilde{n}-\ell) + \ell \right\rceil + \frac{\ell}{2} \right\rfloor \leqslant \left\lfloor \frac{1}{2} \left(\left(\tfrac{1}{2}\right)^i (\tilde{n}-\ell) + \ell + 1 \right) + \frac{\ell}{2} \right\rfloor \\ &= \left\lfloor \left(\tfrac{1}{2}\right)^{i+1} (\tilde{n}-\ell) + \ell + \tfrac{1}{2} \right\rfloor \leqslant \left\lceil \left(\tfrac{1}{2}\right)^{i+1} (\tilde{n}-\ell) + \ell \right\rceil = m_{i+1}.\end{aligned}$$

Since Algorithm 3.2 uses a greedy strategy,

$$\mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_{m_{i+1}}) \leqslant \mathrm{cost}_{\mathrm{rad}}\left(\mathcal{P}_{\left\lfloor \frac{m_i+\ell}{2} \right\rfloor}\right)$$

for $i = 0, \dots, u-1$. Combining this inequality with Lemma 3.15 (applied to $m = m_i$), we obtain

$$\mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_{m_{i+1}}) \leqslant \mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_{m_i}) + 2\,\mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_n) + 2\,\mathrm{opt}_k\,.$$

By repeatedly applying this inequality for $i = 0, \dots, u-1$ and summing up the costs, we get

$$\begin{aligned}\mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_{m_u}) &< \mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_{\tilde{n}}) + 2u \cdot \left(\mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_n) + \mathrm{opt}_k\right) \\ &\overset{(9)}{<} 2(u+1) \cdot \left(\mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_n) + \mathrm{opt}_k\right).\end{aligned}$$

Since $\tilde{n} < 2\ell$, we get $u < \log_2(\ell) + 1$ and the lemma follows using $m_u = \ell + 1$. ☐

The following lemma finishes the analysis except for the last merge step.

Lemma 3.17. *Let $Y \subset \mathbb{R}^d$ be finite and $\ell \leqslant |Y|$ such that Y is (ℓ, opt_k)-coverable. Furthermore, let $Z \subset \mathbb{R}^d$ with $|Z| = \ell$ such that for all $n \in \mathbb{N}$ with $\ell + 1 \leqslant n \leqslant |Y|$, every cluster in $\mathcal{P}_n$ is (Z, opt_k)-connected to another cluster in $\mathcal{P}_n$, where $\mathcal{P}_1, \dots, \mathcal{P}_{|Y|}$ denotes the hierarchical clustering computed by Algorithm 3.2 on input Y. Then,*

$$\mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_{\ell+1}) < 2(\log_2(\ell) + 2) \cdot \left(24d \cdot e^{24d} + 1\right) \cdot \mathrm{opt}_k\,.$$

Proof. Let $n := \min(|Y|, 2\ell)$. Then, using Proposition 3.11,

$$\mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_n) < 24d \cdot e^{24d} \cdot \mathrm{opt}_k\,.$$

The result follows by using this bound together with Lemma 3.16. ☐

3.3.4 Proof of Theorem 3.10

By Lemma 3.14, there exists a subset $Y \subseteq X$, a number $\ell \leqslant k$, and a hierarchical clustering $\mathcal{P}_1, \dots, \mathcal{P}_{|Y|}$ of Y with

$$\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_k) \leqslant \mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_\ell)\,.$$

Furthermore, there exists a set $Z \subset \mathbb{R}^d$ such that every cluster from $\mathcal{P}_{\ell+1}$ is (Z, opt_k)-connected to another cluster in $\mathcal{P}_{\ell+1}$. Thus, $\mathcal{P}_{\ell+1}$ contains two clusters A, B that intersect the same ball of radius opt_k. Hence,

$$\mathrm{cost}_{\mathrm{rad}}(\mathcal{C}_k) \leqslant \mathrm{rad}(A \cup B) \leqslant 2\,\mathrm{cost}_{\mathrm{rad}}(\mathcal{P}_{\ell+1}) + \mathrm{opt}_k\,.$$

The theorem follows using Lemma 3.17 and $\ell \leqslant k$. ☐

Algorithm 3.3: AgglomerativeCompleteLinkage(X)

```
Input: finite set of input points $X \subset \mathbb{R}^d$

$\mathcal{C}_{|X|} \longleftarrow \{\{x\} \mid x \in X\}$
for $i = |X| - 1, \dots, 1$ do
    find distinct clusters $A, B \in \mathcal{C}_{i+1}$ minimizing $\mathrm{diam}(A \cup B)$
    $\mathcal{C}_i \longleftarrow (\mathcal{C}_{i+1} \setminus \{A, B\}) \cup \{A \cup B\}$
end
return $\mathcal{C}_{|X|}, \dots, \mathcal{C}_1$
```

3.4 DIAMETER k-CLUSTERING

In this section, we analyze the agglomerative complete linkage clustering algorithm for Problem 2.7 stated as Algorithm 3.3. Again, the only difference to Algorithm 3.1 and Algorithm 3.2 is the minimization of the diameter in the minimization step.

Note that in this section *cost* always means diameter cost and opt_k refers to the cost of an optimal diameter k-clustering of $X \subset \mathbb{R}^d$ where $k \in \mathbb{N}$ with $k \leqslant |X|$. That is, opt_k denotes the cost of an optimal solution to Problem 2.7. Analogously to the (discrete) radius case, any cluster C is contained in a ball of radius $\mathrm{diam}(C)$ and thus, the set X is (k, opt_k)-coverable.

This is analogous to Observation 3.4 and Observation 3.9.

Observation 3.18. *The cost of all computed clusterings is equal to the diameter of the cluster created last. Furthermore, the diameter of the union of any two clusters is always an upper bound for the cost of the clustering to be computed next.*

The following theorem states our main result.

Theorem 3.19. *Let $X \subset \mathbb{R}^d$ be a finite set of points. Then, for arbitrary metrics that are based on a norm and for all $k \in \mathbb{N}$ with $k \leqslant |X|$, the partition $\mathcal{C}_k$ of X into k clusters as computed by Algorithm 3.3 satisfies*

$$\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_k) = O(\log k) \cdot \mathrm{opt}_k,$$

where opt_k denotes the cost of an optimal solution to Problem 2.7, and the constant hidden in the O-notation is doubly exponential in the dimension d.

As in the proof of Theorem 3.5 and Theorem 3.10, we first bound the cost of the intermediate $2k$-clustering. However, we have to apply a different analysis again. This time, the new analysis results in a bound that depends doubly exponential on the dimension.

3.4.1 ANALYSIS OF THE $2k$-CLUSTERING

Proposition 3.20. *Let $X \subset \mathbb{R}^d$ be finite. Then, for all $k \in \mathbb{N}$ with $2k \leqslant |X|$, the partition $\mathcal{C}_{2k}$ of X into $2k$ clusters as computed by Algorithm 3.3 satisfies*

$$\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{2k}) < 2^{3\sigma}(28d + 6) \cdot \mathrm{opt}_k,$$

where $\sigma = (42d)^d$ *and* opt_k *denotes the cost of an optimal solution to Problem 2.7.*

In our analysis of the k-center problem, we made use of the fact that merging two clusters lying inside a ball of some radius r results in a new cluster of radius at most r. This is no longer true for the diameter k-clustering problem. We are not able to derive a bound for the diameter of the new cluster that is significantly less than $2r$. The additional factor of 2 makes our analysis from Section 3.3.1 useless for the diameter case since it would result in an approximation factor that is linear in the size of X rather than only depending on k and d.

To prove Proposition 3.20, we split the merge steps of Algorithm 3.3 into two stages. The first stage consists of the merge steps down to a $\left(2^{2^{O(d \log d)}}k\right)$-clustering. The analysis of the first stage is based on the following notion of similarity. Two clusters are called similar if one cluster can be translated such that every point of the translated cluster is near a point of the second cluster. Then, by merging similar clusters, the diameter essentially increases by the length of the translation vector. During the first stage, we guarantee that there is a sufficiently large number of similar clusters left. The cost of the intermediate $\left(2^{2^{O(d \log d)}}k\right)$-clustering can be upper bounded by $O(d) \cdot \mathrm{opt}_k$.

The second stage consists of the steps reducing the number of remaining clusters from $2^{2^{O(d \log d)}}k$ to only $2k$. Since the number of remaining merge steps as well as the upper bound of the cost are independent of $\|X\|$, we are able to show an approximation factor that only depends on k and d. In this stage, we are no longer able to guarantee that a sufficiently large number of similar clusters exists. Therefore, we analyze the merge steps of the second stage using a weaker argument, very similar to the one used in the second step of the analysis in the discrete k-center case (cf. Section 3.2.2). As long as there are more than $2k$ clusters left, we are able to find sufficiently many pairs of clusters that intersect the same cluster of an optimal k-clustering. Therefore, we can bound the cost of merging such a pair by the sum of the diameters of the two clusters plus the diameter of the optimal cluster. We show that the cost of the intermediate $2k$-clustering is upper bounded by $2^{2^{O(d \log d)}} \cdot \mathrm{opt}_k$. Let us remark that we do not obtain our main result if we already use this argument for the first stage.

Both stages are again subdivided into phases, such that in each phase the number of remaining clusters is reduced by one fourth.

stage one

The following lemma will be used to bound the increase of the cost during a single phase of the first stage.

Lemma 3.21. *Let* $\lambda \in \mathbb{R}$ *with* $0 < \lambda < 1$ *and* $\rho = \left\lceil \left(\frac{3}{\lambda}\right)^d \right\rceil$. *Furthermore, let* $m \in \mathbb{N}$ *with* $2^{\rho+1}k < m \leqslant |X|$. *Then,*

$$\mathrm{cost}_{\mathrm{diam}}\left(\mathcal{C}_{\lfloor \frac{3m}{4} \rfloor}\right) \leqslant (1+2\lambda)\,\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_m) + 4\sqrt[d]{\frac{2^{\rho+1}k}{m}}\,\mathrm{opt}_k\,. \tag{10}$$

Proof. Let $\tau := \mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_m)$. From every cluster $C \in \mathcal{C}_m$, we fix an arbitrary point and denote it by p_C. Then, the distance from p_C to any $q \in C$ is at most τ and

$$C - p_C := \{p - p_C \mid p \in C\} \subset B^d_\tau(0).$$

By Lemma 3.3, a ball of radius τ can be covered by ρ balls of radius $\lambda\tau$. Hence, we are able to fix $y_1, \dots, y_\rho \in \mathbb{R}^d$ with

$$B^d_\tau(0) \subseteq \bigcup_{i=1}^{\rho} B^d_{\lambda\tau}(y_i).$$

For $C \in \mathcal{C}_m$, we define the *configuration* of C by

$$\mathrm{Conf}(C) := \left\{ y_i \mid 1 \leqslant i \leqslant \rho \text{ and } B^d_{\lambda\tau}(y_i) \cap (C - p_C) \neq \varnothing \right\}.$$

That is, we identify each cluster $C \in \mathcal{C}_m$ with the subset of the balls $B^d_{\lambda\tau}(y_1), \dots, B^d_{\lambda\tau}(y_\rho)$ that intersect $C - p_C$. Note that no cluster from $C \in \mathcal{C}_m$ has an empty configuration and the number of possible configurations is upper bounded by 2^ρ.

As in the proof of Lemma 3.7 we define $t := \left\lfloor \frac{3m}{4} \right\rfloor$ and deduce $|\mathcal{C}_m \cap \mathcal{C}_{t+1}| > \frac{m}{2}$. It follows that there exist

$$j > \frac{1}{2^\rho} \cdot \frac{m}{2} = \frac{m}{2^{\rho+1}}$$

distinct clusters $C_1, \dots, C_j \in \mathcal{C}_m \cap \mathcal{C}_{t+1}$ with the same configuration. Using $m > 2^{\rho+1}k$, we deduce $j > k$.

Let $P := \{p_{C_1}, \dots, p_{C_j}\}$. Since X is (k, opt_k)-coverable, so is $P \subset X$. Therefore, by Lemma 3.2, there exist distinct $a, b \in \{1, \dots, j\}$ such that

$$\|p_{C_a} - p_{C_b}\| \leqslant 4\sqrt[d]{\frac{2^{\rho+1}k}{m}}\,\mathrm{opt}_k\,.$$

Next, we derive a bound for the diameter of the union of the corresponding clusters C_a and C_b. That is, we need to bound the largest distance $\|u - v\|$ between two points $u, v \in C_a \cup C_b$. If $u, v \in C_a$ or $u, v \in C_b$, then $\|u - v\| \leqslant \mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_m)$. Hence, let $u \in C_a$ and $v \in C_b$. Using the triangle inequality, for any $w \in \mathbb{R}^d$,

cf. Figure 7

$$\|u - v\| \leqslant \|p_{C_a} - p_{C_b}\| + \|u + p_{C_b} - p_{C_a} - w\| + \|w - v\|.$$

For $\|p_{C_a} - p_{C_b}\|$, we already derived an upper bound. To bound $\|u + p_{C_b} - p_{C_a} - w\|$, let $y \in \mathrm{Conf}(C_a) = \mathrm{Conf}(C_b)$ such that

$$u - p_{C_a} \in B^d_{\lambda\tau}(y).$$

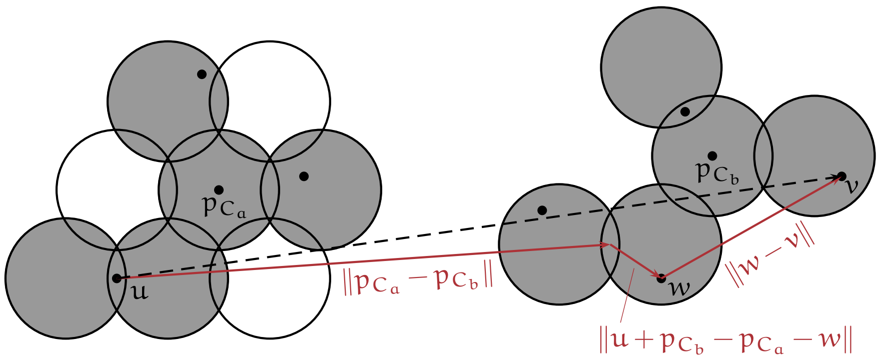


Figure 7: Congruent configurations.

Furthermore, let $w \in C_b$ with

$$w - p_{C_b} \in B^d_{\lambda\tau}(y).$$

Then,

$$\begin{aligned}\|u + p_{C_b} - p_{C_a} - w\| &= \|u - p_{C_a} - (w - p_{C_b})\| \\ &\leqslant 2\lambda \cdot \mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_m) \\ &= 2\lambda\tau.\end{aligned}$$

Since $v, w \in C_b$, their distance is bounded by

$$\|w - v\| \leqslant \mathrm{diam}(C_b) \leqslant \mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_m)\,.$$

We conclude that merging the clusters C_a and C_b results in a cluster whose diameter can be upper bounded by

$$\mathrm{diam}(C_a \cup C_b) \leqslant (1 + 2\lambda)\,\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_m) + 4\sqrt[d]{\frac{2^{\rho+1}k}{m}}\,\mathrm{opt}_k\,.$$

Using Observation 3.18 and the fact that C_a and C_b are part of the clustering $\mathcal{C}_{t+1}$, we can upper bound the cost of $\mathcal{C}_t$ by

$$\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_t) \leqslant \mathrm{diam}\,(C_a \cup C_b)\,.$$

$\square$

Note that the parameter λ from Lemma 3.21 establishes a trade-off between the two terms on the right-hand side of Inequality (10). For the analysis of the first stage, we have to choose λ carefully. In the proof of the following lemma, we use $\lambda = \ln\frac{4}{3}/4d$ and apply Lemma 3.21 for $\left\lceil \log_{\frac{4}{3}} \frac{|X|}{2^{\sigma+1}k} \right\rceil$ consecutive phases, where $\sigma = (42d)^d$. Then, we are able to upper bound the cost of the clustering computed after the first stage by a term that is linear in d and opt_k and independent of $|X|$ and k. The number of remaining clusters is independent of the number of input points $|X|$ and depends only on the dimension d and the desired number of clusters k.

Lemma 3.22. *Let* $2^{\sigma+1}k < |X|$ *for* $\sigma = (42d)^d$. *Then, on input* X, *Algorithm 3.3 computes a clustering* $\mathcal{C}_{2^{\sigma+1}k}$ *with*

$$\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{2^{\sigma+1}k}) < (28d+4)\,\mathrm{opt}_k\,.$$

Proof. Let $u := \left\lceil \log_{\frac{3}{4}} \frac{2^{\sigma+1}k}{|X|} \right\rceil$ and define $m_i := \left\lceil \left(\frac{3}{4}\right)^i |X| \right\rceil$ for $i = 0, \dots, u$. Furthermore, let $\lambda := \ln \frac{4}{3}/4d$. This implies $\rho \leqslant \sigma$ for the parameter ρ of Lemma 3.21. Then, $m_u \leqslant 2^{\sigma+1}k$ and $m_i > 2^{\sigma+1}k \geqslant 2^{\rho+1}k$ for $i = 0, \dots, u-1$. Since

$$\left\lfloor \frac{3m_i}{4} \right\rfloor = \left\lfloor \frac{3}{4} \left\lceil \left(\frac{3}{4}\right)^i |X| \right\rceil \right\rfloor \leqslant \left\lfloor \left(\frac{3}{4}\right)^{i+1} |X| + \frac{3}{4} \right\rfloor \leqslant \left\lceil \left(\frac{3}{4}\right)^{i+1} |X| \right\rceil = m_{i+1}$$

and Algorithm 3.3 uses a greedy strategy,

$$\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{m_{i+1}}) \leqslant \mathrm{cost}_{\mathrm{diam}}\left(\mathcal{C}_{\left\lfloor \frac{3m_i}{4} \right\rfloor}\right)$$

for $i = 0, \dots, u-1$. Combining this inequality with Lemma 3.21 (applied to $m = m_i$), we obtain

$$\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{m_{i+1}}) \leqslant (1+2\lambda)\,\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{m_i}) + 4\sqrt[d]{\frac{2^{\rho+1}k}{m_i}}\,\mathrm{opt}_k\,.$$

By repeatedly applying this inequality for $i = 0, \dots, u-1$ and using $\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{2^{\sigma+1}k}) \leqslant \mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{m_u})$ and $\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{m_0}) = 0$, we get

$$\begin{aligned}
\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{2^{\sigma+1}k}) &\leqslant \sum_{i=0}^{u-1} \left((1+2\lambda)^i \cdot 4\sqrt[d]{\frac{2^{\sigma+1}k}{\left(\frac{3}{4}\right)^{u-1-i}|X|}}\,\mathrm{opt}_k \right) \\
&= 4\sqrt[d]{\frac{2^{\sigma+1}k}{\left(\frac{3}{4}\right)^{u-1}|X|}}\,\mathrm{opt}_k \cdot \sum_{i=0}^{u-1} \left((1+2\lambda)^i \cdot \sqrt[d]{\left(\frac{3}{4}\right)^i} \right).
\end{aligned}$$

Using $u - 1 < \log_{\frac{3}{4}} \frac{2^{\sigma+1}k}{|X|}$,

$$\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{2^{\sigma+1}k}) < 4\,\mathrm{opt}_k \cdot \sum_{i=0}^{u-1} \left(\frac{1+2\lambda}{\sqrt[d]{\frac{4}{3}}} \right)^i . \tag{11}$$

By taking only the first two terms of the series expansion of the exponential function, we get $1 + 2\lambda = 1 + \frac{\ln \frac{4}{3}}{2d} < e^{\frac{\ln \frac{4}{3}}{2d}} = \sqrt[2d]{\frac{4}{3}}$. Substituting this bound into Inequality (11) and extending the sum yields

$$\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{2^{\sigma+1}k}) < 4\,\mathrm{opt}_k \cdot \sum_{i=0}^{\infty} \left(\frac{1}{\sqrt[2d]{\frac{4}{3}}} \right)^i < 4\,\mathrm{opt}_k \cdot \sum_{i=0}^{\infty} \left(\frac{1}{1+2\lambda} \right)^i .$$

Solving the geometric series leads to

$$\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{2^{\sigma+1}k}) < 4\left(\frac{1}{2\lambda} + 1\right) \cdot \mathrm{opt}_k < (28d+4) \cdot \mathrm{opt}_k\,.$$

□

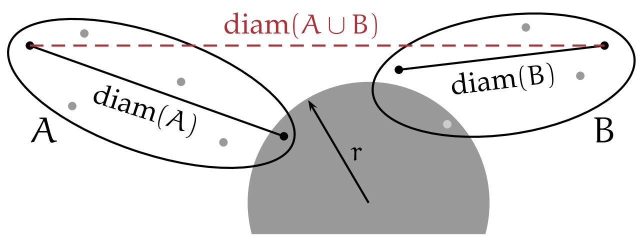


Figure 8: Merging two clusters intersecting a ball of radius r.

STAGE TWO

The second stage covers the remaining merge steps until Algorithm 3.3 computes the clustering $\mathcal{C}_{2k}$. However, compared to stage one, the analysis of a single phase yields a weaker bound. The following lemma provides an analysis of a single phase of the second stage. It is very similar to Lemma 3.7 and Lemma 3.8 in the analysis of the discrete k-center problem.

Lemma 3.23. *Let $m \in \mathbb{N}$ with $2k < m \leqslant |X|$. Then,*

$$\mathrm{cost}_{\mathrm{diam}}\left(\mathcal{C}_{\lfloor \frac{3m}{4} \rfloor}\right) \leqslant 2 \cdot \left(\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_m) + \mathrm{opt}_k\right).$$

Proof. As in the proof of Lemma 3.7, we define $t := \lfloor \frac{3m}{4} \rfloor$ and deduce $|\mathcal{C}_m \cap \mathcal{C}_{t+1}| > \frac{m}{2} > k$. Since X is (k, opt_k)-coverable, there exists a point $y \in \mathbb{R}^d$ such that $B^d_{\mathrm{opt}_k}(y)$ intersects two clusters $A, B \in \mathcal{C}_m \cap \mathcal{C}_{t+1}$. We conclude that merging A and B would result in a cluster whose diameter can be upper bounded by

$$\mathrm{diam}(A \cup B) \leqslant 2\,\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_m) + 2\,\mathrm{opt}_k\,.$$

cf. Figure 8

The result follows using $A, B \in \mathcal{C}_{t+1}$ and Observation 3.18. □

Lemma 3.24. *Let $n \in \mathbb{N}$ with $n \leqslant 2^{\sigma+1}k$ and $2k < n \leqslant |X|$ for $\sigma = (42d)^d$. Then, on input X, Algorithm 3.3 computes a clustering $\mathcal{C}_{2k}$ with*

$$\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{2k}) < 2^{3\sigma}\left(\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_n) + 2\,\mathrm{opt}_k\right).$$

Proof. Let $u := \left\lceil \log_{\frac{3}{4}} \frac{2k}{n} \right\rceil$ and define $m_i := \left\lceil \left(\frac{3}{4}\right)^i n \right\rceil$ for $i = 0, \dots, u$. Then, $m_u \leqslant 2k$ and $m_i > 2k$ for $i = 0, \dots, u-1$. Analogously to the proof of Lemma 3.22, we get $\left\lfloor \frac{3m_i}{4} \right\rfloor \leqslant m_{i+1}$ and using Lemma 3.23,

$$\mathrm{cost}_{\mathrm{diam}}\left(\mathcal{C}_{m_{i+1}}\right) \leqslant 2\left(\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{m_i}) + \mathrm{opt}_k\right)$$

for $i = 0, \dots, u-1$. By repeatedly applying this inequality for $i = 0, \dots, u-1$ and using $\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{2k}) \leqslant \mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{m_u})$, we get

$$\begin{aligned}\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_{2k}) &\leqslant 2^u\,\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_n) + \sum_{i=1}^{u} 2^i\,\mathrm{opt}_k \\ &< 2^u\left(\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_n) + 2\,\mathrm{opt}_k\right).\end{aligned}$$

Hence using $u \leqslant \left\lceil \log_{\frac{4}{3}} 2^\sigma \right\rceil < 3\sigma$, the result follows. □

Proof of Proposition 3.20. The Proposition follows immediately by combining Lemma 3.22 and Lemma 3.24. □

3.4.2 ANALYSIS OF THE REMAINING MERGE STEPS

We analyze the remaining merge steps analogously to the k-center problem. Therefore, in this section we only discuss the differences, most of which are slightly modified bounds for the cost of merging two clusters.

The connectivity property from Section 3.3.2 remains the same. However, for any two (Z, r)-connected clusters A, B, we use

cf. Figure 8

$$\operatorname{diam}(A \cup B) \leqslant \operatorname{diam}(A) + \operatorname{diam}(B) + 2r \tag{12}$$

as a replacement for Inequality (6). Furthermore, Lemma 3.14 also holds for the diameter k-clustering problem, i. e., with

$$\text{cost}_{\text{diam}}(\mathcal{C}_k) \leqslant \text{cost}_{\text{diam}}(\mathcal{P}_\ell)\,.$$

Using Inequality (12) in the proof of Lemma 3.15, we get

$$\operatorname{diam}(A \cup B) \leqslant \text{cost}_{\text{diam}}(\mathcal{P}_m) + \text{cost}_{\text{diam}}(\mathcal{P}_n) + 2\,\text{opt}_k$$

as a replacement for Inequality (7) while Inequality (8) can be replaced by

$$\operatorname{diam}(A_1 \cup A_2) \leqslant \text{cost}_{\text{diam}}(\mathcal{P}_m) + 2(\text{cost}_{\text{diam}}(\mathcal{P}_n) + 2\,\text{opt}_k).$$

That is, for the diameter k-clustering problem the two upper bounds are different. However, the second one is larger than the first one. Using it in both cases, the inequality stated in Lemma 3.15 changes slightly to

$$\text{cost}_{\text{diam}}\left(\mathcal{P}_{\lfloor \frac{m+\ell}{2} \rfloor}\right) \leqslant \text{cost}_{\text{diam}}(\mathcal{P}_m) + 2\left(\text{cost}_{\text{diam}}(\mathcal{P}_n) + 2\,\text{opt}_k\right)\,.$$

Together with $\text{cost}_{\text{diam}}(\mathcal{P}_{\tilde{n}}) \leqslant 2\,\text{cost}_{\text{diam}}(\mathcal{P}_n) + 2\,\text{opt}_k$ as a replacement for Inequality (9), the bound stated in Lemma 3.16 becomes

$$\text{cost}_{\text{diam}}(\mathcal{P}_{\ell+1}) < 2\left(\log_2(\ell) + 2\right)\left(\text{cost}_{\text{diam}}(\mathcal{P}_n) + 2\,\text{opt}_k\right)\,.$$

Thus, using Proposition 3.20, the upper bound for the cost of the $(\ell+1)$-clustering of Y stated in Lemma 3.17 becomes

$$\text{cost}_{\text{diam}}(\mathcal{P}_{\ell+1}) < 2\left(\log_2(\ell) + 2\right)\left(2^{3\sigma}(28d + 6) + 2\right) \cdot \text{opt}_k$$

for $\sigma = (42d)^d$. Analogously to Section 3.3.4, this proves Theorem 3.19.

3.5 THE ONE-DIMENSIONAL CASE

For $d = 1$, we are able to show that Algorithm 3.3 computes an approximation to the diameter k-clustering problem (Problem 2.7) with an approximation factor of at most 3. We even know that for any data set $X \subset \mathbb{R}$ the approximation factor of the computed solution is strictly below 3. However, we do not show an approximation factor of $3 - \epsilon$ for some $\epsilon > 0$. The proof of this upper bound makes use of the total order of the real numbers and is certainly not generalizable to higher dimensions.

Theorem 3.25. *Let $d = 1$. Then, Algorithm 3.3 gives a solution to the diameter k-clustering problem (Problem 2.7) with cost less than three times the cost of an optimal solution.*

For any set $C \subset \mathbb{R}$ it holds that $\mathrm{diam}(C) = 2\,\mathrm{rad}(C)$. That is, in the one-dimensional case, the diameter and the radius of a cluster differ by a constant factor. Thus, for any two clusters A and B,

$$\mathrm{diam}(A) < \mathrm{diam}(B) \;\Leftrightarrow\; \mathrm{rad}(A) < \mathrm{rad}(B).$$

It follows that Algorithm 3.3 and Algorithm 3.2 choose the same clusters in the merge step and we immediately get the following corollary.

We assume that there are no ties.

Corollary 3.26. *Let $d = 1$. Then, Algorithm 3.2 gives a solution to the k-center problem (Problem 2.8) with cost less than three times the cost of an optimal solution.*

Before proving Theorem 3.25, we discuss some features of one-dimensional clusterings and introduce a suitable terminology. Afterwards, we formulate the central argument used in the proof of Theorem 3.25 as a separate lemma.

In the following, we consider solely clusterings of the finite input data set $X \subset \mathbb{R}$. Since Algorithm 3.3 uses a greedy strategy to choose the clusters to be merged, it merges only clusters whose points lie directly next to each other. That is, the created clusters are cohesive in the sense that they contain all points of X that lie between their smallest and their largest point. Without loss of generality, we may assume that the same holds for the clusters of an optimal solution. Obviously, any clustering can be transformed to have this property without changing the largest diameter. It follows that all considered clusters are uniquely determined by their smallest and their largest point. Therefore, in the figures of this section, clusters are depicted by line segments between their smallest and their largest point. We call the smallest point of a cluster its *left boundary* and the largest point its *right boundary*.

We start with considering the relative position of two clusters $A, B \subset X$ from possibly different clusterings of $X \subset \mathbb{R}$. We say that A *lies inside* of B, if $A \subset B$. Furthermore, A *lies to the left* of B if the right boundary of A is strictly less than the left boundary of B, i. e., $\max_{x \in A} x <$

$\min_{x \in B} x$. Analogously, A *lies to the right* of B, if $\max_{x \in B} x < \min_{x \in A} x$. For two distinct clusters A, B from the same clustering $\mathcal{C}$, it follows that A lies either to the left or to the right of B. We say that such clusters A and B are neighboring if no further cluster $C \in \mathcal{C}$ lies in between. A set of $n > 2$ clusters from the same clustering $\mathcal{C}$ is called neighboring if the clusters can be denoted as $C_1, \ldots, C_n$ such that for $i = 1, \ldots, n-1$, the clusters C_i and C_{i+1} are neighboring. For two neighboring clusters A, B with A lying to the left of B, we define the *gap* between A and B as the open interval between the right boundary of A and the left boundary of B. It follows that a gap contains no elements from X. Furthermore, for two clusters A, B from different clusterings we define A to be *left aligned to* B, if A and B share their left boundary, i.e., $\min_{x \in A} x = \min_{x \in B}$. Analogously, A is right aligned to B, if $\max_{x \in A} x = \max_{x \in B} x$. If it is clear from the context or if it is irrelevant which cluster a cluster A is aligned to, we simply say A is left (or right) aligned.

For the remainder of this section, we fix an optimal diameter k-clustering $\mathcal{S}$ of X. That is, $\mathcal{S}$ is an optimal solution to Problem 2.7 for $d = 1$. Without loss of generality, we assume $\mathcal{S} = \{S_1, \ldots, S_k\}$ with S_i lying to the left of S_{i+1} for $i = 1, \ldots, k-1$. Furthermore, we denote the hierarchical clusterings computed by Algorithm 3.3 on input X by $\mathcal{C}_{|X|}, \ldots, \mathcal{C}_1$. For our line of reasoning, we focus on two particular clusterings $\mathcal{C}_\ell$ and $\mathcal{C}_m$ with $\ell \leqslant m$. Let $m \in \mathbb{N}$ be the smallest index, such that all clusters of $\mathcal{C}_m$ have a diameter of at most opt_k and let $\ell \in \mathbb{N}$ be the smallest index, such that all clusters of $\mathcal{C}_\ell$ have a diameter strictly less than $3 \cdot \mathrm{opt}_k$. Since Algorithm 3.3 uses a greedy strategy, it is sufficient to show that ℓ is at most k to prove Theorem 3.25. To do this, we use the following technical lemma.

The smaller the index, the later the clustering is computed.

Lemma 3.27. *Let $\mathcal{P} \subset \mathcal{S}$ be a set of neighboring clusters from the optimal solution $\mathcal{S}$ with the following properties:*

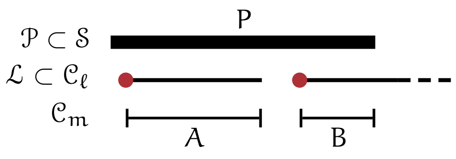
1. *Gaps between neighboring clusters from $\mathcal{P}$ have length at most opt_k.*
2. *The rightmost cluster of $\mathcal{P}$ is right aligned to a cluster $C \in \mathcal{C}_m$ and it is the only one with this property.*

Furthermore, let $\mathcal{L} \subset \mathcal{C}_\ell$ be such that:

3. *For each cluster $C \in \mathcal{L}$, the left boundary of C is contained in one of the clusters from $\mathcal{P}$.*
4. *The left boundary of the leftmost cluster of $\mathcal{L}$ is contained in the leftmost cluster of $\mathcal{P}$.*

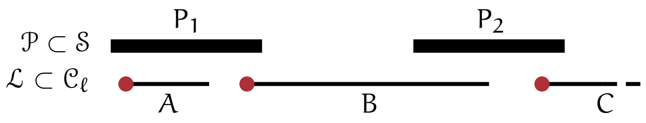
Then, it holds that $|\mathcal{L}| \leqslant |\mathcal{P}|$.

Proof. We prove the lemma by induction over the size of $\mathcal{P}$ with two base cases. First, let $|\mathcal{P}| = 1$. We denote $\mathcal{P} = \{P\}$ and assume $|\mathcal{L}| \geqslant 2$ for contradiction. By definition, each cluster from $\mathcal{L}$ is the union of one or



$\mathcal{P} \subset \mathcal{S}$
P
$\mathcal{L} \subset \mathcal{C}_\ell$
$\mathcal{C}_m$
A
B

(a) Sketch of contradiction for $|\mathcal{P}| = 1$.



$\mathcal{P} \subset \mathcal{S}$
P_1
P_2
$\mathcal{L} \subset \mathcal{C}_\ell$
A
B
C

(b) Sketch of contradiction for $|\mathcal{P}| = 2$.

Figure 9: The two base cases of the induction.

more clusters from $\mathcal{C}_m$ as illustrated in Figure 9a. It follows that there exist at least two clusters $A, B \in \mathcal{C}_m$ such that their left boundaries are contained in the single cluster P. Then, the right boundaries of A and B also have to be contained in P. Otherwise, there could not exist a cluster $C \in \mathcal{C}_m$ that is right aligned to P as required for the second property. It follows that A and B both lie inside of P which has a diameter of at most opt_k. That is, the union of A and B would have a diameter of at most opt_k, too. Thus, Algorithm 3.3 would have created a clustering $\mathcal{C}_{m-1}$ with cost of at most opt_k. However, this contradicts the definition of m.

Second, let $|\mathcal{P}| = 2$, denote $\mathcal{P} = \{P_1, P_2\}$ with P_1 lying to the left of P_2 and assume $|\mathcal{L}| \geqslant 3$ for contradiction. Let $C \in \mathcal{L}$ be the rightmost cluster of $\mathcal{L}$ and recall that the gap between the two neighboring clusters P_1 and P_2 contains no points from X. It follows that there have to exist at least two clusters $A, B \in \mathcal{L}$ to the left of C, which both lie inside $P_1 \cup P_2$ (cf. Figure 9b). Furthermore, the right boundaries of A and B have to be strictly less than the right boundary of P_2. Otherwise, the left boundary of C could not be contained in one of the clusters of $\mathcal{P}$ as required in the third property. Using the first property, it follows that the union of A and B has a diameter of strictly less than $3 \cdot \mathrm{opt}_k$. However, analogously to the first base case, this contradicts the definition of ℓ.

Finally, for the inductive step, let $|\mathcal{P}| \geqslant 3$ and assume that for any two sets $\mathcal{P}'$ and $\mathcal{L}'$ as defined in the lemma but with $|\mathcal{P}'| < |\mathcal{P}|$, it holds that $|\mathcal{L}'| \leqslant |\mathcal{P}'|$. Let $P_1, P_2 \in \mathcal{P}$ be the two leftmost clusters of $\mathcal{P}$ with P_1 lying to the left of P_2. Furthermore, let $A, B, C \in \mathcal{L}$ be the three leftmost clusters of $\mathcal{L} \subset \mathcal{C}_\ell$ ordered from left to right as illustrated in Figure 10. Every cluster in $\mathcal{C}_\ell$ is the union of one or more clusters from $\mathcal{C}_m$. Thus, the second property yields that the rightmost cluster of $\mathcal{P}$ is the only cluster from $\mathcal{P}$ that may be right aligned to a cluster from $\mathcal{C}_\ell$. However, this cluster from $\mathcal{C}_\ell$ can not be the cluster B, since to the right of B there has to be room for the cluster C. It follows that B is not right aligned to a cluster from $\mathcal{P}$.

For $|\mathcal{L}| < 3$, there is nothing to show.

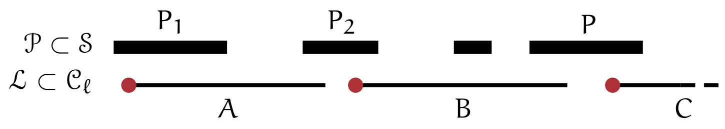

P_1
P_2
P
$\mathcal{P} \subset \mathcal{S}$
$\mathcal{L} \subset \mathcal{C}_\ell$
A
B
C

Figure 10: The inductive step.

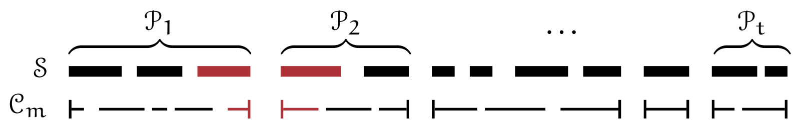


Figure 11: Partition of the optimal solution.

The definition of ℓ yields $\operatorname{diam}(A \cup B) \geqslant 3 \cdot \operatorname{opt}_k$ and the first property provides $\operatorname{diam}(P_1 \cup P_2) \leqslant 3 \cdot \operatorname{opt}_k$. Since B can not be right aligned to P_2, the right boundary of B and the left boundary of C both have to be contained in a cluster $P \in \mathcal{P}$ that lies to the right of P_2.

We define a new set $\mathcal{P}'$ by removing all clusters from $\mathcal{P}$ that lie to the left of P. It follows that $|\mathcal{P}'| \leqslant |\mathcal{P}| - 2$. Furthermore, we define $\mathcal{L}' := \mathcal{L} \setminus \{A, B\}$. Then, all four properties of $\mathcal{P}$ and $\mathcal{L}$ as required by the lemma are also satisfied by $\mathcal{P}'$ and $\mathcal{L}'$. Using the inductive hypothesis $|\mathcal{L}'| \leqslant |\mathcal{P}'|$, we conclude

$$|\mathcal{L}| = |\mathcal{L}'| + 2 \leqslant |\mathcal{P}'| + 2 \leqslant |\mathcal{P}|.$$

□

Proof of Theorem 3.25. In the following, we denote $\mathcal{C}_m = \{C_1, \dots, C_m\}$ with C_i lying to the left of C_{i+1} for $i = 1, \dots, m-1$. Since $\mathcal{S}$ and $\mathcal{C}_m$ are clusterings of the same set X, it follows that C_1 is left aligned to S_1 and C_m is right aligned to S_k. Furthermore, it follows that if for any $1 \leqslant i \leqslant m-1$ and $1 \leqslant j \leqslant k-1$ the cluster C_i is right aligned to the cluster S_j, then C_{i+1} has to be left aligned to S_{j+1} (cf. Figure 11). That is, $\mathcal{C}_m$ and $\mathcal{S}$ have a common gap between the neighboring clusters C_i, C_{i+1} and S_j, S_{j+1}, respectively. We use these common gaps of $\mathcal{C}_m$ and $\mathcal{S}$ to define a partition $\mathcal{P}_1, \dots, \mathcal{P}_t$ of the optimal solution $\mathcal{S}$. Let $t \in \mathbb{N}$ be the number of clusters in $\mathcal{S}$ that are left aligned to a cluster of $\mathcal{C}_m$ (which is also the number of right aligned clusters). Then, for $i = 1, \dots, t$ let L_i be the i-th cluster of $\mathcal{S}$ (counted from the left) that is left aligned to a cluster of $\mathcal{C}_m$ and let R_i be the i-th right aligned cluster. Using L_i and R_i, we define $\mathcal{P}_i$ to be the set of neighboring clusters of $\mathcal{S}$ from L_i to R_i.

Based on the partition $\mathcal{P}_1, \dots, \mathcal{P}_t$ of $\mathcal{S}$ we define a partition of $\mathcal{C}_\ell$. To this, for each $i \in \{1, \dots, t\}$, we pick all clusters from $\mathcal{C}_\ell$ whose left boundary is contained in a cluster from $\mathcal{P}_i$. This results in a partition $\mathcal{L}_1, \dots, \mathcal{L}_t$ of $\mathcal{C}_\ell$ with

$$\mathcal{L}_i := \left\{ C \in \mathcal{C}_\ell \,\middle|\, \left(\min_{x \in C} x\right) \in P \text{ for a cluster } P \in \mathcal{P}_i \right\}.$$

Then, for each $i = 1, \dots, t$, the sets $\mathcal{P}_i$ and $\mathcal{L}_i$ satisfy the first three properties required in Lemma 3.27 for the sets $\mathcal{P}$ and $\mathcal{L}$. In order to also satisfy the fourth property, we need to modify the sets $\mathcal{P}_1, \dots, \mathcal{P}_t$ slightly. To this, for each $i = 1, \dots, t$, we define $\mathcal{P}_i^*$ by removing the leftmost clusters from $\mathcal{P}_i$ that do not contain the left boundary of a cluster $C \in \mathcal{L}_i$. Then, we are able to apply Lemma 3.27 with $\mathcal{P} = \mathcal{P}_i^*$ and $\mathcal{L} = \mathcal{L}_i$ for $i = 1, \dots, t$. Using $|\mathcal{P}_i^*| \leqslant |\mathcal{P}_i|$, it follows that $|\mathcal{L}_i| \leqslant |\mathcal{P}_i|$. Since, $\mathcal{P}_1, \dots, \mathcal{P}_t$ and $\mathcal{L}_1, \dots, \mathcal{L}_t$ are partitions of $\mathcal{S}$ and $\mathcal{C}_\ell$, respectively,

$$\ell = \sum_{i=1}^{t} |\mathcal{L}_i| \leqslant \sum_{i=1}^{t} |\mathcal{P}_i| = k.$$

As mentioned above, this proves the theorem. □

3.6 LOWER BOUNDS

In this section, we present constructions of several data sets for the diameter k-clustering problem. The data sets yield lower bounds for the approximation factor of Algorithm 3.3. Some of the data sets are also suitable to prove lower bounds for Algorithm 3.2 and Algorithm 3.1 for the k-center problem and the discrete k-center problem, respectively. To keep the constructions simple, we make use of the fact that we presented the clustering algorithms without a particular tie-breaking strategy. That is, whenever one of the algorithms is able to choose between several possible merge steps, we simply assume that we can govern its choice.

In Section 3.6.1, we show that for data sets $X \subset \mathbb{R}$ (i. e., $d = 1$), Algorithm 3.3 has an approximation factor of at least 2.5. Recall that in in Section 3.5 we stated that in the one-dimensional case Algorithm 3.3 computes a solution to the diameter k-clustering problem with an approximation factor strictly below 3. Hence, for $d = 1$, we obtain almost matching upper and lower bounds for the cost of the solution computed by Algorithm 3.3. The same holds for Algorithm 3.2 and the k-center problem since for $d = 1$ the two problems as well as the two algorithms are equivalent.

In Section 3.6.2, we show that the dimension d has an impact on the approximation factor of Algorithm 3.3. This follows from a two-dimensional data set yielding a lower bound of 3 for the metric based on the L_∞-norm. Note that this exceeds the upper bound from the one-dimensional case.

In Section 3.6.4, we show that there exist data sets such that Algorithm 3.3 computes an approximation to the diameter k-clustering problem (Problem 2.7) with an approximation factor of $\Omega(\sqrt[p]{\log k})$ for metrics based on an L_p-norm $(1 \leqslant p < \infty)$ and $\Omega(\log k)$ for the metric based on the L_∞-norm. In case of the L_1- and the L_∞-norm, this matches an already known lower bound that has been shown using a rather artificial metric [Dasgupta and Long, 2005]. However, the bound in [Dasgupta and Long, 2005] is derived from a two-dimensional data set, while the dimension of our construction depends on the number of clusters k.

Finally, in Section 3.6.4, we show that the lower bound of $\Omega(\sqrt[p]{\log k})$ for any L_p-norm and $\Omega(\log k)$ for the L_∞-norm can be adapted to the discrete k-center problem. In case of the L_2-norm, we thus obtain almost matching upper and lower bounds for the cost of the solution computed by Algorithm 3.1. Furthermore, we are able to restrict the dependency of the approximation factor of Algorithm 3.1 on d and k.

3.6.1 ARBITRARY METRICS AND $d = 1$

We first show a lower bound for the approximation factor of Algorithm 3.3. To do this, we use a sequence of data sets from $\mathbb{R}^d$ with $d = 1$. Since up to normalization there is only one norm for $d = 1$, without loss of generality we assume the Euclidean metric.

Proposition 3.28. *For all $\varepsilon > 0$ and $k \geqslant 4$ there exists a data set $X \subset \mathbb{R}$ such that Algorithm 3.3 computes a solution to the diameter k-clustering problem (Problem 2.7) with cost at least $\frac{5}{2} - \varepsilon$ times the cost of an optimal solution.*

Proof. It is sufficient to consider the case $k = 4$. The construction can easily be extended to $k > 4$ by adding separated dummy points. In the following we construct a data set for any fixed $n \in \mathbb{N}$. That is, we actually obtain a series of data sets. For any $n \in \mathbb{N}$ we show that the computed solution has an approximation factor of $\frac{5}{2} - f(n)$ for $f(n) > 0$ and $\lim_{n\to\infty} f(n) = 0$.

For $x \in \mathbb{R}$, let

$$V(x) := \{x + i \mid i \in \mathbb{N} \text{ and } 0 \leqslant i \leqslant 2^n - 1\}.$$

That is, $V(x)$ consists of 2^n equidistant points, where the distance between neighboring points is 1 and $\operatorname{diam}(V(x)) = 2^n - 1$. Furthermore, we define

$$\begin{aligned} l(x) &:= x - 2^{n-1}, \\ r(x) &:= x + 2^n - 1 + 2^{n-1} = x + 3 \cdot 2^{n-1} - 1, \\ W(x) &:= V(x) \cup \{l(x), r(x)\}. \end{aligned}$$

It follows that $\operatorname{diam}(W(x)) = 2^{n+1} - 1$ as shown in Figure 12.

Based on the sets $W(x)$, we define

$$X := \bigcup_{i=1}^{4} W(x_i)$$

where $x_i := i \cdot (7 \cdot 2^{n-1} - 2)$ for $i = 1, \dots, 4$. Then, there is a gap of $3 \cdot 2^{n-1} - 1$ between $W_n(x_i)$ and $W_n(x_{i+1})$, i.e.

$$\operatorname{diam}(\{r(x_i), l(x_{i+1})\}) = 3 \cdot 2^{n-1} - 1 \quad \text{for } i = 1, \dots, 3. \tag{13}$$

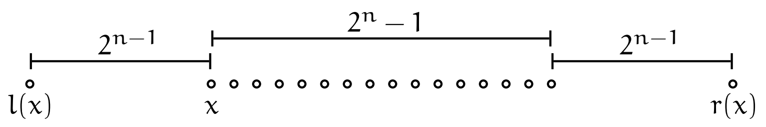


Figure 12: Sketch of the set $W(x)$.

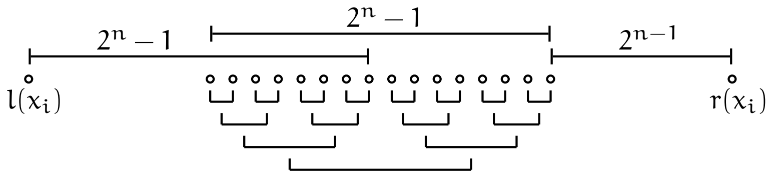


Figure 13: Part of the dendrogram for $W(x_i)$.

The optimal 4-clustering of X is

$$\mathcal{C}_4^{opt} = \{W(x_1), W(x_2), W(x_3), W(x_4)\}$$

and $\text{cost}_{\text{diam}}\left(\mathcal{C}_4^{opt}\right) = 2^{n+1} - 1$.

The solution computed by Algorithm 3.3 may have larger cost. At the beginning, the minimum distance between two points from X is 1. The possible pairs of points with distance 1 come from the sets $V(x_i)$ for $i = 1, \ldots, 4$. Since the distance between $V(x_i)$ and $l(x_i)$ or $r(x_i)$ is 2^{n-1}, we can assume that the algorithm merges all points of $V(x_i)$ for $i = 1, \ldots, 4$ as shown in Figure 13. It follows that Algorithm 3.3 computes the 12-clustering

$$\begin{aligned}\mathcal{C}_{12} = \big\{&\{l(x_1)\}, V(x_1), \{r(x_1)\},\\ &\{l(x_2)\}, V(x_2), \{r(x_2)\},\\ &\{l(x_3)\}, V(x_3), \{r(x_3)\},\\ &\{l(x_4)\}, V(x_4), \{r(x_4)\}\big\}.\end{aligned}$$

For $i = 1, \ldots, 4$, the diameters of $\{l(x_i)\} \cup V(x_i)$ and $V(x_i) \cup \{r(x_i)\}$ are equal to $3 \cdot 2^{n-1} - 1$ and these are the best possible merge steps. Therefore, by (13), we can assume that Algorithm 3.3 merges $r(x_i)$ and $l(x_{i+1})$ for $i = 1, \ldots, 3$ first. This results in the 7-clustering

$$\begin{aligned}\mathcal{C}_7 = \big\{&\{l(x_1)\} \cup V(x_1),\\ &\{r(x_1), l(x_2)\}, V(x_2),\\ &\{r(x_2), l(x_3)\}, V(x_3),\\ &\{r(x_3), l(x_4)\}, V(x_4) \cup \{r(x_4)\}\big\}\end{aligned}$$

where $V(x_2)$ and $V(x_3)$ have a diameter of $2^n - 1$ while the remaining clusters have a diameter of $3 \cdot 2^{n-1} - 1$ (see Figure 14). Between two neighboring clusters of $\mathcal{C}_7$, there is a gap of 2^{n-1}.

In the next step of Algorithm 3.3, the best possible choice is to merge $\{r(x_1), l(x_2)\}$ with $V(x_2)$, $\{r(x_2), l(x_3)\}$ with $V(x_2)$ or $V(x_3)$, or

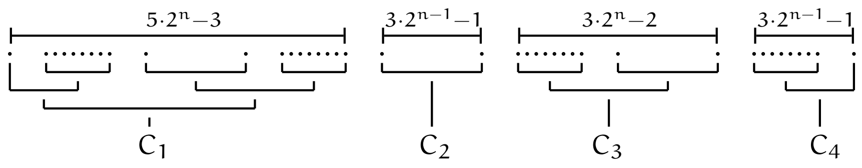


Figure 14: Part of the dendrogram for X.

$\{r(x_3), l(x_4)\}$ with $V(x_3)$. We let the algorithm merge $\{r(x_1), l(x_2)\}$ with $V(x_2)$ and $\{r(x_3), l(x_4)\}$ with $V(x_3)$. This results in a 5-clustering where the clusters have alternating lengths of $3 \cdot 2^{n-1} - 1$ and $3 \cdot 2^n - 2$ with gaps of 2^{n-1} between them. Then, in the step resulting in $\mathcal{C}_4$, Algorithm 3.3 has to create a cluster of diameter $5 \cdot 2^n - 3$ as shown in Figure 14. Therefore, the computed solution has an approximation factor of

$$\frac{\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_4)}{\mathrm{cost}_{\mathrm{diam}}\left(\mathcal{C}_4^{\mathrm{opt}}\right)} = \frac{5 \cdot 2^n - 3}{2^{n+1} - 1}.$$

For n going to infinity this approximation factor converges from below to $\frac{5}{2}$. $\square$

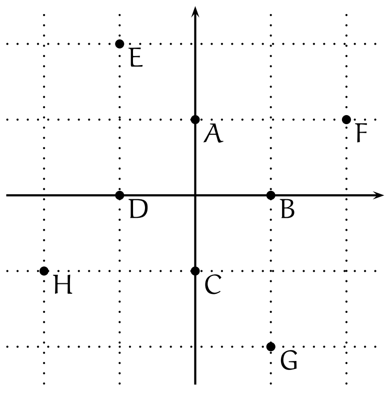
Analogously to Section 3.5, the bound for the one-dimensional diameter k-clustering problem immediately yields the same bound for the one-dimensional k-center problem.

Corollary 3.29. *For all $\varepsilon > 0$ and $k \geqslant 4$ there exists a data set $X \subset \mathbb{R}$ such that Algorithm 3.2 computes a solution to the k-center problem (Problem 2.8) with cost at least $\frac{5}{2} - \varepsilon$ times the cost of an optimal solution.*

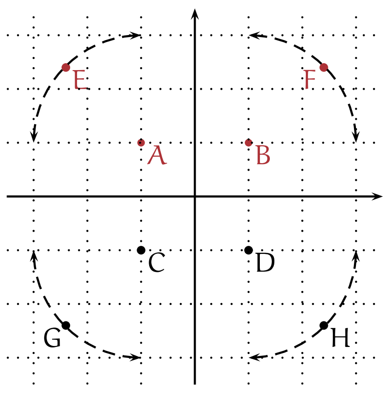
3.6.2 l_∞-METRIC AND $d = 2$

In this section, we construct a data set that uses only eight points from $\mathbb{R}^2$ and yields a lower bound of 3 for the metric based on the L_∞-norm. Recall that in Section 3.5, we showed that for $d = 1$ the approximation factor of a computed solution is always strictly less than 3. Therefore, the lower bound of 3 for $d = 2$ implies that the dimension d has an impact on the approximation factor of Algorithm 3.3.

Proposition 3.30. *For the metric based on the L_∞-norm, there exists a data set $X \subset \mathbb{R}^2$ such that Algorithm 3.3 computes a solution to the diameter k-clustering problem (Problem 2.7) with three times the cost of an optimal solution.*



(a) Lower bound for the metric based on the L_∞-norm.



(b) Lower bound for the metric based on the L_2-norm. The points C, D, G, H have a z-coordinate of 0, while the points A, B, E, F have a z-coordinate of $2\sqrt{x}$.

Proof. We prove the proposition by constructing a data set for $k = 4$. Consider eight points $A, \ldots, H \in \mathbb{R}^2$ with

$$\begin{aligned} A &= (0,1), & E &= (-1,2), \\ B &= (1,0), & F &= (2,1), \\ C &= (0,-1), & G &= (1,-2), \\ D &= (-1,0), & H &= (-2,-1). \end{aligned}$$

The optimal 4-clustering of these points is

$$\mathcal{C}_4^{\mathrm{opt}} = \{\{A, E\}, \{B, F\}, \{C, G\}, \{D, H\}\}$$

which has a maximum L_∞-diameter of 1 (cf. Figure 15a). However, it is also possible that Algorithm 3.3 starts by merging A with B and C with D. Then, in the third step, the algorithm merges E or F with $\{A, B\}$, G or H with $\{C, D\}$, or $\{A, B\}$ with $\{C, D\}$. We assume the latter. Thus, the fourth merge step creates a cluster of L_∞-diameter 3. □

3.6.3 EUCLIDEAN METRIC AND $d = 3$

For the Euclidean case, we are able to construct a 3-dimensional data set that yields a lower bound of 2.56. This is below the upper bound of 3 from the one-dimensional case. Therefore, this data set does not show an impact of the dimension d in the Euclidean case as in the previous section. However, this lower bound is still better than the lower bound of 2.5 from the one-dimensional case. This suggests that in higher dimensions it might be easier to construct good lower bounds.

Proposition 3.31. *For the Euclidean metric there exists a data set $X \subset \mathbb{R}^3$ such that Algorithm 3.3 computes a solution to the diameter k-clustering problem (Problem 2.7) with cost 2.56 times the cost of an optimal solution.*

Proof. We prove the proposition by constructing a data set for $k = 4$. For any fixed $x \in \mathbb{R}$ with $0 < x < 2$ consider eight points $A, \dots, H \in \mathbb{R}^2$ with

$$\begin{aligned}
A &= (-1,\ 1, 2\sqrt{x}), & E &= (-(1+x),\ 1+\sqrt{4-x^2}\ , 2\sqrt{x}),\\
B &= (\ 1,\ 1, 2\sqrt{x}), & F &= (\ 1+x\ ,\ 1+\sqrt{4-x^2}\ , 2\sqrt{x}),\\
C &= (-1, -1,\ 0), & G &= (-(1+x), -(1+\sqrt{4-x^2}),\ 0),\\
D &= (\ 1, -1,\ 0), & H &= (\ 1+x\ , -(1+\sqrt{4-x^2}),\ 0).
\end{aligned}$$

The optimal 4-clustering of these points is

$$\mathcal{C}_4^{\mathrm{opt}} = \{\{A, E\}, \{B, F\}, \{C, G\}, \{D, H\}\},$$

which has a maximum L_2-diameter of 2 (cf. Figure 15b). However, since $\|A - B\| = \|C - D\| = 2$ it is possible that Algorithm 3.3 starts by merging A with B and C with D. Then, the cheapest merge adds one of the points E, F to the cluster $\{A, B\}$ or it adds one of the points G, H to the cluster $\{C, D\}$ or it merges $\{A, B\}$ with $\{C, D\}$. We assume the latter. The resulting cluster $\{A, B, C, D\}$ has a diameter of $2\sqrt{2+x}$. Then, in the fourth merge step, the algorithm will either merge one of the pairs E, F and G, H or one of the pairs E, G and F, H. The choice depends on the parameter x. Note that Algorithm 3.3 will not merge the cluster $\{A, B, C, D\}$ with one of the remaining four points, since this is always more expensive. The diameter of the created cluster is maximized for $x \approx 1.56$. If we fix $x = 1.56$, the algorithm merges E with F or G with H. This results in a 4-clustering of cost 5.12, while the optimal solution has cost 2. $\square$

3.6.4 l_p-METRIC $(1 \leqslant p \leqslant \infty)$ IN VARIABLE DIMENSION

In the following, we consider the diameter k-clustering problem with respect to the metric based on the L_1-norm. We provide a data set with dimension $O(k)$ such that Algorithm 3.3 computes a solution with an approximation factor of $\Omega(\log k)$.

Proposition 3.32. *For the metric based on the L_1-norm, there exists a data set $X \subset \mathbb{R}^d$ with $d = k + \log_2 k$ such that Algorithm 3.3 computes a solution to the diameter k-clustering problem (Problem 2.7) with $\frac{1}{2}\log_2 k$ times the cost of an optimal solution.*

Proof. For the sake of simplicity, assume k to be a power of 2. In the following, we consider the $(k + \log_2 k)$-dimensional set X of $|X| = k^2$ points defined by

$$X := \left\{ \begin{bmatrix} e_i \\ b \end{bmatrix} \middle|\ \forall 1 \leqslant i \leqslant k \text{ and } b \in \{0,1\}^{\log_2 k} \right\}.$$

Here, $e_i \in \mathbb{R}^k$ denotes the i-th canonical unit vector. Consider the k-clustering

$$\mathcal{C}_k^* := \left\{ C_b \,\middle|\, b \in \{0,1\}^{\log_2 k} \right\},$$

where for each $b \in \{0,1\}^{\log_2 k}$ the cluster C_b is given by

$$C_b := \left\{ \begin{bmatrix} e_i \\ b \end{bmatrix} \,\middle|\, \forall 1 \leqslant i \leqslant k \right\}.$$

The largest diameter of $\mathcal{C}_k^*$ is $\mathrm{cost}_{\mathrm{diam}}(\mathcal{C}_k^*) = 2$. Hence, for the diameter opt_k of an optimal solution, it follows that

$$\mathrm{opt}_k \leqslant 2. \tag{14}$$

To investigate the merge steps of Algorithm 3.3, note that

$$\mathrm{diam}\left(\left\{ \begin{bmatrix} e_i \\ b_1 \end{bmatrix}, \begin{bmatrix} e_j \\ b_2 \end{bmatrix} \right\} \right) = \begin{cases} h(b_1,b_2) & \text{if } i = j \\ 2 + h(b_1,b_2) & \text{if } i \neq j \end{cases}$$

where $h(b_1,b_2)$ denotes the Hamming distance between the strings $b_1, b_2 \in \{0,1\}^{\log_2 k}$. Hence, we may assume that Algorithm 3.3 starts by merging the points $[e_i, 0, b']^\top$ and $[e_i, 1, b']^\top$ for all $1 \leqslant i \leqslant k$ and all $b' \in \{0,1\}^{\log_2(k)-1}$, thereby forming $\frac{1}{2}k^2$ clusters of diameter 1.

Next, we show inductively that Algorithm 3.3 keeps merging pairs of clusters that agree on the first k coordinates until the algorithm halts. Assume that there is some number $1 \leqslant t \leqslant \log_2 k$ such that the clustering computed so far consists solely of the clusters

$$C_{i,b'}^{(t)} = \left\{ \begin{bmatrix} e_i \\ b \\ b' \end{bmatrix} \,\middle|\, b \in \{0,1\}^t \right\}$$

for all $1 \leqslant i \leqslant k$ and all $b' \in \{0,1\}^{\log_2(k)-t}$. Note that for $t = 1$ this is the case after the first $\frac{1}{2}k^2$ merges. Then,

$$\mathrm{diam}\left(C_{i,b_1}^{(t)} \cup C_{j,b_2}^{(t)} \right) = \begin{cases} t + h(b_1,b_2) & \text{if } i = j \\ 2 + t + h(b_1,b_2) & \text{if } i \neq j \end{cases}.$$

Hence, as above, we may assume that in the next $\frac{1}{2^{t+1}}k^2$ steps Algorithm 3.3 merges the clusters $C_{i,0b'}^{(t)}$ and $C_{i,1b'}^{(t)}$ for all $1 \leqslant i \leqslant k$ and all $b' \in \{0,1\}^{\log_2(k)-(t+1)}$. The resulting clusters have a diameter of $t+1$. Furthermore,

$$C_{i,b'}^{(t+1)} = C_{i,0b'}^{(t)} \cup C_{i,1b'}^{(t)}.$$

Algorithm 3.3 keeps merging clusters in this way until after $t = \log_2 k$ rounds we end up with the k-clustering $\mathcal{C}_k = \{C_i \mid 1 \leqslant i \leqslant k\}$ where

$$C_i = \left\{ \begin{bmatrix} e_i \\ b \end{bmatrix} \,\middle|\, b \in \{0,1\}^{\log_2 k} \right\}.$$

These clusters C_i are of diameter $\log_2 k$. Comparing to (14), we deduce that Algorithm 3.3 computes a solution to Problem 2.7 with at least $\frac{1}{2}\log_2 k$ times the cost of an optimal solution. □

Considering the diameter k-clustering problem with respect to an arbitrary L_p-metric (with $1 \leqslant p < \infty$), note that the behavior of Algorithm 3.3 does not change if we consider the p-th power of the L_p-distance instead of the L_p-distance. Also note that for all $x, y \in \{0,1\}^d$ we have $\|x-y\|_p^p = \|x-y\|_1$. Since the data set X from Proposition 3.32 is a subset of $\{0,1\}^d$, we immediately obtain the following corollary.

Corollary 3.33. *For the metric based on any L_p-norm with $1 \leqslant p < \infty$, there exists a data set $X \subset \mathbb{R}^d$ with $d = k + \log_2 k$ such that Algorithm 3.3 computes a solution to the diameter k-clustering problem (Problem 2.7) with $\sqrt[p]{\frac{1}{2}\log_2 k}$ times the cost of an optimal solution.*

Additionally, considering the diameter k-clustering problem with respect to the L_∞-metric, it is known that subset of n points from an arbitrary metric space can be embedded isometrically into $(\mathbb{R}^n, L_\infty)$ [Fréchet, 1910]. That is, the distance between the embedded points is equal to the original distances. The construction of the embedded set is simple. For $1 \leqslant i, j \leqslant n$, let d_{ij} be the distance between x_i and x_j in the metric space. Then, for each point x_i with $1 \leqslant i \leqslant n$ define $x_i' := (d_{i1}, \dots, d_{in})^T$. It immediately follows $\|x_i' - x_j'\|_\infty = d_{ij}$.

Hence, the data set of size $n = k^2$ from Proposition 3.32 yields a data set in $\mathbb{R}^{k^2}$ satisfying the same approximation bound with respect to the L_∞-distance. We obtain the following corollary.

Corollary 3.34. *For the metric based on the L_∞-norm, there exists a data set $X \subset \mathbb{R}^d$ with $d = k^2$ such that Algorithm 3.3 computes a solution to the diameter k-clustering problem (Problem 2.7) with $\frac{1}{2}\log_2 k$ times the cost of an optimal solution.*

THE DISCRETE k-CENTER PROBLEM

The data set X from Proposition 3.32 also yields lower bounds for the approximation factor of the agglomerative solution to the discrete k-center problem. Just note that for the data set X in every step of the algorithm the minimal discrete radius of a cluster equals the diameter of the cluster. We immediately obtain the following corollaries.

Corollary 3.35. *For the metric based on any L_p-norm with $1 \leqslant p < \infty$, there exists a data set $X \subset \mathbb{R}^d$ with $d = k + \log_2 k$ such that Algorithm 3.1 computes a solution to the discrete k-center problem (Problem 2.9) with $\sqrt[p]{\frac{1}{2}\log_2 k}$ times the cost of an optimal solution.*

Corollary 3.36. *For the metric based on the L_∞-norm, there exists a data set $X \subset \mathbb{R}^d$ with $d = k^2$ such that Algorithm 3.1 computes a solution to the discrete k-center problem (Problem 2.9) with $\frac{1}{2}\log_2 k$ times the cost of an optimal solution.*

Moreover, in case of the L_2-norm, we obtain the following corollary.

Corollary 3.37. *For the metric based on the* L_2*-norm, there exists a data set* $X \subset \mathbb{R}^d$ *with* $d = O(\log^3 k)$ *such that Algorithm 3.1 computes a solution to the discrete* k*-center problem (Problem 2.9) with* $\Omega(\sqrt{\log k})$ *times the cost of an optimal solution.*

Note that Corollary 3.37 follows by embedding the data set from Corollary 3.35 into the $O(\log^3 k)$-dimensional Euclidean space without altering the behavior of the agglomerative algorithm or the lower bound of $\Omega(\sqrt{\log k})$ (using the Johnson-Lindenstrauss embedding [Johnson and Joram, 1984]). For this embedded data set, the bound given in Section 3.2 states an upper bound of $20d + 2\log(k) + 2 = O(\log^3 k)$ times the cost of an optimal solution. Hence, in case of the discrete k-center problem using the L_2-metric, the upper bound from our analysis almost matches the lower bound.

Furthermore, this implies that the approximation factor of Algorithm 3.1 cannot be simultaneously independent of d and $\log k$. More precisely, the approximation factor cannot be sublinear in $\sqrt[6]{d}$ and in $\sqrt{\log k}$.

Part II

ON PARAMETER ESTIMATION

The second part of this thesis is about the parameter estimation problem for statistical models. Statistical models offer an alternate view on the structure of a given data set that differs substantially from the hard clustering approach discussed in Part I. Using this approach, the data set is assumed to be generated by a stochastic process. Then, the parameters of this process can be interpreted as a description of the structure of the data. While a partition of the data as computed by hard clustering algorithms is only valid for the given data set, the parameters of the stochastic process may also be used to describe other data sets generated by the same process.

The main result of Part II is the analysis of a probabilistic variant of the classical EM algorithm for the parameter estimation problem of Gaussian mixture models. We show that the probabilistic algorithm is considerably faster than the classical algorithm while, with high probability, its parameter updates are close to the deterministic EM updates. Furthermore, we confirm our theoretical results in an experimental analysis.

4

STATISTICAL MODELS

One way to deal with large data sets is to reduce the amount of information that has to be processed by investigating the distribution of the data. That is, if we are merely interested in the correlation of the data, then a loss of the information residing in the single data elements is acceptable. In this chapter we deal with the description of data by statistical models. That is, we assume that a given data set was generated by a particular statistical process. Then, the parameters of this process give a small description of the data set.

In this chapter we discuss the problem of obtaining suitable parameters for particular statistical models. We focus our attention on the well understood Gaussian mixture models. Gaussian mixtures are frequently used in practice since they are suitable for the explanation of many real world data sets. One often cited example is the Old Faithful data set. It is named after a famous geyser in the Yellowstone National Park where its data is taken from. The data set is two-dimensional and contains 272 entries. Each entry corresponds to one observation of an eruption and consists of the eruption time in minutes and the waiting time to the next eruption. The data set is shown in Figure 15. We see that the eruptions seem to follow two different time patterns. The measurements decompose into two ellip-

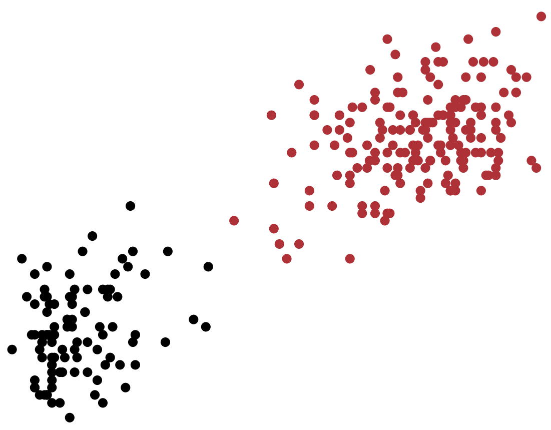


Figure 15: The Old Faithful data set.

soidally shaped point clouds. As we will discuss in Section 4.3, this makes the data set a suitable candidate for Gaussian mixture models.

For the understanding of this and the following chapters, some basic knowledge of discrete and continuous probability theory is indispensable. In the following, without introducing the necessary concepts, we denote the probability of an observation A by $\Pr(A)$ and we denote the probability of A given an observation B by $\Pr(A \mid B)$. For observations from continuous domains, $\Pr(\cdot)$ denotes the probability density function. Although formally incorrect, we use the term probability in the discrete as well as in the continuous case. However, it is always clear from the context whether a probability or density function is meant. Furthermore, we denote random variables by upper case letters and concrete observations by lower case letters. That is, $\Pr(X = x)$ denotes the probability that the random variable X takes on the value x (in the continuous case, it denotes the density of X at the point x). If the focus does not lie on the density of concrete observations but on the density function, then we simply write $\Pr(X)$ which strictly speaking is a random variable itself. A short introduction to probability theory can be found in [Bishop, 2006]. For a profound treatment of the matter see [Grimmett and Stirzaker, 2001].

4.1 THE PARAMETER ESTIMATION PROBLEM

We consider the parameters to be given as a vector θ from a proper parameter domain.

A *statistical model* for some domain $\mathbb{X}$ consists of a probability distribution on $\mathbb{X}$ and a set of associated parameters. The probability distribution is given by a density function $p_\theta : \mathbb{X} \to \mathbb{R}$ that is governed by a parameter vector θ. Then, data from the domain $\mathbb{X}$ is modeled by a random variable $X \in \mathbb{X}$ with

$$\Pr(X = x \mid \theta) = p_\theta(x)$$

for $x \in \mathbb{X}$. If we assume that the data consists of $n \in \mathbb{N}$ elements that are drawn independently from the same distribution, we denote $X = (X_1, \ldots, X_n) \in \mathbb{X}^n$ for independent and identically distributed random variables $X_1, \ldots, X_n \in \mathbb{X}$. Then, the density p_θ is extended to the domain $\mathbb{X}^n$ by computing the product probabilities over the single observations. More precisely,

$$\Pr(X = (x_1, \ldots, x_n) \mid \theta) = p_\theta(x_1, \ldots, x_n) = \prod_{i=1}^{n} p_\theta(x_i).$$

Note that we denote data sets by ordered tuples. Besides simplifying the discussion (e. g., regarding repetitions) it is natural to look at the independent draws from the same distribution as a sequential process. Since we consider the data to be the result of a random process, we use the terms data (set) and observation interchangeably.

Using a statistical model to describe a given data set, the central question is how to optimize the parameter vector θ. A natural approach is to try and maximize the probability that θ is responsible for a given observation X, i. e., to compute

$$\hat{\theta} := \arg\max_{\theta} \Pr(\theta \mid X) \,.$$

$\Pr(\theta \mid X)$ is called the *posterior probability* since it is obtained after the observation of X. However, for this probability to be well defined, we need to assume a probability distribution defined on the domain of the parameters. Then, using the probability $\Pr(\theta)$ (called *prior probability*), Bayes' theorem yields

$$\Pr(\theta \mid X) = \frac{\Pr(X \mid \theta) \Pr(\theta)}{\Pr(X)},$$

where $\Pr(X \mid \theta)$ is given by the statistical model. Since the denominator on the right-hand side does not depend on θ, we can state

$$\Pr(\theta \mid X) \propto \Pr(X \mid \theta) \Pr(\theta) \,.$$

The approach to optimize θ based on this relationship is called the *Bayesian approach*.

A resource for the Bayesian approach is [Bishop, 2006].

In this thesis we consider an alternate approach called the *frequentist approach*. Adopting the frequentist approach, no assumptions are made on the parameters. The optimization of the parameter vector θ is done by maximizing the probability of the observation X given θ, i. e., to compute

$$\theta^* := \arg\max_{\theta} \Pr(X \mid \theta) \,.$$

It is important to note that this argumentum maximi is not guaranteed to be well defined. The probability $\Pr(X \mid \theta)$ does not have to have a global maximum (e. g., for Gaussian mixtures, cf. Section 4.3.2). However, for a well posed problem there should at least exist a local maximum that can be searched for.

Another well known pitfall of this approach is its vulnerability to the problem of overfitting. This means that a parameter vector θ at a global or local maximum of $\Pr(X \mid \theta)$ may be so well adapted to the concrete observation X that the resulting statistical model is less suitable to describe similar data from the same application.

For a detailed treatment of overfitting see [Hawkins, 2004].

If the probability $\Pr(X \mid \theta)$ is viewed as a function of the parameter vector θ, then it is called the *likelihood* of θ given the observation X and is denoted as

$$\mathcal{L}(\theta \mid X) = \Pr(X \mid \theta) \,.$$

Therefore, θ^* as defined above, is called the *maximum likelihood estimate*. The problem of computing or approximating the maximum likelihood estimate is called the *parameter estimation problem*.

Instead of maximizing the likelihood it is equivalent and sometimes analytically more convenient to maximize its logarithm (cf. Section 4.3). The function $\ln \mathcal{L}(\theta \mid X)$ is called the *log-likelihood* of θ given

X. For observations $X = (X_1, \ldots, X_n)$ as described above, the log-likelihood takes the form

$$\ln \prod_{i=1}^{n} p_\theta(X_i) = \sum_{i=1}^{n} \ln p_\theta(X_i) = \sum_{i=1}^{n} \ln \mathcal{L}(\theta \,|\, X_i)\,.$$

That is, each sub-observation X_i contributes an additive term to the log-likelihood cost function. This contribution is simply the log-likelihood of the parameter vector θ given the single observation X_i.

In order to be able to interpret the optimization of θ as the minimization of a cost function, it is common to minimize the *negative log-likelihood*, i.e., to compute

$$\theta^* := \arg\min_\theta \, -\ln \mathcal{L}(\theta \,|\, X)\,.$$

4.2 GENERAL MIXTURE MODELS

In this section, we study the parameter estimation problem for a particular class of statistical models, namely the mixture models or mixture distributions. Mixture models are well suited to model data that can be partitioned into several subsets which can be seen as to be generated by different distributions over the same domain $\mathbb{X}$. For a mixture of k distributions, the parameter vector takes the form

$$\theta = (w_1, \ldots, w_k, \theta_1, \ldots, \theta_k),$$

where $w_1, \ldots, w_k \in \mathbb{R}^+$ with $\sum_{j=1}^{k} w_j = 1$ are called the weights and θ_j is the parameter vector for the j-th component distribution. We denote the probability density for the corresponding mixture by

$$p_\theta(x) = \sum_{j=1}^{k} w_j p_{\theta_j}(x),$$

where p_{θ_j} denotes the probability density of the j-th component distribution. Since the weights sum up to one, it follows that

$$\int_{\mathbb{X}} p_\theta(x)\; \partial x = \sum_{j=1}^{k} w_j \int_{\mathbb{X}} p_{\theta_j}(x)\; \partial x = \sum_{j=1}^{k} w_j = 1.$$

Thus, $p_\theta(x)$ is indeed a probability density. Drawing a single observation $X = x$ from a mixture distribution can be described as a two step process. First, choose one of the component distributions with probability proportional to its weight. Second, draw x from the chosen component distribution.

For the likelihood of θ given an observation $X = (x_1, \ldots, x_n)$ it follows that

$$\mathcal{L}(\theta \,|\, X) = \prod_{i=1}^{n} p_\theta(x_i) = \prod_{i=1}^{n} \sum_{j=1}^{k} w_j p_{\theta_j}(x_i)$$

and for the log-likelihood we get

$$\ln \mathcal{L}(\theta \mid X) = \sum_{i=1}^{n} \ln p_\theta(x_i) = \sum_{i=1}^{n} \ln \sum_{j=1}^{k} w_j p_{\theta_j}(x_i).$$

Note that this expression contains the logarithm of a sum. Therefore, in the mixture case, the likelihood is difficult to optimize directly. However, in Chapter 5 we discuss an algorithmic approach to approximate the maximum likelihood estimate.

4.3 GAUSSIAN MIXTURE MODELS

Gaussian distributions (or short, Gaussians) are statistical models for points from the d-dimensional space $\mathbb{R}^d$. A single Gaussian is an unimodal distribution. That is, points drawn from a single Gaussian are concentrated around a single point, called the *mean* of the Gaussian. Furthermore, the points form an ellipsoidal shape and their density drops exponentially with the distance to the mean. By mixing several Gaussian distributions, it is possible to model multimodal data sets.

4.3.1 SINGLE GAUSSIAN DISTRIBUTIONS

The Gaussian distribution (also known as the normal distribution) is a continuous unimodal probability distribution for real valued random variables. In the one-dimensional case, it is called the *univariate Gaussian distribution* and is given by the probability density $\mathcal{N}_{\mu,\sigma^2} : \mathbb{R} \to \mathbb{R}$ with

$$\mathcal{N}_{\mu,\sigma^2}(x) := \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right),$$

where $\mu, \sigma^2 \in \mathbb{R}$ and $\sigma^2 > 0$. The expected value of a random variable $X \sim \mathcal{N}_{\mu,\sigma^2}$ is given by

For a derivation of these equalities see Section 2.3 in [Bishop, 2006].

$$\mathrm{E}[X] = \int_{-\infty}^{\infty} x \cdot \mathcal{N}_{\mu,\sigma^2}(x)\ \partial x = \mu$$

and the variance of X is given by

$$\mathrm{E}\left[(X-\mu)^2\right] = \int_{-\infty}^{\infty} (x-\mu)^2 \cdot \mathcal{N}_{\mu,\sigma^2}(x)\ \partial x = \sigma^2.$$

Therefore, the parameter μ is called the mean of the Gaussian distribution and the parameter σ^2 is called its variance.

The extension of the Gaussian distribution to the d-dimensional space $\mathbb{R}^d$ is called the *multivariate Gaussian distribution* (cf. Figure 16) and is given by the probability density $\mathcal{N}_{\mu,\Sigma} : \mathbb{R}^d \to \mathbb{R}$ with

$$\mathcal{N}_{\mu,\Sigma}(x) := \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right), \qquad (15)$$

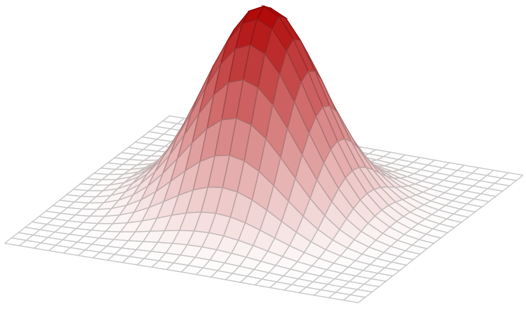


Figure 16: Density of a two-dimensional Gaussian distribution.

where $\mu \in \mathbb{R}^d$ and $\Sigma \in \mathbb{R}^{d\times d}$ is a symmetric positive-definite matrix. Analogously to the univariate case, the expected value of a random variable $X \sim \mathcal{N}_{\mu,\Sigma}$ is given by

$$\mathrm{E}\left[X\right] = \int_{\mathbb{R}^d} x \cdot \mathcal{N}_{\mu,\Sigma}(x)\ \partial x = \mu$$

and the covariance matrix of X is given by

$$\mathrm{E}\left[(X-\mu)(X-\mu)^T\right] = \int_{\mathbb{R}^d} (x-\mu)(x-\mu)^T \cdot \mathcal{N}_{\mu,\sigma^2}(x)\ \partial x = \Sigma.$$

Therefore, the parameter Σ is called the covariance matrix of the Gaussian distribution. In the following, we take a closer look at the geometrical form of the Gaussian density for different types of covariance matrices.

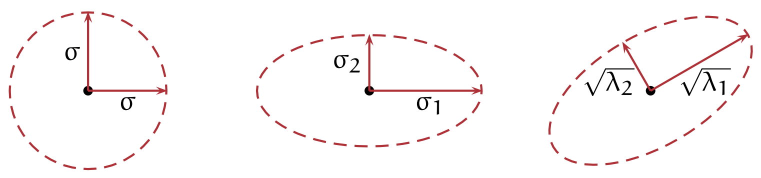
For Σ being a multiple of the identity matrix, i. e.,

$$\Sigma = \sigma^2 I_d \in \mathbb{R}^{d\times d}$$

for $\sigma^2 > 0$, we deduce

$$\begin{aligned}\mathcal{N}_{\mu,\sigma^2 I_d}(x) &= \frac{1}{\sqrt{(2\pi\sigma^2)^d}} \exp\left(-\frac{1}{2\sigma^2}\|x-\mu\|^2\right) \qquad (16)\\ &= \frac{1}{\sqrt{(2\pi\sigma^2)^d}} \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{d}(x_i-\mu_i)^2\right)\\ &= \prod_{i=1}^{d} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(x_i-\mu_i)^2\right)\\ &= \prod_{i=1}^{d} \mathcal{N}_{\mu_i,\sigma^2}(x_i),\end{aligned}$$

where $x = (x_1,\ldots,x_d)^T$ and $\mu = (\mu_1,\ldots,\mu_d)^T$. That is, each coordinate of a random variable $X \sim \mathcal{N}_{\mu,\sigma^2 I_d}$ is distributed independently according to a univariate Gaussian around the corresponding coordinate of μ and with variance σ^2. Furthermore, Equation (16) yields that points with the same density lie on a sphere centered at μ (cf. Figure 17a).



(a) $\Sigma = \sigma^2 I_d$ (b) diagonal Σ (c) sym. pos.-def. Σ

Figure 17: Points with constant density for two-dimensional Gaussians with different types of covariance matrices.

If Σ is a diagonal matrix with diagonal elements $\sigma_1^2, \dots, \sigma_d^2 \in \mathbb{R}^+$, then $\Sigma^{-1} = \operatorname{Diag}\left(\frac{1}{\sigma_1^2}, \dots, \frac{1}{\sigma_1^2}\right)$ and

$$\begin{aligned}\mathcal{N}_{\mu,\operatorname{Diag}(\sigma_1^2,\dots,\sigma_d^2)}(x) &= \frac{1}{\sqrt{(2\pi)^d \prod_{i=1}^d \sigma_i^2}} \exp\left(\sum_{i=1}^d -\frac{(x_i-\mu_i)^2}{2\sigma_i^2}\right) \\ &= \prod_{i=1}^d \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(x_i-\mu_i)^2}{2\sigma_i^2}\right) \\ &= \prod_{i=1}^d \mathcal{N}_{\mu_i,\sigma_i^2}(x_i).\end{aligned}$$

That is, the i-th coordinate of $X \sim \mathcal{N}_{\mu,\operatorname{Diag}(\sigma_1^2,\dots,\sigma_d^2)}$ is distributed independently according to a univariate Gaussian around μ_i again. However, in the diagonal case the variance of the i-th coordinate is σ_i^2. That is, a Gaussian with diagonal covariance matrix can be thought of to be derived from a Gaussian with covariance I_d by shrinking and stretching along the coordinate axes. More precisely, by multiplying the i-th coordinate of $X = (X_1, \dots, X_d) \sim \mathcal{N}_{\mu,I_d}$ with the standard deviation σ_i for $i = 1, \dots, d$, we get a random variable $Y = (\sigma_1 X_1, \dots, \sigma_d X_d)^T \sim \mathcal{N}_{\mu,\operatorname{Diag}(\sigma_1^2,\dots,\sigma_d^2)}$ (cf. Figure 17b).

For a general symmetric positive-definite Σ, we consider its unique Cholesky Factorization

See Theorem 4.2.7 in [Golub and Van Loan, 2012].

$$\Sigma = LL^T,$$

where L is an invertible lower triangular matrix with positive diagonal entries. The matrix L can be uniquely decomposed as

$$L = VD$$

for an orthonormal matrix V and a diagonal matrix D with positive entries. While the linear map $x \mapsto Dx$ stretches (or shrinks) x along the coordinate axes, the linear map $x \mapsto Vx$ is a rotation around and reflection at the origin. That is, applying the linear map $x \mapsto Lx$ to the d-dimensional unit ball results in an ellipsoid centered at the origin. The above decomposition of Σ is unique. Furthermore, for any orthonormal matrix V and any diagonal matrix D with positive entries, the matrix LL^T with $L = VD$ is symmetric positive-definite. Therefore, it is common to identify a symmetric positive-definite matrix with the associated ellipsoid.

We use the decomposition of Σ stated above to decompose Σ^{-1} as

$$\begin{aligned}\Sigma^{-1} &= \left(L^T\right)^{-1} L^{-1} = \left(L^{-1}\right)^T L^{-1} \\ &= \left((VD)^{-1}\right)^T (VD)^{-1} \\ &= \left(D^{-1}V^{-1}\right)^T D^{-1}V^{-1}.\end{aligned}$$

Then, for the density of the multivariate Gaussian distribution, we deduce

$$\mathcal{N}_{\mu,\Sigma}(x) = \frac{1}{\sqrt{(2\pi)^d|\Sigma|}} \exp\left(-\frac{\|D^{-1}V^{-1}(x-\mu)\|^2}{2}\right). \tag{17}$$

That is, a Gaussian with general symmetric positive-definite covariance matrix can be thought of to be derived from a Gaussian with diagonal covariance by rotating around the mean (cf. Figure 17c).

By ignoring the normalization factor $|\Sigma|^{-\frac{1}{2}}$, Equation (17) offers an alternate view on the density function. The evaluation of $\mathcal{N}_{\mu,\Sigma}$ at a point x can be interpreted as a two step process. In the first step, x is rotated around μ by the linear map

$$x \mapsto V^{-1}(x-\mu) + \mu$$

and afterwards the coordinates of x are scaled relative to μ by the linear map

$$x \mapsto D^{-1}(x-\mu) + \mu.$$

This yields the point

$$y = D^{-1}V^{-1}(x-\mu) + \mu. \tag{18}$$

In the second step, the point y is passed to the density $\mathcal{N}_{\mu,I_d}$.

Points with constant density have to be mapped to a sphere around μ in the first step (cf. Equation (16)). To see which points satisfy this property, let z be an arbitrary point on the translation of the unit sphere centered at μ. Then, the linear map

$$x \mapsto VD(x-\mu) + \mu$$

maps z onto the translation of the ellipsoid associated to Σ centered at μ. Substituting $x = VD(z-\mu) + \mu$ in Equation 18 yields

$$y = D^{-1}V^{-1}VD(z-\mu) + \mu = z.$$

This equality also holds for z lying on an arbitrary sphere centered at μ. We conclude that points with constant density lie on concentric, confocal ellipsoids. More precisely, these ellipsoids result from scaling the ellipsoid associated to Σ and translating it to be centered at μ.

Note that the lengths of the semi-axes of the ellipsoid associated to Σ are the square roots of the eigenvalues of Σ (cf. Figure 17c).

See Chapter 2.3 in [Bishop, 2006].

For a single Gaussian distribution, the parameter vector θ takes the form

$$\theta = (\mu, \Sigma).$$

Then, for the likelihood of θ given an observation $X = (x_1, \dots, x_n)$, we get

$$\begin{aligned}\mathcal{L}(\theta \mid X) = \Pr(X \mid \theta) &= \prod_{i=1}^{n} \mathcal{N}_{\mu,\Sigma}(x_i) \\ &= \prod_{i=1}^{n} \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} \exp\left(-\frac{1}{2}(x_i - \mu)^T \Sigma^{-1} (x_i - \mu)\right).\end{aligned}$$

As indicated in the previous section, it is analytically much simpler to consider the log-likelihood. The result of applying the logarithm is that the product transforms into a sum and that the exponential function disappears. For the negative log-likelihood of a parameter vector $\theta = (\mu, \Sigma)$ given an observation $X = (x_1, \dots, x_n)$, we get

$$\begin{aligned}-\ln \mathcal{L}(\mu, \Sigma \mid X) &= \sum_{i=1}^{n} -\ln \mathcal{N}_{\mu,\Sigma}(x_i) \qquad (19)\\ &= \sum_{i=1}^{n} \left(\ln \sqrt{(2\pi)^d |\Sigma|} + \frac{1}{2}(x_i - \mu)^T \Sigma^{-1} (x_i - \mu)\right) \\ &= \sum_{i=1}^{n} \left(\frac{d}{2} \ln 2\pi + \frac{1}{2} \ln |\Sigma| + \frac{1}{2}(x_i - \mu)^T \Sigma^{-1} (x_i - \mu)\right) \\ &= \frac{nd}{2} \ln 2\pi + \frac{n}{2} \ln |\Sigma| + \sum_{i=1}^{n} \frac{1}{2}(x_i - \mu)^T \Sigma^{-1} (x_i - \mu).\end{aligned}$$

The parameter estimation problem for a single Gaussian distribution, i. e., the maximization of $\mathcal{L}(\mu, \Sigma \mid X)$ with respect to μ and Σ, is easy to solve. There exist closed form formulas for the values μ^* and Σ^* of the maximum likelihood estimate θ^*, namely

For a derivation of these equalities see [Bilmes, 1997].

$$\mu^* = \frac{1}{n} \sum_{i=1}^{n} x_i \quad \text{and} \quad \Sigma^* = \frac{1}{n} \sum_{i=1}^{n} (x_i - \mu^*)(x_i - \mu^*)^T.$$

4.3.2 MIXTURE OF GAUSSIAN DISTRIBUTIONS

As mentioned above, single Gaussian distributions are well suited to model ellipsoidally shaped data sets. However, if the data set splits into several possibly separated ellipsoidally shaped parts, then we need a more general type of distribution called *Gaussian mixture distribution*.

Gaussian mixtures are a special case of mixture distributions (cf. Section 4.2), where all components are single Gaussian distributions.

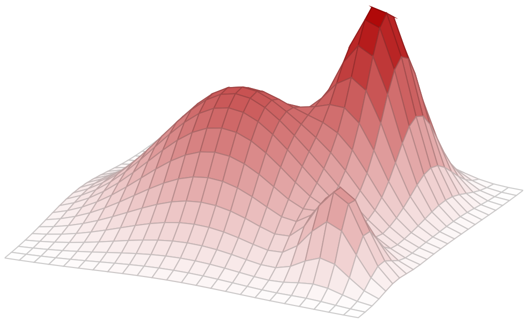


Figure 18: Density of a mixture of three Gaussian distributions.

For a mixture of k Gaussians, the parameter vectors $\theta_1, \dots, \theta_k$ of the component Gaussians take the form

$$\theta_j = (\mu_j, \Sigma_j),$$

where μ_j is the mean of the j-th component and Σ_j is its covariance matrix. Then, the parameter vector of the mixture takes the form

$$\theta = (w_1, \dots, w_k, \mu_1, \dots, \mu_k, \Sigma_1, \dots, \Sigma_k)$$

and the probability density $\mathcal{M}_\theta$ of the mixture (cf. Figure 18) is defined as

$$\mathcal{M}_\theta(x) := \sum_{j=1}^{k} w_j \mathcal{N}_{\mu_j, \Sigma_j}(x).$$

For the likelihood of θ given an observation $X = (x_1, \dots, x_n)$ it follows that

$$\mathcal{L}(\theta \,|\, X) = \prod_{i=1}^{n} \mathcal{M}_\theta(x_i) = \prod_{i=1}^{n} \sum_{j=1}^{k} w_j \mathcal{N}_{\mu_j, \Sigma_j}(x_i)$$

and for the log-likelihood we get

$$\ln \mathcal{L}(\theta \,|\, X) = \sum_{i=1}^{n} \ln \mathcal{M}_\theta(x_i) = \sum_{i=1}^{n} \ln \sum_{j=1}^{k} w_j \mathcal{N}_{\mu_j, \Sigma_j}(x_i). \tag{20}$$

Note that in contrast to the case of a single Gaussian distribution, we are not able to move the logarithm operator in front of the Gaussian density function (cf. Equation (19)). The summation over the k components thus prevents the derivation of a simple formula for the log-likelihood.

In case of Gaussian mixture models, it is hopeless to try and deduce closed formulas for the maximum likelihood estimate anyway. The problem is that for the mixture of at least two Gaussians the maximum likelihood estimate is not well defined. On the contrary, it is always possible to generate parameters with arbitrary large log-likelihood.

To see this, we fix an arbitrary point of the observed data $X = (x_1, \dots, x_n)$ and customize the parameters for one of the k component Gaussians. Without loss of generality, we consider the data point x_1 and the parameters w_1, μ_1, and Σ_1. While w_1 can be set to any constant $c \in \mathbb{R}$ with $0 < c < 1$, we define $\mu_1 = x_1$. Furthermore, for any $\alpha \in \mathbb{R}$ with $\alpha > 0$ we define $\Sigma^\alpha \in \mathbb{R}^{d \times d}$ to be a symmetric positive definite matrix with determinant $|\Sigma^\alpha| = \alpha$. Then, for α tending to zero and $x \neq x_1$, the density $\mathcal{N}_{\mu_1, \Sigma^\alpha}(x)$ also tends to zero (cf. Equation (15)). This is because of the exponential drop of the Gaussian density function. Thus, for $\Sigma_1 = \Sigma^\alpha$ and α small enough, the contribution of the points x_i with $x_i \neq x$ to the log-likelihood is dominated by the components 2 to k (cf. Equation (20)). More precisely, for $\alpha \to 0$ their contribution converges to some constant that depends on $\theta_2, \dots, \theta_k$. However, for α small enough, the contribution of $x = x_1$ to the log-likelihood is dominated by the first component. For $\alpha \to 0$ the density $\mathcal{N}_{\mu_1, \Sigma^\alpha}(x_1)$ converges to ∞. Thus, the contribution of $x = x_1$ also converges to ∞.

As a consequence, one should always keep in mind that in case of the mixture of Gaussian distributions the parameter estimation problem is an ill-posed problem.

4.4 HANDY NOTIONS FROM PROBABILITY THEORY

In this section, we introduce some notions from probability theory which we need for the discussion of the EM algorithm in Chapter 5 and for the analysis of the SEM algorithm in Chapter 6.

4.4.1 DISTINCTION BETWEEN DISTRIBUTION AND DENSITY

Probability distributions are usually given by the corresponding density functions. Therefore, it is common to identify a distribution with its density. Although formally not correct, this convention generally does not lead to any ambiguities. We join this common practice for the remainder of this thesis since it simplifies the discussion.

4.4.2 ON INFORMATION THEORY

In the following, we give a brief description of two concepts from information theory, namely the entropy of a random variable and the Kullback-Leibler divergence between two random variables. We define both quantities for the discrete as well as for the continuous case.

For the following definitions of entropy and differential entropy, we use the convention that

$$0 \ln 0 = 0$$

which arises from continuity since $x \ln x \to 0$ as $x \to 0$.

Definition 4.1. *Let* X *be a random variable over a discrete domain* $\mathbb{X}$ *with probability mass function* $p : \mathbb{X} \to \mathbb{R}$*. Then, the* entropy *of* X *is defined as*

$$H(X) := -\sum_{x \in \mathbb{X}} p(x) \ln p(x).$$

Since the entropy of X depends only on the probability mass p, it is also denoted as $H(p)$. The entropy of X is a non-negative quantity which can be interpreted as the amount of information that is obtained by observing the value of X. We use the entropy only for notational purposes. For a detailed discussion of the entropy see Chapter 2.1 in [Cover and Thomas, 2012]. The continuous correspondence of the entropy is called the differential entropy.

Definition 4.2. *Let* X *be a random variable over a continuous domain* $\mathbb{X}$ *with continuous and differentiable probability density function* $p : \mathbb{X} \to \mathbb{R}$*. Then, the* differential entropy *of* X *is defined as*

$$H(X) := -\int_{\mathbb{X}} p(x) \ln p(x) \, \partial x.$$

Again, we also denote the differential entropy of X as $H(p)$, since it depends only on the probability density p. For a discussion of the relation between entropy and differential entropy, see Chapter 8.3 in [Cover and Thomas, 2012].

For the following definitions of the Kullback-Leibler divergence, we use the conventions that

$$0 \ln \frac{0}{0} = 0 \quad \text{and} \quad p \ln \frac{p}{0} = \infty$$

for $p > 0$.

Definition 4.3. *Let* X, Y *be random variables over a discrete domain* $\mathbb{X}$*. Furthermore, let* p *and* q *be the probability mass functions of* X *and* Y*, respectively. Then, the* Kullback-Leibler divergence *between* X *and* Y *is defined as*

$$\mathrm{KL}(X \,\|\, Y) := \sum_{x \in \mathbb{X}} p(x) \ln \frac{p(x)}{q(x)}.$$

Note that the Kullback-Leibler divergence is not symmetric, which is the reason why it should not be called Kullback-Leibler distance. For a motivation and discussion of the Kullback-Leibler divergence see Chapter 2.3 in [Cover and Thomas, 2012]. Just as the entropy, the Kullback-Leibler divergence can also be defined for the continuous case.

Definition 4.4. *Let* X, Y *be random variables over a continuous domain* $\mathbb{X}$*. Let* p *and* q *be the probability density functions of* X *and* Y*, respectively. Furthermore, let* p *and* q *be continuous and differentiable. Then, the* Kullback-Leibler divergence *between* X *and* Y *is defined as*

$$\mathrm{KL}(X \,\|\, Y) := \int_{\mathbb{X}} p(x) \ln \frac{p(x)}{q(x)} \, \partial x.$$

A discussion of the Kullback-Leibler divergence for the continuous case can be found in Chapter 8.5 of [Cover and Thomas, 2012].

In the discrete as well as in the continuous case, we also denote $\mathrm{KL}(X \,\|\, Y)$ as $\mathrm{KL}(p \,\|\, q)$ since it depends only on p and q. Furthermore, the Kullback-Leibler divergence between X and Y is a non-negative quantity that is zero if and only if $p = q$.

5 THE CLASSICAL EM ALGORITHM

Throughout this chapter, we deal with an algorithmic approach to solving the parameter estimation problem for some fixed statistical model. More precisely, we discuss a general scheme for the approximation of the corresponding maximum likelihood estimate (see Chapter 4.1), called the Expectation-Maximization algorithm (or simply EM algorithm). Since its introduction in [Dempster et al., 1977] a lot of work has been done to analyze and improve the EM algorithm. Today it is very well understood [McLachlan and Krishnan, 2008]. In the following, we explain the algorithm, discuss its convergence properties, and apply it to general mixture distributions as well as Gaussian mixtures. The discussion of the general EM algorithm and its convergence properties follows the presentation in Chapter 2.3 of [Bishop, 2006]. The application to general and Gaussian mixtures is based on [Bilmes, 1997].

The EM algorithm is used when the observed data can be seen as incomplete and when the corresponding complete data helps significantly to solve the parameter estimation problem. The main scope of the algorithm are statistical models for which direct optimization is impossible. For example, let us assume that the observed data was generated by a mixture of Gaussian distributions. Then the computation of the maximum likelihood estimate is computationally intractable (see Chapter 4.3.2). However, if we knew for each observed point from which component distribution it was generated, then the solution could be computed separately for each component by closed form formulas (see Chapter 4.3.1). Therefore, for Gaussian mixture models we define the origin of each data point as the so-called hidden values.

In the following, we denote the incomplete observation by the random variable X and the hidden values by the random variable Z. Furthermore, we denote the joint distribution of the complete data (X, Z) by

Recall that we identify a distribution with its density.

$$p_\theta(x, z) = \Pr(X = x, Z = z \,|\, \theta)\,,$$

where θ denotes a vector of model parameters. Then, for the distribution of the incomplete data we deduce

$$p_\theta(x) = \int p_\theta(x, Z)\,\partial Z \;=\; \Pr(X = x \,|\, \theta)\,. \tag{21}$$

Algorithm 5.1: Classical EM algorithm

```
Input: incomplete observation $X$, initial model $\theta$

while <condition> do
1   | $q(Z) \longleftarrow \Pr(Z \mid X, \theta)$            // for hidden r.v. Z
2   | maximize $E_{Z\sim q}[\ln \Pr(X, Z \mid \theta)]$ w. r. t. $\theta$
end
return $\theta$
```

Note that for Gaussian mixtures the integral becomes a sum over the discrete space of possible assignments of each point to one of the component distributions. The complete data likelihood $p_\theta(X, Z)$ often has a much simpler form than the incomplete data likelihood $p_\theta(X)$. Using Bayes' theorem,

$$p_\theta(x, z) = \Pr(Z = z \mid \theta) \Pr(X = x \mid \theta, Z = z) .$$

In case of mixture distributions for example, this leads to simple update equations of the EM algorithm (see Section 5.2 and Section 5.3).

cf. Chapter 4.1

The parameter estimation problem given the incomplete observation X is to maximize the likelihood function

$$\mathcal{L}(\theta \mid X) = \Pr(X \mid \theta) = \int p_\theta(X, Z) \, \partial Z$$

over the choice of θ. For the discussion of the EM algorithm we furthermore need the distribution $p_{X,\theta}$ of the hidden values given the incomplete observation. Using Bayes' theorem,

$$p_{X,\theta}(z) = \frac{p_\theta(X, z)}{p_\theta(X)} = \Pr(Z = z \mid X, \theta) . \tag{22}$$

In Chapter 7.3 we describe a simple method of obtaining initial parameters.

The EM algorithm is an iterative method that, given an initial estimate of the model parameters, tries to approximate the maximum likelihood estimate by maximizing the expected complete data log-likelihood. That is, it computes

$$\theta^{\text{new}} := \operatorname*{arg\,max}_{\theta} \mathop{E}_{Z\sim p_{X,\theta^{\text{old}}}} [\ln \Pr(X, Z \mid \theta)] .$$

The expectation is taken over the distribution of the hidden values that is induced by the previous parameter estimate θ^{old}. The EM algorithm in its general form is stated as Algorithm 5.1, where the condition of the while loop may be filled in with any reasonable choice, e. g., a convergence criterion or a fixed number of iterations.

Each iteration of the loop in Algorithm 5.1 consists of two steps. Step 1 derives the distribution over the hidden values that is induced by the previous parameter estimate. Fixing this distribution, the expectation over the hidden values that is to be maximized can be stated as a function of the parameters θ. Therefore, the first step is called the *expectation step*. However, no particular expectation is computed in this first step. In applications of the EM algorithm for concrete statistical models, the derived distribution is often discrete as for example for Gaussian mixtures. Then, the implementation of the expectation step usually consists of precomputations of the corresponding density function for possible values of Z. In case of Gaussian mixture models, it is sufficient to precompute the probability that a certain point was generated by a certain component for each point and each component distribution (see Section 5.3). Step 2 is called the *maximization step* and computes the new set of parameters by maximizing the expectation over the choice of θ.

For the EM algorithm to be applicable, we have to assume that the maximization in Step 2 is computationally tractable. In practice, this is often the case (e. g., for mixtures of distributions from the exponential family). There also exist computationally simpler variants of the EM algorithm that may be used if the maximization step is not (efficiently) computable. The General EM (or GEM) algorithm for example tries to only increase the expectation in the second step (cf. [Bishop, 2006]). Furthermore, there exist probabilistic variants like the SEM algorithm discussed in Chapter 6. The SEM algorithm simplifies the computation by avoiding to deal with the expectation at all. Instead, it simply guesses the hidden values which considerably simplifies the maximization step.

To analyze the convergence properties of the EM algorithm in the following section, we assume that two preconditions are met. First, as mentioned above, the maximization step has to be computationally tractable. Second, the log-likelihood function for the considered statistical model and the expectation from the maximization step both have to be differentiable. The above mentioned and frequently used mixtures of distributions from the exponential family satisfy these preconditions.

5.1 CONVERGENCE OF THE EM ALGORITHM

The parameter vectors computed by the EM algorithm converge to a stationary point of the likelihood function. To see this, we here adopt the line of argument in Chapter 2.3 of [Bishop, 2006].

In this section, we denote the log-likelihood function given an incomplete observation X by $\ln p_\theta(X)$ as a function of θ. Note that applying the logarithm does not change the stationary points. *cf. Equation* (21)

Theorem 5.1. *Let θ^0 be the initial parameter estimate and for $i > 0$ let θ^i be the parameter estimate θ computed by Algorithm 5.1 in the i-th iteration of Step 2. Then, for increasing i the log-likelihood of θ^i converges to a stationary point of $\ln p_\theta(X)$.*

At first sight, the significance of Theorem 5.1 may appear low. It does not even guarantee the convergence to a local maximum, let

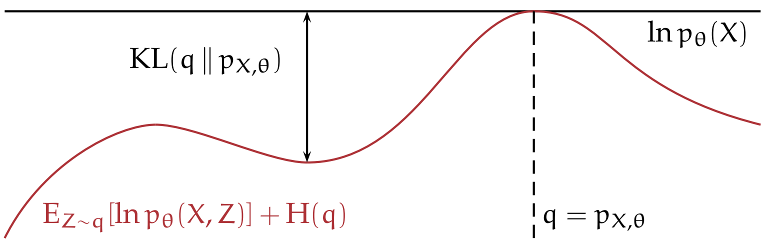

$\mathrm{KL}(q \,\|\, p_{X,\theta})$
$\ln p_\theta(X)$
$\mathrm{E}_{Z\sim q}[\ln p_\theta(X,Z)] + H(q)$
$q = p_{X,\theta}$

Figure 19: Decomposition of $\ln p_\theta(X)$ over the choice of q.

alone the convergence to a global one. However, in general we can not hope for more since it is known that the EM algorithm can get stuck on a saddle point of the likelihood function (see Chapter 3.6 of [McLachlan and Krishnan, 2008]). Moreover, since a global maximum does not even have to exist (cf. Chapter 4.3.2), this does not have to be an undesirable property. In practice, the local maxima may be the original target. And even a saddle point can be of great interest when the local maxima fall victim to the problem of overfitting mentioned in Chapter 4.1.

To prove Theorem 5.1, we use the following lemma, which decomposes the log-likelihood of the incomplete data as shown in Figure 19. Note that the complete data log-likelihood is denoted by $\ln p_\theta(X, Z)$ and recall that $p_{X,\theta}$ denotes the distribution over the hidden values that is induced by the incomplete observation X and the parameter estimate θ.

cf. Equation (22)

Lemma 5.2. *Let q be an arbitrary distribution over the hidden values Z. Then, the log-likelihood of a parameter vector θ given an observation X can be written as*

cf. Figure 19

$$\ln p_\theta(X) = \mathop{\mathrm{E}}_{Z\sim q}[\ln p_\theta(X,Z)] + H(q) + \mathrm{KL}(q \,\|\, p_{X,\theta}) \,.$$

Since the Kullback-Leibler divergence is always non-negative and it is zero if and only if both densities are identical (see Chapter 4.4.2), Lemma 5.2 immediately yields the following corollary.

Corollary 5.3. *Let q be an arbitrary distribution over the hidden values Z. Then, the log-likelihood of a parameter vector θ given an observation X can be lower bounded by*

$$\ln p_\theta(X) \geqslant \mathop{\mathrm{E}}_{Z\sim q}[\ln p_\theta(X,Z)] + H(q) \,.$$

Furthermore, the bound holds with equality if and only if $q = p_{X,\theta}$.

Proof of Lemma 5.2. Using the definition of the Kullback-Leibler divergence (see Chapter 4.4.2),

$$\begin{aligned}
&\ln p_\theta(X) - \mathrm{KL}(q \,\|\, p_{X,\theta}) \\
&= \ln p_\theta(X) + \int q(Z) \ln\left(\frac{p_{X,\theta}(Z)}{q(Z)}\right) \partial Z \\
&= \int q(Z) \left(\ln p_\theta(X) + \ln\left(\frac{p_{X,\theta}(Z)}{q(Z)}\right)\right) \partial Z \\
&= \int q(Z) \ln\left(\frac{\ln\left(p_\theta(X)\, p_{X,\theta}(Z)\right)}{q(Z)}\right) \partial Z.
\end{aligned}$$

From the definition of $p_{X,\theta}$ in Equation (22), we deduce

$$p_\theta(X)\, p_{X,\theta}(Z) = p_\theta(X, Z).$$

Thus,

$$\begin{aligned}
&\ln p_\theta(X) - \mathrm{KL}(q \,\|\, p_{X,\theta}) \\
&= \int q(Z) \ln\left(\frac{\ln p_\theta(X, Z)}{q(Z)}\right) \partial Z \\
&= \int q(Z) \ln p_\theta(X, Z)\, \partial Z - \int q(Z) \ln q(Z)\, \partial Z \\
&= \mathop{\mathrm{E}}_{Z \sim q} [\ln p_\theta(X, Z)] + \mathrm{H}(q)\,,
\end{aligned}$$

where the last equality follows using the definitions of the expectation and of the entropy (see Chapter 4.4.2). □

This proof also follows Chapter 2.3 in [Bishop, 2006].

Proof of Theorem 5.1. In the following, we consider a single iteration of Algorithm 5.1 and show that the parameter updates do not decrease the likelihood. Furthermore, we argue that the likelihood remains constant only if it already reached a stationary point. We denote the parameters before the considered round by θ^{old} and the updated parameters by θ^{new}.

Each round of the EM algorithm starts with the selection of the distribution $q = p_{X,\theta^{\text{old}}}$ over the hidden values Z, where

$$p_{X,\theta^{\text{old}}}(Z) = \Pr\left(Z \,\middle|\, X, \theta^{\text{old}}\right)$$

as specified in Algorithm 5.1. Based on this distribution, we define

$$f_{X,\theta^{\text{old}}}(\theta) := \mathop{\mathrm{E}}_{Z \sim p_{X,\theta^{\text{old}}}} [\ln p_\theta(X, Z)] + \mathrm{H}\left(p_{X,\theta^{\text{old}}}\right).$$

Then, Corollary 5.3 yields

$$\ln p_\theta(X) \geqslant f_{X,\theta^{\text{old}}}(\theta), \tag{23}$$

i. e., $f_{X,\theta^{\text{old}}}$ is a lower bound for $\ln p_\theta$. Furthermore,

$$\ln p_{\theta^{\text{old}}}(X) = f_{X,\theta^{\text{old}}}(\theta^{\text{old}}) \tag{24}$$

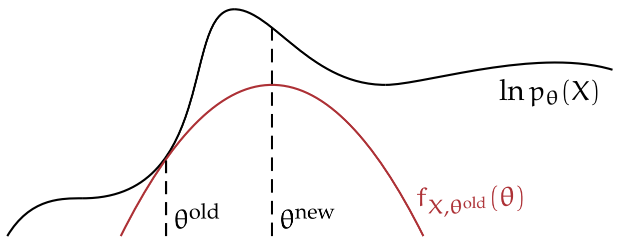


Figure 20: Optimization of $\ln p_\theta(X)$ over the choice of θ.

as sketched in Figure 20.

Since $H\left(p_{X,\theta^{\text{old}}}\right)$ is independent of θ, in order to maximize $f_{X,\theta^{\text{old}}}(\theta)$ over the choice of θ, it is sufficient to maximize $E_{Z\sim p_{X,\theta^{\text{old}}}}[\ln p_\theta(X,Z)]$. That is, the maximization step of Algorithm 5.1 computes

$$\theta^{\text{new}} := \arg\max_\theta f_{X,\theta^{\text{old}}}(\theta)$$

as shown in Figure 20. In particular,

$$f_{X,\theta^{\text{old}}}(\theta^{\text{new}}) \geqslant f_{X,\theta^{\text{old}}}(\theta^{\text{old}}) = \ln p_{\theta^{\text{old}}}(X),$$

where the equality at the end follows from Equation (24). Using Inequality (23) with $\theta = \theta^{\text{new}}$,

$$\ln p_{\theta^{\text{new}}}(X) \geqslant \ln p_{\theta^{\text{old}}}(X).$$

It remains to show that

$$\ln p_{\theta^{\text{new}}}(X) > \ln p_{\theta^{\text{old}}}(X)$$

if θ^{old} is not a stationary point of $\ln p_\theta(X)$ as a function of θ. Since the gradient of $g(\theta) := \ln p_\theta(X)$ is zero if and only if θ is a stationary point of g,

$$\frac{\partial g(\theta)}{\partial \theta}(\theta^{\text{old}}) \neq 0.$$

However, from Inequality (23) and Equation (24) it follows that g touches $f_{X,\theta^{\text{old}}}$ at the point θ^{old} as shown in Figure 20. Therefore, for the gradient of $f_{X,\theta^{\text{old}}}$ at the point θ^{old} we get

$$\frac{\partial f_{X,\theta^{\text{old}}}(\theta)}{\partial \theta}(\theta^{\text{old}}) = \frac{\partial g(\theta)}{\partial \theta}(\theta^{\text{old}}) \neq 0.$$

It follows that θ^{old} does not maximize $f_{X,\theta^{\text{old}}}$ and thus

$$f_{X,\theta^{\text{old}}}(\theta^{\text{new}}) > f_{X,\theta^{\text{old}}}(\theta^{\text{old}}).$$

Analogously to the derivation of non-decrease of the likelihood, we deduce

$$\ln p_{\theta^{\text{new}}}(X) > \ln p_{\theta^{\text{old}}}(X).$$

□

5.2 THE EM ALGORITHM FOR MIXTURE DISTRIBUTIONS

In this section, we discuss the application of the EM algorithm to general mixture distributions as introduced in Chapter 4.2. Let

$$\theta = (w_1, \dots, w_k, \theta_1, \dots, \theta_k)$$

be the parameter vector of the mixture, i. e., $w_1, \dots, w_k \in \mathbb{R}^+$ are the weights and $\theta_1, \dots, \theta_k$ are the parameter vectors of the component distributions. Then, the probability of a single observation $X = x$ is

$$p_\theta(x) = \sum_{j=1}^{k} w_j p_{\theta_j}(x).$$

In order to apply the EM algorithm, we have to decide on the hidden values Z. That is, we have to carefully choose which unobserved information would help us to solve the parameter estimation problem. Recall that drawing a single observation from the mixture can be described as a two step process. First, draw the j-th component with probability w_j, and second, draw x with probability $p_{\theta_j}(x)$. Based on this interpretation, we define $Z := (z_j) \in \{0,1\}^k$ with $\sum_{j=1}^{k} z_j = 1$ to be the vector of binary random variables indicating which component was chosen. Then, for $j = 1, \dots, k$ we get $\Pr(z_j = 1 \mid \theta) = w_j$.

The probability of an observation $X = (x_1, \dots, x_n)$ consisting of $n \in \mathbb{N}$ independent draws from the mixture distribution is

$$p_\theta(x_1, \dots, x_n) = \prod_{i=1}^{n} \sum_{j=1}^{k} w_j p(x_i | \theta_j).$$

We define $Z := (z_{ij}) \in \{0,1\}^{n \times k}$ with $\sum_{j=1}^{k} z_{ij} = 1$ for $i = 1, \dots, n$ to be the binary matrix indicating which component is responsible for each sub-observation. In the mixture case, this indicator matrix Z is the natural choice for the hidden values.

For $i = 1, \dots, n$ and $j = 1, \dots, k$ we get $\Pr(z_{ij} = 1 \mid \theta) = w_j$. If we denote the i-th row of the indicator matrix by z_i, then the probability to observe a particular row z_i can be written as

$$p_\theta(z_i) = \sum_{j=1}^{k} z_{ij} w_j.$$

Note that all but one summand of this sum are zero.

Using this indicator matrix Z, the log-likelihood of θ given the complete data (X, Z) is given by

$$\begin{aligned} \ln \Pr(X, Z \mid \theta) &= \sum_{i=1}^{n} \ln p_\theta(x_i, z_i) = \sum_{i=1}^{n} \ln \left(p_{z_i,\theta}(x_i) p_\theta(z_i)\right) \\ &= \sum_{i=1}^{n} \ln p_{z_i,\theta}(x_i) + \sum_{i=1}^{n} \ln p_\theta(z_i). \end{aligned}$$

Note that this decomposition is very useful since the two sums depend on disjoint parts of the parameter vector θ. The first sum can be written as

Here, all but one summand of the inner sum are zero.

$$\sum_{i=1}^{n} \ln p_{z_i,\theta}(x_i) = \sum_{i=1}^{n} \ln \sum_{j=1}^{k} z_{ij} p_{\theta_j}(x_i),$$

while the second sum can be written as

$$\sum_{i=1}^{n} \ln p_{\theta}(z_i) = \sum_{i=1}^{n} \ln \sum_{j=1}^{k} z_{ij} w_j.$$

That is, the first sum only depends on $\theta_1, \ldots, \theta_k$ and the second sum only depends on $w_1, \ldots, w_k$. With the linearity of the expectation, it follows that the maximization step of the EM algorithm can be considered as two separate maximizations.

Obviously, without specifying the type of the component distributions, nothing can be said about the updates of the EM algorithm for the parameter vectors $\theta_1, \ldots, \theta_k$. However, there do exist closed formulas for the updates of the weights $w_1, \ldots, w_k$. Given an observation X and a parameter vector θ, we define the *responsibility* r_j of the j-th component to be the expected number of observations drawn from the j-th component distribution, i. e.,

$$r_j := \sum_{i=1}^{n} \mathop{\mathrm{E}}_{Z \sim p_{X,\theta}} \left[z_{ij} \right] = \sum_{i=1}^{n} \Pr\left(z_{ij} = 1 \,\middle|\, X, \theta \right).$$

By applying Bayes' theorem twice, the responsibilities can be expressed as

$$\begin{aligned} r_j &= \sum_{i=1}^{n} \frac{\Pr(x_i, z_{ij} = 1 \mid \theta)}{\Pr(x_i \mid \theta)} \\ &= \sum_{i=1}^{n} \frac{\Pr(z_{ij} = 1 \mid \theta) \Pr(x_i \mid z_{ij} = 1, \theta)}{\Pr(x_i \mid \theta)} \\ &= \sum_{i=1}^{n} \frac{w_j p_{\theta_j}(x_i)}{\sum_{\ell=1}^{k} w_\ell p_{\theta_\ell}(x_i)}. \end{aligned}$$

That is, the responsibilities are computable by evaluating the component distributions and using the current parameters θ only. The EM updates for the weights are the normalized responsibilities, i. e., for $j = 1, \ldots, k$,

For a derivation of these updates see [Bilmes, 1997].

$$w_j^{\text{new}} := \frac{r_j}{n}.$$

5.3 THE EM ALGORITHM FOR GAUSSIAN MIXTURES

We finish our introduction to the EM algorithm with an application to Gaussian mixture models (cf. Chapter 4.3). Let

$$\theta = (w_1, \ldots, w_k, \mu_1, \ldots, \mu_k, \Sigma_1, \ldots, \Sigma_k)$$

be the parameter vector of the Gaussian mixture, i. e., $w_1, \dots, w_k \in \mathbb{R}^+$ are the weights, $\mu_1, \dots, \mu_k \in \mathbb{R}^d$ are the means of the component Gaussians and $\Sigma_1, \dots, \Sigma_k \in \mathbb{R}^{d \times d}$ are the corresponding covariance matrices. As discussed in Section 5.2, the EM updates of the weights are given by

$$\begin{aligned} w_j^{\text{new}} := \frac{r_j}{n} &= \frac{1}{n} \sum_{i=1}^{n} \Pr\big(z_{ij} = 1 \,\big|\, X, \theta\big) \\ &= \frac{1}{n} \sum_{i=1}^{n} \frac{w_j \mathcal{N}_{\mu_j, \Sigma_j}(x_i)}{\sum_{\ell=1}^{k} w_\ell \mathcal{N}_{\mu_\ell, \Sigma_\ell}(x_i)} \end{aligned}$$

for $j = 1, \dots, k$, where r_j is the responsibility of the j-th component as defined in Section 5.2.

The derivation of the update equations for the means and the covariance matrices is rather elaborate and can for example be looked up in [Bilmes, 1997]. For the j-th mean, the update equation is

$$\begin{aligned} \mu_j^{\text{new}} &:= \frac{\sum_{i=1}^{n} \Pr\big(z_{ij} = 1 \,\big|\, X, \theta\big)\, x_i}{\sum_{i=1}^{n} \Pr\big(z_{ij} = 1 \,\big|\, X, \theta\big)} \\ &= \frac{1}{r_j} \sum_{i=1}^{n} \frac{w_j \mathcal{N}_{\mu_j, \Sigma_j}(x_i)\, x_i}{\sum_{\ell=1}^{k} w_\ell \mathcal{N}_{\mu_\ell, \Sigma_\ell}(x_i)}. \end{aligned}$$

That is, the contribution of each point to the j-th component mean is proportional to its probability in the j-th component in relation to its overall probability. Finally, for the j-th covariance, the update equation is

$$\begin{aligned} \Sigma_j^{\text{new}} &:= \frac{\sum_{i=1}^{n} \Pr\big(z_{ij} = 1 \,\big|\, X, \theta\big)\, (x_i - \mu_j^{\text{new}})(x_i - \mu_j^{\text{new}})^T}{\sum_{i=1}^{n} \Pr\big(z_{ij} = 1 \,\big|\, X, \theta\big)} \\ &= \frac{1}{r_j} \sum_{i=1}^{n} \frac{w_j \mathcal{N}_{\mu_j, \Sigma_j}(x_i)\, (x_i - \mu_j^{\text{new}})(x_i - \mu_j^{\text{new}})^T}{\sum_{\ell=1}^{k} w_\ell \mathcal{N}_{\mu_\ell, \Sigma_\ell}(x_i)}. \end{aligned}$$

6 THE STOCHASTIC EM ALGORITHM

In this chapter, we study a probabilistic variant of the EM algorithm that was originally proposed as the Stochastic EM or SEM algorithm [Celeux and Diebolt, 1985]. The algorithm replaces the maximization step of the EM algorithm with two simple steps. First, it guesses the hidden values by sampling a fixed instance. Second, it maximizes the complete-data likelihood over the choice of the parameter vector. Depending on the application, this simplification yields a considerable speedup. Furthermore, there exist applications where the maximization step of the classical EM algorithm is computationally intractable (e. g., when it involves high dimensional integrations). In these cases, the simplified maximization step may lead to practicable update equations. However, the focus of this chapter lies on a comparison between the computations of the classical EM algorithm and its probabilistic variant. We discuss only statistical models for which both algorithms are applicable.

The models generated by the Stochastic EM algorithm are studied in [Ip, 1994]. For mixtures of distributions from the exponential family, the author shows that the sequence of models generated by the SEM iterations is an ergodic Markov chain converging weakly to a stationary distribution over models. Furthermore, the mean of this stationary distribution is analyzed under appropriate assumptions. It is shown that for the number of input points going to infinity, this mean converges to the maximum likelihood estimate. However, the mean of the stationary distribution usually can not be obtained by a single run of the algorithm. Instead, a large number of restarts is necessary to retrieve a reasonable approximation of the mean distribution.

Practitioners are often satisfied with the solutions computed by the EM algorithm. Therefore, we analyze the SEM algorithm in relation to the classical EM algorithm. In contrast to the previous analysis of the SEM algorithm that focused on stochastic properties of the above mentioned Markov chain, we study a single run of the algorithm. In our approach, we look at it as a probabilistic algorithm that tries to imitate the behavior of a deterministic algorithm while using its randomness to speed up the computations. Our results suggest that in most cases where the classical EM algorithm is applicable, one

Algorithm 6.1: Probabilistic EM algorithm

```
Input: incomplete observation $X$, initial model $\theta$

while <condition> do
1       $q(Z) \longleftarrow \Pr(Z \mid X, \theta)$            // for hidden r.v. Z
2a      draw $z \propto q(Z = z)$
2b      maximize $\Pr(X, Z = z \mid \theta)$ w. r. t. $\theta$
end
return $\theta$
```

can use the SEM algorithm to obtain similar results more efficiently. Moreover, in our experiments discussed in Chapter 7, we observe that sometimes, compared with the EM algorithm, the SEM algorithm computes parameters with a larger likelihood. This is most likely due to its inherent randomness that allows the algorithm to escape from a saddle point or an undesired local maximum of the likelihood function (cf. Chapter 5.1). Note that the algorithm is hypothetically able to even guess the 'true' hidden values, although this generally happens only with negligible probability.

In Section 6.1, we present the generic SEM algorithm. Afterwards, in Section 6.2, we show that general mixture distributions are suitable candidates for the application of the SEM algorithm. In Section 6.3, we compare the updates of the EM and SEM algorithm for Gaussian mixture models. We consider a single update step and show that with high probability the algorithms perform almost the same computations for sufficiently large input data sets. Our experimental results (see Chapter 7.4) confirm that this still holds for a large number of successive steps. Moreover, we show that for Gaussian mixtures, the simplified maximization step of the SEM algorithm leads to considerably better running times compared to the classical EM algorithm. This result is confirmed by our experiments as well.

6.1 THE GENERIC SEM ALGORITHM

The SEM algorithm is stated as Algorithm 6.1. The algorithm differs from the classical EM algorithm in the second step. It replaces the maximization in Step 2 of Algorithm 5.1 with two alternate steps (cf. Step 2a and Step 2b of Algorithm 6.1). Instead of operating with the expectation over the choice of Z, it simply guesses the hidden values. This is done in Step 2a by sampling an instance for Z using the distribution q from Step 1. Afterwards, the complete-data likelihood given (X, Z) is maximized over the choice of θ in Step 2b.

Algorithm 6.2: SEM algorithm for mixtures

```
Input: incomplete observation $X = (x_1, \dots, x_n)$,
       initial model $\theta = (w_1, \dots, w_k, \theta_1, \dots, \theta_k)$

while <condition> do
1    $q(Z) \longleftarrow \Pr(Z \mid X, \theta)$          // for indicator matrix Z
2a   draw $Z = (z_{ij}) \propto q(Z)$
2b   for j=1,...,k do
         $w_j^{\text{new}} \longleftarrow \frac{1}{n} \sum_{i=1}^{n} z_{ij}$
         $\theta_j^{\text{new}} \longleftarrow \arg\max_{\theta_j} \Pr(Y_j \mid \theta_j)$
             where $Y_j$ is a vector of all $x_i$ with $z_{ij} = 1$
     end
end
return $\theta$
```

6.2 the sem algorithm for mixture distributions

A crucial question concerning the SEM algorithm is for which class of models it should be applied. In this section we show that mixture distributions are suitable candidates. There is only one necessary precondition, namely that for each component distribution separately, the maximum likelihood estimate is computationally tractable.

The resulting SEM algorithm for mixture distributions is stated as Algorithm 6.2. As for the EM algorithm, we denote the incomplete observation by $X = (x_1, \dots, x_n)$ and the model parameters by

$$\theta = (w_1, \dots, w_k, \theta_1, \dots, \theta_k).$$

Again, we choose the indicator matrix $(z_{ij}) \in \{0,1\}^{n \times k}$ as the hidden values Z. Just as in case of the EM algorithm, the maximization step can be carried out separately for the weights $w_1, \dots, w_k$ and for the component parameters $\theta_1, \dots, \theta_k$. To do this, we define $n_1, \dots, n_k$ to be the number of data elements that are assigned to each component in Step 2a, i. e.,

cf. Chapter 5.2

$$n_j = \sum_{i=1}^{n} z_{ij} \quad \text{and} \quad \sum_{j=1}^{k} n_j = n.$$

Then, for the weight updates we simply compute the fraction of the input data that is assigned to each component, i. e.,

In Section 6.3.2, we show that these updates are good approximations of the EM updates.

$$w_j^{\text{new}} := \frac{n_j}{n}.$$

Furthermore, by $Y_j \in \mathbb{X}^{n_j}$ we denote the vector of elements x_i for $i = 1, \dots, n$ with $z_{ij} = 1$. Then, for the updates of $\theta_1, \dots, \theta_k$, we maximize the probability $p_{\theta_j}(Y_j)$ over the choice of θ_j for $j = 1, \dots, k$.

6.2.1 CORRECTNESS

In the following, we show that Algorithm 6.2 is a correct instantiation of Algorithm 6.1 for the mixture case. That is, we show that computing

$$w_j^{\text{new}} := \frac{1}{n}\sum_{i=1}^{n} z_{ij} \quad \text{and} \quad \theta_j^{\text{new}} := \arg\max_{\theta_j} \Pr(Y_j \mid \theta_j)$$

does indeed maximize $\Pr(X, Z = z \mid \theta)$. To prove the correctness of the weight updates, we use the following lemma. Note that for $k \in \mathbb{N}$, we denote the standard $(k-1)$-simplex by

$$S^{k-1} = \left\{ (p_1, \dots, p_k) \in \mathbb{R}_+^k \;\middle|\; \sum_{j=1}^{k} p_j = 1 \right\}.$$

Lemma 6.1. *Let $k, n, n_1, \dots, n_k \in \mathbb{N}$ with $\sum_{j=1}^{k} n_j = n$. Then, the function $f : S^{k-1} \to \mathbb{R}$ with*

$$f(p) = f(p_1, \dots, p_k) = \sum_{j=1}^{k} n_j \ln(p_j)$$

is maximized by $p = (\frac{n_1}{n}, \dots, \frac{n_k}{n})$.

Proof. Let $q = (q_1, \dots, q_k)$ with $q_j = \frac{n_j}{n}$ for $j = 1, \dots, k$. Instead of f, we maximize

$$\begin{aligned} \frac{f(p)}{n} + H(q) &= \sum_{j=1}^{k} q_j \ln p_j - \sum_{j=1}^{k} q_j \ln q_j \\ &= \sum_{j=1}^{k} q_j \ln \frac{p_j}{q_j} = -\mathrm{KL}(q \,\|\, p) \end{aligned}$$

where $H(\cdot)$ denotes the entropy and $\mathrm{KL}(\cdot \,\|\, \cdot)$ denotes the Kullback-Leibler divergence (see Chapter 4.4.2). Then, the lemma follows from the non-negativity of the Kullback-Leibler divergence and the fact that $\mathrm{KL}(q \,\|\, p) = 0$ if and only if $p = q$. □

Proposition 6.2. *Let $\theta = (w_1, \dots, w_k, \theta_1, \dots, \theta_k)$ be the parameter vector of a mixture model. Then, for any incomplete observation $X = (x_1, \dots, x_n)$ with the hidden indicator matrix $Z = (z_{ij})$, the complete-data likelihood $\mathcal{L}(X, Z \mid \theta)$ is maximized over the choice of θ by setting*

$$w_j = \frac{1}{n}\sum_{i=1}^{n} z_{ij} \quad \textit{and} \quad \theta_j = \arg\max_{\theta'_j} \Pr(Y_j \mid \theta'_j)$$

for $j = 1, \dots, k$, where Y_j is a vector of all x_i with $z_{ij} = 1$.

Proof. We denote the new model computed by Algorithm 6.2 in some fixed round by $\theta^{X,Z}$. Since

$$\Pr(X, Z \mid \theta) = \Pr(Z \mid \theta) \Pr(X \mid Z, \theta),$$

it is sufficient to show that $\theta^{X,Z}$ separately maximizes $\Pr(Z \mid \theta)$ as well as $\Pr(X \mid Z, \theta)$.

Recall that the i-th row of the indicator matrix (z_{ij}) is a binary unit vector indicating which component is responsible for the point x_i. That is, $z_{ij} \in \{0, 1\}$ with $\sum_{j=1}^{k} z_{ij} = 1$ and

$$\Pr\big(z_{ij} = 1 \,\big|\, \theta\big) = w_j = \sum_{j=1}^{k} z_{ij} w_j .$$

We conclude

$$\Pr(Z \mid \theta) = \prod_{i=1}^{n} \sum_{j=1}^{k} z_{ij} w_j = \prod_{j=1}^{k} w_j^{n_j} .$$

Then,

$$\ln \Pr(Z \mid \theta) = \sum_{j=1}^{k} n_j \ln(w_j)$$

and applying Lemma 6.1 yields that $\theta^{X,Z}$ maximizes $\Pr(Z \mid \theta)$.

It remains to show that $\theta^{X,Z}$ maximizes $\Pr(X \mid Z, \theta)$. However, due to the independence of the observations,

$$\begin{aligned} \ln \Pr(X \mid Z, \theta) &= \sum_{i=1}^{n} \ln \Pr(x_i \mid Z, \theta) \\ &= \sum_{i=1}^{n} \sum_{j=1}^{k} z_{ij} \ln \Pr\big(x_i \,\big|\, \theta_j\big) \\ &= \sum_{j=1}^{k} \ln \Pr\big(Y_j \,\big|\, \theta_j\big) . \end{aligned}$$

□

6.2.2 LIMITATIONS

Strictly speaking, for the maximization step to be well defined, we have to assure that $n_j \geqslant \zeta$ for $j = 1, \dots, k$ for some model dependent threshold $\zeta \in \mathbb{N}$. That is, at least ζ observations have to be assigned to each component distribution. Otherwise, it might not be possible to determine the new parameters. This is obvious for $n_j = 0$ at least, since generally, without any data, there is no basis for parameter estimation. For Gaussian mixture models, we discuss the threshold ζ in Section 6.3.

If $n_j < \zeta$, it might be a good idea to drop the under-determined component and to replace it. This can be done by drawing new component parameters from an a-priori distribution, for example.

6.3 THE SEM ALGORITHM FOR GAUSSIAN MIXTURES

In this section we show that for mixtures of Gaussians, the EM algorithm and the SEM algorithm compute almost the same parameter updates. More precisely, we consider multivariate Gaussian mixtures over the d-dimensional vector space $\mathbb{R}^d$.

Recall from Chapter 4.3 that for a mixture of $k \in \mathbb{N}$ multivariate Gaussians over $\mathbb{R}^d$ the parameters of each component consists of a mean $\mu \in \mathbb{R}^d$ and a covariance matrix $\Sigma \in \mathbb{R}^{d\times d}$. That is, the parameter vector of the model takes the form

$$\theta = (w_1, \dots, w_k, \mu_1, \dots, \mu_k, \Sigma_1, \dots, \Sigma_k).$$

As usual, we denote the incomplete observation by $X = (x_1, \dots, x_n)$ with $x_i \in \mathbb{R}^d$ and choose the hidden values Z to be the indicator matrix $(z_{ij}) \in \{0,1\}^{n\times k}$. Furthermore, based on the distribution q derived in Step 1 of Algorithm 6.2, we denote the probability that x_i was generated by the j-th component by

The expectation of a binary random variable is simply the probability that it becomes 1.

$$p_{ij} := \Pr(z_{ij} = 1 \mid X, \theta) = \mathop{\mathrm{E}}_{Z\sim q}\left[z_{ij}\right]. \tag{25}$$

Then, as discussed in Chapter 5.3, the classical EM algorithm in each step deterministically computes the following updates for $j = 1, \dots, k$:

$$w_j^{EM} := \frac{1}{n}\sum_{i=1}^{n} p_{ij}, \qquad \mu_j^{EM} := \frac{\sum_{i=1}^{n} p_{ij} x_i}{\sum_{i=1}^{n} p_{ij}},$$
$$\Sigma_j^{EM} := \frac{\sum_{i=1}^{n} p_{ij}(x_i - \mu_j^{EM})(x_i - \mu_j^{EM})^T}{\sum_{i=1}^{n} p_{ij}}. \tag{26}$$

After sampling the indicator matrix $(z_{ij}) \in \{0,1\}^{n\times k}$, the Probabilistic EM algorithm computes the following updates for $j = 1, \dots, k$:

$$w_j^{SEM} := \frac{1}{n}\sum_{i=1}^{n} z_{ij}, \qquad \mu_k^{SEM} := \frac{\sum_{i=1}^{n} z_{ij} x_i}{\sum_{i=1}^{n} z_{ij}},$$
$$\Sigma_j^{SEM} := \frac{\sum_{i=1}^{n} z_{ij}(x_i - \mu_j^{SEM})(x_i - \mu_j^{SEM})^T}{\sum_{i=1}^{n} z_{ij}}, \tag{27}$$

where μ_j^{SEM} and Σ_j^{SEM} are the maximum likelihood solution for the parameter estimation problem of a single Gaussian distribution with respect to the points assigned to the j-th component (cf. Chapter 4.3.1). Note that by writing z_{ij} as

$$z_{ij} = \Pr(z_{ij} = 1 \mid Z) = \Pr(z_{ij} = 1 \mid X, \theta, Z),$$

the SEM update equations can be interpreted as to arise from the EM update equations by incorporating the additional knowledge of which component each point was generated by. In contrast to the EM algorithm, the parameters computed by the SEM algorithm are in fact random variables with respect to the random experiment of drawing the z_{ij}. This offers the alternate interpretation that the EM updates arise from the SEM updates by replacing the indicator matrix with its expectation (cf. Equation (25)).

6.3.1 LIMITATIONS

As already observed in Section 6.2.2, strictly speaking, we have to assure that $|Y_j| = n_j$ is large enough for each $j = 1, \ldots, k$. In case of multivariate Gaussians, we need that $|Y_j| \geqslant d+1$ at least. Otherwise, the covariance matrix Σ_j^{SEM} is not symmetric positive definite. Assuming that the observations $x_1, \ldots, x_n \in \mathbb{R}^d$ are given in general linear position, assigning at least $d+1$ points to each Gaussian is sufficient, though.

6.3.2 PROXIMITY OF THE UPDATE EQUATIONS

In this section, we provide the main results of this chapter. The theorems presented below provide bounds for the differences between the parameter updates as computed by the EM and SEM algorithm. As one would expect, it can not be shown that the differences between the mean updates are small if the corresponding weight updates differ considerably. The same holds for the differences between the covariance updates. Furthermore, it can not be shown that the differences between the covariance updates are small if the corresponding mean updates differ considerably. A difference only in one coordinate of the mean updates already leads to useless bounds for the corresponding entries of the covariance matrix. Therefore, we first provide the bound for the weight updates. Afterwards, we state the bound for the mean updates which depends on the differences between the weight updates. Finally, we close with the bound for the covariance updates. This bound depends on the differences of the weight updates as well as on the differences between the mean updates.

Before we state the theorems, we provide some preliminary definitions. For vectors $v \in \mathbb{R}^d$, we denote the ℓ-th coordinate of v by $(v)_\ell$. For matrices $M \in \mathbb{R}^{d \times d}$, we denote the (ℓ_1, ℓ_2)-th entry of M by $(M)_{\ell_1 \ell_2}$. Furthermore, we define the spread of the ℓ-th coordinate of the input data set by

$$\Delta_\ell := \max_i (x_i)_\ell - \min_i (x_i)_\ell. \tag{28}$$

As in Chapter 5.2, we define the overall responsibility of the j-th Gaussian component by

$$r_j := \sum_{i=1}^{n} p_{ij}.$$

In the following, we develop the quantities that will be used to measure the differences between the mean and covariance updates computed by the EM and SEM algorithm. For $j \in \{1, \ldots, k\}$, the difference between the computed means for the j-th component Gaussian is

$$\mu_j^{SEM} - \mu_j^{EM} = \frac{\sum_{i=1}^{n} z_{ij}(x_i - \mu_j^{EM})}{\sum_{i=1}^{n} z_{ij}}.$$

We ignore the normalization factor $\sum_{i=1}^{n} z_{ij}$ in the denominator. Then, each summand $z_{ij}(x_i - \mu_j^{EM})$ of the numerator is a random variable that corresponds to the contribution of x_i to the difference between μ_j^{SEM} and μ_j^{EM}. For $\ell \in \{1, \dots, d\}$, the variance in the ℓ-th coordinate of this contribution is

$$\mathrm{Var}\left(z_{ij}(x_i - \mu_j^{EM})_\ell\right) = p_{ij}(1 - p_{ij})\left(x_i - \mu_j^{EM}\right)_\ell^2.$$

Since the z_{ij} are independent random variables, the variance in the ℓ-th coordinate of the overall contribution of all points is

$$\mathrm{Var}\left(\sum_{i=1}^{n} z_{ij}(x_i - \mu_j^{EM})_\ell\right) = \sum_{i=1}^{n} p_{ij}(1 - p_{ij})\left(x_i - \mu_j^{EM}\right)_\ell^2.$$

In our analysis of the proximity of the SEM updates, we measure the difference in the ℓ-th coordinate of the j-th mean in terms of the corresponding standard deviation

$$\tau_{j\ell} := \sqrt{\sum_{i=1}^{n} p_{ij}(1 - p_{ij})\left(x_i - \mu_j^{EM}\right)_\ell^2}. \tag{29}$$

Analogously, for the difference in the (ℓ_1, ℓ_2)-th entry of the j-th covariance, we use

$$\rho_{j\ell_1\ell_2} := \sqrt{\sum_{i=1}^{n} p_{ij}(1 - p_{ij})\left(\Upsilon_{ij} - \Sigma_j^{EM}\right)_{\ell_1\ell_2}^2}, \tag{30}$$

where $\Upsilon_{ij} = (x_i - \mu_j^{EM})(x_i - \mu_j^{EM})^T$.

To bound the difference between the updates of the EM algorithm and the SEM algorithm, we state three separate theorems. The first one bounds the difference of the weight updates for a single component distribution. The theorem holds for Gaussian mixtures as well as for general mixture models (cf. Section 6.2 and Chapter 5.2).

Theorem 6.3 (Proximity of weights). *Let* $j \in \{1, \dots, k\}$, $\delta \in \mathbb{R}$ *with* $2e^{-r_j/3} \leqslant \delta \leqslant 1$, *and* $\lambda_w = \sqrt{\frac{3 \ln 2/\delta}{r_j}}$. *Then, with probability of at least* $1 - \delta$,

$$\left|w_j^{SEM} - w_j^{EM}\right| \leqslant \lambda_w w_j^{EM}.$$

The next theorem bounds the difference in a single coordinate of the mean updates for a single component. One can not expect good estimates for the means if the weights are not approximated well. Thus, in the second theorem we assume that the corresponding weight is already approximated with an accuracy of λ_w. The derived bound for the computed mean depends on λ_w by a factor of $\frac{1}{1-\lambda_w}$.

Theorem 6.4 (Proximity of means). *Let* $j \in \{1, \dots, k\}$ *and let* A *be the event that*

$$\left|w_j^{SEM} - w_j^{EM}\right| \leqslant \lambda_w w_j^{EM}, \tag{31}$$

where $0 < \lambda_w < 1$.

Then, for $\ell \in \{1, \dots, d\}$ *and* $0 < \delta < 1$, *the conditional probability of*

$$\left|\left(\mu_j^{SEM} - \mu_j^{EM}\right)_\ell\right| \leqslant \frac{\lambda_\mu}{1 - \lambda_w} \cdot \frac{\tau_{j\ell}}{r_j}$$

given the event A *is at least* $1 - \delta$ *for*

$$\lambda_\mu = \begin{cases} \sqrt{2e \ln 2/\delta} & \text{if } \frac{\tau_{j\ell}}{\Delta_\ell} \geqslant \frac{1}{e}\sqrt{2e \ln 2/\delta} \\ \frac{2\Delta_\ell}{\tau_{j\ell}} \ln 2/\delta & \text{otherwise.} \end{cases}$$

The third theorem is the equivalent of Theorem 6.4 for a single entry of the covariance matrix. Similar to Theorem 6.4, the derived bound depends on the weight accuracy λ_w. Additionally, we assume that the ℓ-th coordinate of the corresponding mean is already approximated with an accuracy λ_μ^ℓ. The bound splits into a sum of two terms. The first term is the analogon of the bound for the mean from Theorem 6.4. The second term is the error that arises from the accuracy of the mean estimates.

Theorem 6.5 (Proximity of covariances). *Let* $j \in \{1, \dots, k\}$ *and let* A *be the event that*

$$\left|w_j^{SEM} - w_j^{EM}\right| \leqslant \lambda_w w_j^{EM} \tag{32}$$

where $0 < \lambda_w < 1$. *Furthermore, let* $\ell_1, \ell_2 \in \{1, \dots, d\}$ *and let* B *be the event that for* $\ell = \ell_1, \ell_2$,

$$\left|\left(\mu_j^{SEM} - \mu_j^{EM}\right)_\ell\right| \leqslant \frac{\lambda_\mu^\ell}{1 - \lambda_w} \cdot \frac{\tau_{j\ell}}{r_j} \tag{33}$$

where $\lambda_\mu^\ell > 0$.

Then, for $0 < \delta < 1$, *the conditional probability of*

$$\left|\left(\Sigma_j^{SEM} - \Sigma_j^{EM}\right)_{\ell_1\ell_2}\right| \leqslant \frac{\lambda_\Sigma}{1 - \lambda_w} \cdot \frac{\rho_{j\ell_1\ell_2}}{r_j} + \frac{\lambda_\mu^{\ell_1}\lambda_\mu^{\ell_2}}{(1 - \lambda_w)^2} \cdot \frac{\tau_{j\ell_1}\tau_{j\ell_2}}{r_j^2}$$

given the events A *and* B *is at least* $1 - \delta$ *for*

$$\lambda_\Sigma = \begin{cases} \sqrt{2e \ln 2/\delta} & \text{if } \frac{\rho_{j\ell_1\ell_2}}{\Delta_{\ell_1}\Delta_{\ell_2}} \geqslant \frac{1}{e}\sqrt{2e \ln 2/\delta} \\ \frac{2\Delta_{\ell_1}\Delta_{\ell_2}}{\rho_{j\ell_1\ell_2}} \ln 2/\delta & \text{otherwise.} \end{cases}$$

To derive a result for the complete update, one has to combine all three theorems using the union bound. Note that all three bounds depend on the responsibilities r_j. As one would expect due to the law of large numbers, the accuracy of our bounds improves with growing r_j.

6.3.3 LIMITATIONS

Note that, generally speaking, it is not possible to approximate a mixture component with too small a weight. Thus, we cannot expect to derive proximity bounds for the SEM updates that do not use assumptions on the weights, which correspond to the responsibilities $r_j = n \cdot w_j^{EM}$. The bounds from Theorem 6.4 and Theorem 6.5 both depend on $\frac{1}{1-\lambda_w}$. That is, these bounds become arbitrarily large for λ_w near 1. Therefore, to get reasonable results, we need λ_w to be considerably small. For instance, to ensure $\lambda_w \leqslant \frac{1}{2}$, the requirements for λ_w in Theorem 6.3 yield $r_j \geqslant 12 \cdot \ln(2/\delta)$. In other words, given a data set of size n, all weights have to be at least $\frac{12 \cdot \ln(2/\delta)}{n}$.

6.3.4 ON CHERNOFF BOUNDS

In the following, we state and prove several Chernoff type bounds. Chernoff bounds are a standard tool from probability theory for the derivation of concentration results. Readers who are familiar with this technique may skip this section or at least the given proofs.

See Corollary 4.6 in [Mitzenmacher and Upfal, 2005].

We start with an elementary Chernoff bound. A proof can be found in [Mitzenmacher and Upfal, 2005], for example.

Lemma 6.6. *Let $X_1, \ldots, X_n$ be independent random variables over $\{0,1\}$ and let $Y = \sum_{i=1}^n X_i$. Then, for each $0 \leqslant \lambda \leqslant 1$,*

$$\Pr(|Y - E[Y]| \geqslant \lambda \cdot E[Y]) \leqslant 2e^{-E[Y]\frac{\lambda^2}{3}}.$$

As a corollary, we get the following lemma which bounds the deviation for a given probability of occurrence. This bound will be used for the difference between the weight updates later.

Lemma 6.7. *Let $X_1, \ldots, X_n$ be independent random variables in $\{0,1\}$ and let $Y = \sum_{i=1}^n X_i$. Then, for $2e^{-\frac{E[Y]}{3}} \leqslant \delta \leqslant 1$ and $\lambda = \sqrt{\frac{3 \ln 2/\delta}{E[Y]}}$,*

$$\Pr(|Y - E[Y]| \geqslant \lambda \cdot E[Y]) \leqslant \delta.$$

To bound the differences between the mean and between the covariance updates, we need a more elaborate Chernoff type bound.

This Lemma and its proof are based on [Levchenko, 2013].

Lemma 6.8. *Let $X_1, \ldots, X_n$ be independent, discrete random variables with $E[X_i] = 0$ and $|X_i| \leqslant C$ for some constant $C > 0$ and $i = 1, \ldots, n$. Furthermore, let $Y = \sum_{n=1}^N X_i$. Then, for any $\lambda \geqslant 0$,*

$$\Pr\left(|Y| \geqslant \lambda\sqrt{\mathrm{Var}(Y)}\right) \leqslant 2e^{\frac{-\lambda^2}{2e^a}}$$

where $a \geqslant 0$ such that $\lambda = \frac{ae^a\sqrt{\mathrm{Var}(Y)}}{C}$.

Analogously to the relation between Lemma 6.6 and Lemma 6.7, we use Lemma 6.8 to derive a bound for a given probability of occurrence.

Lemma 6.9. *Let* $X_1, \dots, X_n$ *be independent, discrete random variables with* $E[X_i] = 0$ *and* $|X_i| \leqslant C$ *for some constant* $C > 0$ *and* $i = 1, \dots, n$. *Furthermore, let* $Y = \sum_{n=1}^{N} X_i$ *and* $0 < \delta < 1$. *Then,*

$$\Pr\left(|Y| \geqslant \lambda \sqrt{\operatorname{Var}(Y)}\right) \leqslant \delta$$

for

$$\lambda = \begin{cases} \sqrt{2e \ln 2/\delta} & \text{if } \frac{\sqrt{\operatorname{Var}(Y)}}{C} \geqslant \frac{1}{e}\sqrt{2e\ln 2/\delta} \\ \frac{2C}{\sqrt{\operatorname{Var}(Y)}} \ln 2/\delta & \text{otherwise.} \end{cases}$$

In the remainder of this section, we prove Lemma 6.8 and Lemma 6.9.

Proof of Lemma 6.8. First, note that we will naturally assume $\operatorname{Var}(Y) = \sum_{i=1}^{n} \operatorname{Var}(X_i) > 0$. Then, for any $\lambda \geqslant 0$, there exists an $a \geqslant 0$ with $\lambda = \frac{ae^a\sqrt{\operatorname{Var}(Y)}}{C}$. Due to symmetry, we only prove

For $\operatorname{Var}(Y) = 0$, there is no need for Lemma 6.8.

$$\Pr\left(Y \geqslant \lambda\sqrt{\operatorname{Var}(Y)}\right) \leqslant e^{\frac{-\lambda^2}{2e^a}}.$$

By Markov's inequality, for any $t > 0$ we obtain

$$\Pr\left(Y \geqslant \lambda\sqrt{\operatorname{Var}(Y)}\right) = \Pr\left(e^{tY} \geqslant e^{t\lambda\sqrt{\operatorname{Var}(Y)}}\right) \leqslant \frac{E\left[e^{tY}\right]}{e^{t\lambda\sqrt{\operatorname{Var}(Y)}}}. \tag{34}$$

Let $x_{i1}, x_{i2}, \dots$ be the possible outcomes of the discrete random variable X_i. We denote the corresponding probabilities by

$$\pi_{ij} := \Pr(X_i = x_{ij}).$$

Then, by the definition of the expectation and the series expansion of the exponential function,

$$\begin{aligned} E\left[e^{tX_i}\right] &= \sum_j \pi_{ij} e^{tx_{ij}} \\ &= \sum_j \pi_{ij}\left(1 + tx_{ij} + \frac{1}{2!}(tx_{ij})^2 + \frac{1}{3!}(tx_{ij})^3 + \dots\right). \end{aligned}$$

Separating the first two summands of the series expansion yields

$$\begin{aligned} &E\left[e^{tX_i}\right] \\ &= \sum_j \pi_{ij} + t\sum_j \pi_{ij}x_{ij} + \sum_j \pi_{ij}\left(\frac{1}{2!}(tx_{ij})^2 + \frac{1}{3!}(tx_{ij})^3 + \dots\right) \\ &= 1 + \sum_j \pi_{ij}(tx_{ij})^2\left(\frac{1}{2!} + \frac{1}{3!}tx_{ij} + \frac{1}{4!}(tx_{ij})^2 + \dots\right), \end{aligned}$$

where the second equality follows from $t\sum_j \pi_{ij}x_{ij} = t\,E[X_i] = 0$. Since $\frac{2}{(m+2)!} \leqslant \frac{1}{m!}$ for $m \in \mathbb{N}$,

$$E\left[e^{tX_i}\right] \leqslant 1 + \sum_j \pi_{ij}(tx_{ij})^2\frac{1}{2}e^{tx_{ij}}.$$

In the following, we assume that t is chosen such that

$$a \geqslant tC. \tag{35}$$

Then, $e^{tx_{ij}} \leqslant e^a$ and it follows that

$$E\left[e^{tX_i}\right] \leqslant 1 + \frac{e^a}{2} t^2 \operatorname{Var}(X_i),$$

where we used

$$\operatorname{Var}(X_i) = E\left[X_i^2\right] - E[X_i]^2 = E\left[X_i^2\right] = \sum_j \pi_{ij} (tx_{ij})^2.$$

Since $X_1, \ldots, X_n$ are independent,

$$E\left[e^{tY}\right] = \prod_{i=1}^{n} E\left[e^{tX_i}\right] \leqslant \prod_{i=1}^{n} \left(1 + \frac{e^a}{2} t^2 \operatorname{Var}(X_i)\right).$$

Using $1 + \alpha \leqslant e^\alpha$ for $\alpha \geqslant 0$, it follows that

$$E\left[e^{tY}\right] \leqslant \prod_{i=1}^{n} e^{\frac{e^a}{2} t^2 \operatorname{Var}(X_i)} = e^{\frac{e^a}{2} t^2 \operatorname{Var}(Y)}. \tag{36}$$

Finally, we fix $t := \frac{\lambda}{e^a \sqrt{\operatorname{Var}(Y)}}$. Then, $tC = a$ and Inequality (35) is satisfied as required. Substituting t in Inequality (36) yields

$$E\left[e^{tY}\right] \leqslant e^{\frac{\lambda^2}{2e^a}}.$$

Furthermore,

$$e^{t\lambda\sqrt{\operatorname{Var}(Y)}} = e^{\frac{\lambda^2}{e^a}}.$$

Thus,

$$\frac{E\left[e^{tY}\right]}{e^{t\lambda\sqrt{\operatorname{Var}(Y)}}} \leqslant e^{\frac{-\lambda^2}{2e^a}}$$

and Inequality (34) yields the claim. □

Proof of Lemma 6.9. As in the proof of Lemma 6.8, we assume $\operatorname{Var}(Y) > 0$. First, let

$$\frac{\sqrt{\operatorname{Var}(Y)}}{C} \geqslant \frac{1}{e}\sqrt{2e \ln 2/\delta}$$

and

$$\lambda = \sqrt{2e \ln 2/\delta}. \tag{37}$$

Then, $\lambda \leqslant \frac{e\sqrt{\operatorname{Var}(Y)}}{C}$. Thus, we are able to choose $0 \leqslant a \leqslant 1$, such that $\lambda = \frac{a e^a \sqrt{\operatorname{Var}(Y)}}{C}$. Applying Lemma 6.8,

$$\Pr\left(|Y| \geqslant \lambda\sqrt{\operatorname{Var}(Y)}\right) \leqslant 2 \exp\left(\frac{-\lambda^2}{2e^a}\right).$$

By substituting Equation (37),

$$\Pr\left(|Y| \geqslant \lambda\sqrt{\operatorname{Var}(Y)}\right) \leqslant 2 \exp\left(\frac{-2e \ln 2/\delta}{2e^a}\right) = 2\left(\frac{\delta}{2}\right)^{\frac{e}{e^a}} \leqslant \delta.$$

It remains to consider the case where

$$\frac{\sqrt{\mathrm{Var}(Y)}}{C} < \frac{1}{e}\sqrt{2e\ln 2/\delta} \tag{38}$$

and

$$\lambda = \frac{2C}{\sqrt{\mathrm{Var}(Y)}} \ln 2/\delta. \tag{39}$$

Let $a \in \mathbb{R}$, such that $\lambda = \frac{ae^a\sqrt{\mathrm{Var}(Y)}}{C}$. Then,

$$ae^a = \frac{2C^2}{\mathrm{Var}(Y)} \ln 2/\delta. \tag{40}$$

By Inequality (38), $\frac{C}{\sqrt{\mathrm{Var}(Y)}} > \frac{e}{\sqrt{2e\ln 2/\delta}}$ and we deduce

$$ae^a > \frac{2e^2}{2e\ln 2/\delta} \ln 2/\delta = e.$$

It follows that $a > 1$. Applying Lemma 6.8,

$$\Pr\left(|Y| \geqslant \lambda\sqrt{\mathrm{Var}(Y)}\right) \leqslant 2\exp\left(\frac{-\lambda^2}{2e^a}\right).$$

By substituting Equation (39),

$$\Pr\left(|Y| \geqslant \lambda\sqrt{\mathrm{Var}(Y)}\right) \leqslant 2\exp\left(-\frac{4C^2}{\mathrm{Var}(Y)}(\ln 2/\delta)^2\frac{1}{2e^a}\right).$$

Using Equation (40),

$$\Pr\left(|Y| \geqslant \lambda\sqrt{\mathrm{Var}(Y)}\right) \leqslant 2\exp\left(-2ae^a\ln 2/\delta\frac{1}{2e^a}\right) = 2\exp\left(-a\ln 2/\delta\right).$$

Since $a > 1$,

$$\Pr\left(|Y| \geqslant \lambda\sqrt{\mathrm{Var}(Y)}\right) < 2\exp\left(-\ln 2/\delta\right) = 2\left(\frac{\delta}{2}\right) = \delta.$$

□

6.3.5 PROOF OF THE PROXIMITY BOUNDS

Using Lemma 6.7, we are able to prove Theorem 6.3.

Proof of Theorem 6.3. Let $W = \sum_{i=1}^{n} z_{ij} = n \cdot w_j^{SEM}$. Then,

$$\underset{Z\sim q}{\mathrm{E}}[W] = \sum_{i=1}^{n} p_{ij} = r_j = n \cdot w_j^{EM}.$$

Thus, applying Lemma 6.7 yields

$$n \cdot |w_j^{SEM} - w_j^{EM}| \geqslant n \cdot \lambda w_j^{EM}$$

with probability at most δ. □

Using Lemma 6.9, we are able to prove Theorem 6.4 and Theorem 6.5.

Proof of Theorem 6.4. Let $j \in \{1, \dots, k\}$, $\ell \in \{1, \dots, d\}$ and define the real random variable

$$M_{ij\ell} := (z_{ij} - p_{ij}) \left(x_i - \mu_j^{EM}\right)_\ell .$$

Since $E_{Z\sim q}\left[z_{ij}\right] = p_{ij}$, it follows that $E_{Z\sim q}\left[M_{ij\ell}\right] = 0$ and

$$\mathrm{Var}(M_{ij\ell}) = p_{ij}(1 - p_{ij})(x_i - \mu_j^{EM})_\ell^2.$$

Moreover, since each μ_j^{EM} is a convex combination of $x_1, \dots, x_n$,

$$|M_{ij\ell}| \leqslant |z_{ij} - p_{ij}| \cdot \left|\left(x_i - \mu_j^{EM}\right)_\ell\right| \leqslant \Delta_\ell.$$

Note that the definition of μ_j^{EM} yields $\sum_{i=1}^n p_{ij} \left(x_i - \mu_j^{EM}\right)_\ell = 0$. Thus, for the random variable $M_{j\ell} := \sum_{i=1}^n M_{ij\ell}$, we deduce

$$M_{j\ell} = \sum_{i=1}^n z_{ij} \left(x_i - \mu_j^{EM}\right)_\ell .$$

Furthermore, $E_{Z\sim q}\left[M_{j\ell}\right] = 0$ and

$$\mathrm{Var}(M_{j\ell}) = \sum_{i=1}^n p_{ij}(1 - p_{ij})(x_i - \mu_j^{EM})_\ell^2 = \tau_{j\ell}^2.$$

Note that the standard deviation of $M_{j\ell}$ is $\tau_{j\ell}$, which was introduced in Section 6.3.2 to measure the difference between the mean updates. Applying Lemma 6.9 with $C = \Delta_\ell$ and λ_μ as provided in the theorem yields

$$\Pr\left(\left|\sum_{i=1}^n z_{ij} \left(x_i - \mu_j^{EM}\right)_\ell\right| \geqslant \lambda_\mu \tau_{j\ell}\right) \leqslant \delta. \tag{41}$$

The difference between the coordinates of the mean updates can be written as

$$\begin{aligned}\left|\left(\mu_j^{SEM} - \mu_j^{EM}\right)_\ell\right| &= \left|\frac{\left(\sum_{i=1}^n z_{ij} x_i - \sum_{i=1}^n z_{ij} \mu_j^{EM}\right)_\ell}{\sum_{i=1}^n z_{ij}}\right| \\ &= \left|\frac{\sum_{i=1}^n z_{ij} \left(x_i - \mu_j^{EM}\right)_\ell}{n \cdot w_j^{SEM}}\right| .\end{aligned}$$

Then, by combining Inequality (41) and Inequality (31), we conclude that the conditional probability of

$$\left|\left(\mu_j^{SEM} - \mu_j^{EM}\right)_\ell\right| \leqslant \frac{\lambda_\mu \tau_{j\ell}}{n(1 - \lambda_w) w_j^{EM}} = \frac{\lambda_\mu \tau_{j\ell}}{(1 - \lambda_w) r_j}$$

given the event A is at least $1 - \delta$. $\square$

cf. Equation (26) *and Equation* (27)

Proof of Theorem 6.5. The covariance updates of the EM and SEM algorithm depend on the corresponding mean updates. Consequently, the difference between the covariance updates depends on the difference between the mean updates. To measure the impact of $\mu_j^{SEM} - \mu_j^{EM}$ on $\Sigma_j^{SEM} - \Sigma_j^{EM}$, let $\widetilde{\Sigma_j^{SEM}}$ be the covariance that would be computed by the SEM algorithm, if $\mu_j^{SEM} = \mu_j^{EM}$. That is,

$$\widetilde{\Sigma_j^{SEM}} := \frac{\sum_{i=1}^n z_{ij}(x_i - \mu_j^{EM})(x_i - \mu_j^{EM})^T}{\sum_{i=1}^n z_{ij}}.$$

By substituting $x_i - \mu_j^{EM} = x_i - \mu_j^{SEM} + \mu_j^{SEM} - \mu_j^{EM}$, we deduce

$$\begin{aligned}\widetilde{\Sigma_j^{SEM}} =& \frac{\sum_{i=1}^n z_{ij}\left(x_i - \mu_j^{SEM}\right)\left(x_i - \mu_j^{SEM}\right)^T}{\sum_{i=1}^n z_{ij}} \\ &+ \frac{\sum_{i=1}^n z_{ij}\left(x_i - \mu_j^{SEM}\right)\left(\mu_j^{SEM} - \mu_j^{EM}\right)^T}{\sum_{i=1}^n z_{ij}} \\ &+ \frac{\sum_{i=1}^n z_{ij}\left(\mu_j^{SEM} - \mu_j^{EM}\right)\left(x_i - \mu_j^{SEM}\right)^T}{\sum_{i=1}^n z_{ij}} \\ &+ \frac{\sum_{i=1}^n z_{ij}\left(\mu_j^{SEM} - \mu_j^{EM}\right)\left(\mu_j^{SEM} - \mu_j^{EM}\right)^T}{\sum_{i=1}^n z_{ij}}.\end{aligned}$$

Note that the first summand is equal to Σ_j^{SEM} and the last summand is equal to $\left(\mu_j^{SEM} - \mu_j^{EM}\right)\left(\mu_j^{SEM} - \mu_j^{EM}\right)^T$. Furthermore, both summands in the middle are zero, since

$$\frac{\sum_{i=1}^n z_{ij}\left(x_i - \mu_j^{SEM}\right)}{\sum_{i=1}^n z_{ij}} = \frac{\sum_{i=1}^n z_{ij} x_i}{\sum_{i=1}^n z_{ij}} - \mu_j^{SEM} = 0.$$

By denoting $v := \mu_j^{SEM} - \mu_j^{EM}$, we conclude

$$\widetilde{\Sigma_j^{SEM}} = \Sigma_j^{SEM} + vv^T.$$

Analogously to the proof of Theorem 6.4, we bound the difference between the entries of $\widetilde{\Sigma_j^{SEM}}$ and Σ_j^{EM}. To do this, we define the real random variable

$$S_{ij\ell_1\ell_2} := (z_{ij} - p_{ij})\left(\Upsilon_{ij} - \Sigma_j^{EM}\right)_{\ell_1\ell_2}$$

where $\Upsilon_{ij} := (x_i - \mu_j^{EM})(x_i - \mu_j^{EM})^T$. Since $E_{Z\sim q}\left[z_{ij}\right] = p_{ij}$, it follows that $E_{Z\sim q}\left[S_{ij\ell_1\ell_2}\right] = 0$ and

$$\operatorname{Var}(S_{ij\ell_1\ell_2}) = p_{ij}(1 - p_{ij})\left(\Upsilon_{ij} - \Sigma_j^{EM}\right)^2_{\ell_1\ell_2}.$$

Moreover, since each Σ_j^{EM} is a convex combination of $\Upsilon_{1j}, \ldots, \Upsilon_{nj}$,

$$|S_{ij\ell_1\ell_2}| \leqslant |z_{ij} - p_{ij}| \cdot \left|\left(\Upsilon_{ij} - \Sigma_j^{EM}\right)_{\ell_1\ell_2}\right| \leqslant \Delta_{\ell_1}\Delta_{\ell_2}.$$

Note that the definition of Σ_j^{EM} yields $\sum_{i=1}^n p_{ij}\left(\Upsilon_{ij} - \Sigma_j^{EM}\right)_{\ell_1\ell_2} = 0$. Thus, for the random variable $S_{j\ell_1\ell_2} := \sum_{i=1}^n S_{ij\ell_1\ell_2}$, we deduce

$$S_{j\ell_1\ell_2} = \sum_{i=1}^n z_{ij}\left(\Upsilon_{ij} - \Sigma_j^{EM}\right)_{\ell_1\ell_2}.$$

Furthermore, $E_{Z\sim q}\left[S_{j\ell_1\ell_2}\right] = 0$ and

$$\mathrm{Var}(S_{j\ell_1\ell_2}) = \sum_{i=1}^n p_{ij}(1-p_{ij})\left(\Upsilon_{ij} - \Sigma_j^{EM}\right)^2_{\ell_1\ell_2} = \rho^2_{j\ell_1\ell_2}.$$

Note that the standard deviation of $S_{j\ell_1\ell_2}$ is $\rho_{j\ell_1\ell_2}$, which was introduced in Section 6.3.2 to measure the difference between the covariance updates. Applying Lemma 6.9 with $C = \Delta_{\ell_1}\Delta_{\ell_2}$ and λ_Σ as provided in the theorem yields

$$\Pr\left(\left|\sum_{i=1}^n z_{ij}\left(\Upsilon_{ij} - \Sigma_j^{EM}\right)_{\ell_1\ell_2}\right| \geqslant \lambda_\Sigma \rho_{j\ell_1\ell_2}\right) \leqslant \delta. \tag{42}$$

The difference between the entries of the covariance updates can be written as

$$\begin{aligned}
\left|\left(\Sigma_j^{SEM} - \Sigma_j^{EM}\right)_{\ell_1\ell_2}\right| &= \left|\left(\widetilde{\Sigma_j^{SEM}} - \nu\nu^T - \Sigma_j^{EM}\right)_{\ell_1\ell_2}\right| \\
&= \left|\left(\frac{\sum_{i=1}^n z_{ij}\Upsilon_{ij}}{\sum_{i=1}^n z_{ij}} - \nu\nu^T - \frac{\sum_{i=1}^n z_{ij}\Sigma_j^{EM}}{\sum_{i=1}^n z_{ij}}\right)_{\ell_1\ell_2}\right| \\
&= \left|\left(\frac{\sum_{i=1}^n z_{ij}\left(\Upsilon_{ij} - \Sigma_j^{EM}\right)}{n\cdot w_j^{SEM}} - \nu\nu^T\right)_{\ell_1\ell_2}\right|.
\end{aligned}$$

Using the triangle inequality,

$$\begin{aligned}
\left|\left(\Sigma_j^{SEM} - \Sigma_j^{EM}\right)_{\ell_1\ell_2}\right| \leqslant &\left|\left(\frac{\sum_{i=1}^n z_{ij}\left(\Upsilon_{ij} - \Sigma_j^{EM}\right)}{n\cdot w_j^{SEM}}\right)_{\ell_1\ell_2}\right| \\
&+ \left|\left(\mu_j^{SEM} - \mu_j^{EM}\right)_{\ell_1}\right| \cdot \left|\left(\mu_j^{SEM} - \mu_j^{EM}\right)_{\ell_2}\right|.
\end{aligned}$$

Then, by combining Inequality (42), Inequality (32) and Inequality (33), we conclude that the conditional probability of

$$\begin{aligned}
\left|\left(\Sigma_j^{SEM} - \Sigma_j^{EM}\right)_{\ell_1\ell_2}\right| &\leqslant \frac{\lambda_\Sigma \rho_{j\ell_1\ell_2}}{n(1-\lambda_w)w_j^{EM}} + \frac{\lambda_\mu^{\ell_1}\lambda_\mu^{\ell_2}}{(1-\lambda_w)^2} \cdot \frac{\tau_{j\ell_1}\tau_{j\ell_2}}{r_j^2} \\
&= \frac{\lambda_\Sigma}{(1-\lambda_w)} \cdot \frac{\rho_{j\ell_1\ell_2}}{r_j} + \frac{\lambda_\mu^{\ell_1}\lambda_\mu^{\ell_2}}{(1-\lambda_w)^2} \cdot \frac{\tau_{j\ell_1}\tau_{j\ell_2}}{r_j^2}.
\end{aligned}$$

given the events A and B is at least $1-\delta$. □

6.3.6 RUNNING TIME ANALYSIS

When the EM and SEM algorithm are both applicable for a particular statistical model, it is natural to ask why the SEM algorithm should be favored. Besides its ability to escape from a stationary point of the likelihood function as mentioned at the beginning of this chapter, the SEM algorithm is particularly faster in general. In the following, we explain where the speedup stems from. First, we motivate the speedup for general mixture models. Afterwards, we show a constant factor speedup for Gaussian mixture models.

Let $X = (x_1, \ldots, x_n)$ be the input data set and assume that it was generated by a general mixture of k distributions. The EM algorithm maximizes the expected complete data log-likelihood. Usually, this implies that each data point is involved in the update of all components (see for example Equation 26). Therefore, the running time of the EM algorithm depends on the term kn. In contrast, the SEM algorithm assigns each point to exactly one component. Hence, each point is involved in the update of a single component only (see for example Equation 27). This observation suggests that the speedup of the SEM algorithm might be up to a factor of k. However, for the sampling step (Step 2a), the SEM algorithm has to compute the probability that a certain point was generated by a certain component for each point and each component distribution. That is, the running time of the SEM algorithm depends on the term kn as well. Nevertheless, we are able to show that regarding the parameter estimation problem of Gaussian mixture models, there is a constant factor speedup. To do this, we discuss the number of computations in a single iteration of both algorithms.

For the sake of simplicity, we abstain from precisely charging all computations. Instead, we consider only the multiplications in $\mathbb{R}$. Furthermore, we consider only the leading term of the number of performed multiplications. That is, we focus on the computations that entail the most multiplications in terms of the parameters n, d and k for considerably large values of n, d and k. Not surprisingly, these are the computations involving the covariance matrices. Furthermore, we assume that $n \gg d$.

Both algorithms start with computing the probabilities $\Pr(x_i \mid \mu_j, \Sigma_j)$ for $i = 1, \ldots, n$ and $j = 1, \ldots, k$. The number of multiplications for the computation of these probabilities is dominated by matrix-vector multiplications between the $d \times d$ covariance matrices Σ_j and the d-dimensional vectors x_i. Therefore, the leading term of the number of multiplications for the common computations is knd^2. *cf. Equation* (25)

The leading term of the number of multiplications for the remaining computations of the EM algorithm is dominated by the update of the covariances. Each point is involved in the update of each component by the outer product of the point with itself. Therefore, the *cf. Equation* (26) leading term of the number of remaining multiplications is knd^2 as well. Thus, the overall number of multiplications of the EM algorithm is dominated by $2knd^2$.

In case of the SEM algorithm, again the updates of the covariances determine the leading term of the number of remaining multiplications. However, due to the hard assignment of the points, the leading term does not depend on k. Instead, the number of multiplications needed to update the covariance matrix is dominated by nd^2. It follows that for n, k and d large enough, the number of multiplications during the remaining SEM computations is negligible in comparison to the number of multiplications during the common computations. Thus, the overall running time of the SEM algorithm is dominated by knd^2.

cf. Equation (27)

This implies a factor 2 speedup. However, depending on details of the implementation, the actual dependency of the multiplications during the common computations may better be assumed to be dominated by c_1knd^2 for some constant c_1. Analogously, for the remaining computations of the EM algorithm we assume a leading term of the dependency of the multiplications of c_2knd^2 for some constant c_2. This results in a speedup of a factor of $1 + \frac{c_2}{c_1}$. Indeed, our experimental results confirm a constant factor speedup (see Chapter 7.4). Moreover, the measured speedup tends a factor of 3 which implies $c_2 > c_1$.

7

EXPERIMENTAL ANALYSIS

To underpin our proximity and running time analysis from Chapter 6, we performed a series of experiments. In our experimental analysis, we compare the computations of the classical EM algorithm and the probabilistic SEM algorithm on a number of different data sets. The data sets divide into several artificial and real world data sets. As in the theoretical analysis, we consider the parameter estimation problem for Gaussian mixture models only.

7.1 IMPLEMENTATION

To get comparable results, we implemented the EM algorithm and the SEM algorithm from scratch in C++. For the linear algebra computations, we use the free C++ template library Eigen (available at `http://eigen.tuxfamily.org/`). For the generation of artificial data sets, to compute initial model parameters, and for the probabilistic steps of the SEM algorithm, we need an efficient pseudo-random number generator. Our implementation uses the Mersenne twister [Matsumoto and Nishimura, 1998] specified as `std::mt19937` in the current C++ standard [ISO, 2012].

As discussed in Chapter 6.3.1, it is possible that the SEM algorithm assigns too few points to a particular component to be able to compute a new covariance. Although we disregard these cases in our analysis, we have to consider them in our implementation. If no point is assigned to a particular component, we solve the problem by sampling a new mean uniformly from the input data set and initializing the covariance matrix with a scaled identity matrix. More precisely, set the covariance matrix to $\sigma^2 I_d$, where we compute

$$\sigma^2 := \frac{1}{2d} \min_{i \neq j} \|\mu_i - \mu_j\|^2$$

as proposed in [Dasgupta and Schulman, 2007] for the initial parameter estimation. If not enough points are assigned to a particular component, we try to mix the under-determined covariance with the previous covariance matrix. If this fails to compute a symmetric positive definite matrix, we simply keep the old covariance matrix. Due to numerical instability, the EM algorithm has to deal with similar *cf. Chapter 5.3*

problems if the responsibility r_j of a component becomes too small. Therefore, we implemented a similar error handling procedure for the EM algorithm. In our experiments, these problems hardly ever occurred. However, it must be said that the error handling may lead to substantially different solutions of the EM and SEM algorithm. Our experimental results presented in Section 7.4 are not affected by the error handling procedures.

7.2 DATA SETS

For our experiments, we used artificial as well as real world data sets. For the generation of the artificial data sets, we considered different combinations of the dimension $d \in \mathbb{N}$ and the number of component distributions $k \in \mathbb{N}$. For each combination, we probabilistically computed several parameter vectors $\theta = (w_1, \ldots, w_k, \mu_1, \ldots, \mu_k, \Sigma_1, \ldots, \Sigma_k)$ for Gaussian mixtures with k components over $\mathbb{R}^d$.

cf. Chapter 4.3.2

For the determination of the weights $w_1, \ldots, w_k$ we drew k real numbers uniformly at random from the interval $[0, 1]$. To compute balanced as well as unbalanced weights, these numbers were afterwards exponentiated with the same integer constant chosen between 0 and 3. Finally, the resulting numbers were normalized to retrieve the weights.

For the determination of the means $\mu_1, \ldots, \mu_k$ we created a Gaussian meta distribution from which all means were drawn. The mean of the meta distribution was set to the origin and the covariance matrix of the meta distribution was created probabilistically itself. To do this, we first created a matrix M by drawing each entry from a one-dimensional Gaussian distribution with mean 0 and variance $\frac{k}{d}$. Afterwards, to retrieve the covariance matrix of the meta distribution, we computed the product of the matrix M with its own transpose. This generally results in a symmetric positive-definite covariance matrix. The variance $\frac{k}{d}$ of the one-dimensional Gaussian used to determine the entries of the matrix M was chosen to control the separation of the created means $\mu_1, \ldots, \mu_k$. The separation increases with increasing k (more components need more space) and it decreases with increasing d (more dimensions allow less separation in each of them). Proceeding this way, we ensured that the mixtures mainly consist of interfusing Gaussians. To get reasonable results it is important that the points generated by different components of a Gaussian mixture are not pairwise well separated. Otherwise, the task of learning the parameters is too easy. Furthermore, the difference in the computations of the EM and SEM algorithm would become too small to be verified in our experiments.

The distance-based clustering approach discussed in Part I, is a suitable method, for example.

Finally, to determine the covariance matrices $\Sigma_1, \ldots, \Sigma_k$, we proceeded as for the covariance matrix of the meta distribution used for the computation of the means. The only difference is that we use a

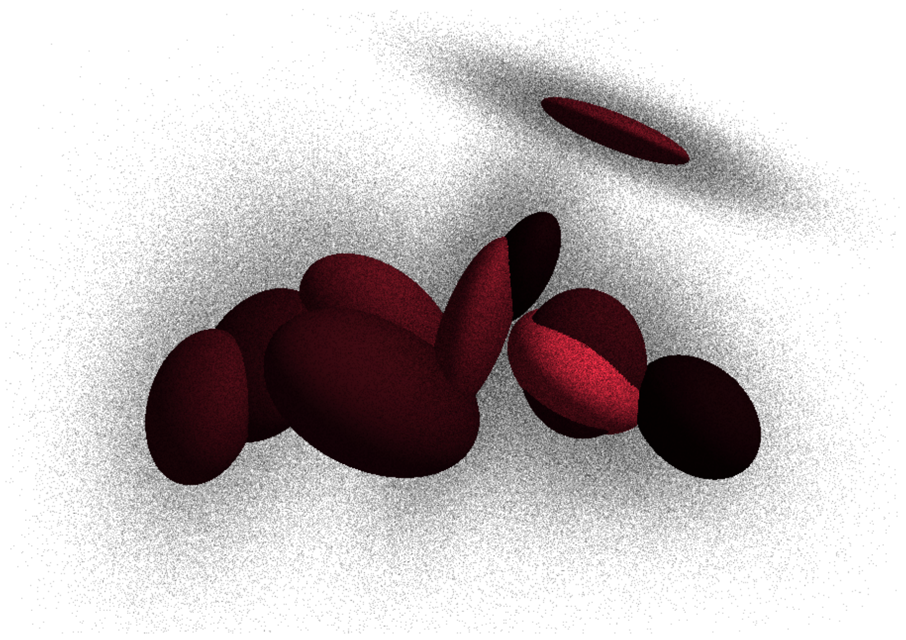


Figure 21: Projection of one of the artificial data sets with $n = 1\,000\,000$. The ellipsoids depict the standard deviation areas of each component distribution. The larger the weight of the component, the brighter is the shade of the corresponding ellipsoid.

one-dimensional Gaussian with variance 1 to draw the entries of the matrix M.

For each θ computed as above, we drew several point sets of different size from the corresponding Gaussian mixture. Our experiments for the different combinations of d and k led to essentially similar results. Therefore, in the following we discuss artificial data sets with $d = k = 10$ only. To confirm our proximity results from Chapter 6, we created several data sets of size $n = 1\,000\,000$ (cf. Figure 21). To investigate the behavior for small data sets, we also created data sets of size $n = 10\,000$. In the remainder of this section, we consider two particular data sets that led to characteristic results, one for $n = 1\,000\,000$ which we denote by *Art1M* and one for $n = 10\,000$ which we denote by *Art10K*. For both artificial data sets we computed Gaussian mixtures with the proper number of component distributions, i. e., with $k = 10$.

As real world data we use three publicly available data sets shown in Figure 7.2. The first one is the *Forest Covertype data set* which is part of the *UCI Machine Learning Library* [Asuncion and Newman, 2007] and contains 581 012 data points. The data set contains cartographic information about four wilderness areas located in the Roosevelt National Forest. Each data point consists of cartographic features of an area of 900 square meters in size. To get data suitable for Gaussian mixture parameter estimation, we used only the quantitative attributes located at the first 10 coordinates (see Figure 22a). That is, we ignored the labels describing which tree species dominates the respective area and 44 qualitative binary attributes. For the Forest Covertype data set we computed Gaussian mixtures with $k = 10$ component distributions.

The second data set is based on the *Amsterdam Library of Object Images (ALOI)* [Geusebroek et al., 2005]. This database consists of 110 250 images of 1 000 small objects, taken under various conditions. We use a 27-dimensional feature vector set that is based on color histograms in HSV color space (see Figure 22b). The features were extracted from the database as described in [Kriegel et al., 2011b] but using 2 bins for hue, saturation and brightness each. The data set is provided by the *ELKI* project [Achtert et al., 2012] and contains 110 250 data points. For the ALOI data set we computed Gaussian mixtures with $k = 3$ component distributions.

The third data set is created from data provided by the *GeoNames geographical database* (available at `http://www.geonames.org/`). The original data set contained several features of 135 082 cities around the world with a population of at least 1000. We extracted the geographic coordinates (latitude and longitude) for each city to compute a projection onto the 3-dimensional unit sphere. The resulting data set is a 3-dimensional model of the populated regions of the globe (see Figure 22c). For the GeoNames data set we computed Gaussian mixtures with $k = 20$ component distributions.

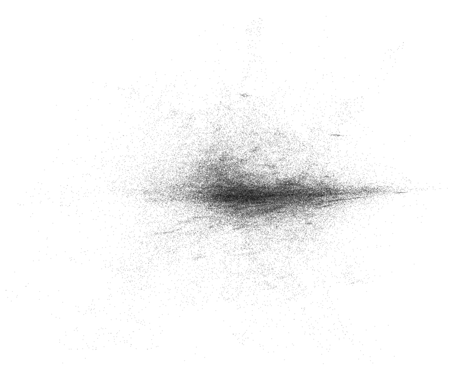
The first two real world data sets were normalized before use. That is, for each coordinate of the input data points the values were translated and scaled to fit into the interval $[0, 1]$. Thus, the spread in each dimension is $\Delta = 1$ (cf. Equation (28)). Without normalization, the difference between the means and covariances computed by the EM and SEM algorithm might be dominated by a single or only few dimensions of the input space. By introducing the normalization step, we ensured that the parameter estimation problem does not effectively reduce to a lower dimensional problem.

7.3 EXPERIMENTS

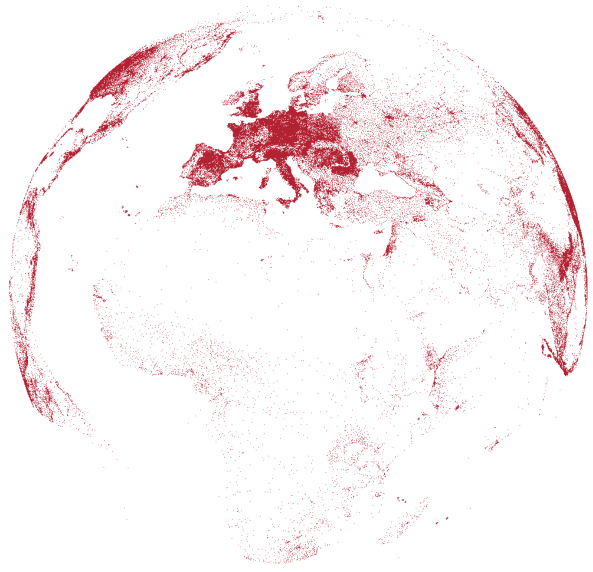
For the comparison of the EM and SEM algorithm, we established four different types of tests. In the first type of tests we compared the negative log-likelihood of the parameters computed by the EM and SEM algorithm. To do this, we ran both algorithms for 50 rounds and computed the negative log-likelihood of the computed parameters after each round.

The goal of the second type of tests was to compare the parameters computed by the two algorithms. Again, we ran both algorithms for 50 rounds. After each round, we computed the differences between the computed parameters. More precisely, for each component distri-

(a) Forest Covertype data set with $d = 10$ and $n = 110\,250$



(b) ALOI features data set with $d = 27$ and $n = 110\,250$



(c) GeoNames data set with $d = 3$ and $n = 135\,082$

Figure 22: Projections of the real world data sets.

bution we computed the absolute distance between the weights, the Euclidean distance between the means and the Frobenius norm of the difference between the covariance matrices.

The aim of the third type of tests was to evaluate our theoretical bounds on the proximity of the EM and SEM computations (cf. Chapter 6.3.2). For the sake of simplicity, we compared only the means computed by the two algorithms. To to this, in each round of the SEM algorithm we also computed the mean updates the EM algorithm would compute if it was given the same previous model parameters. Then, we determined the actual Euclidean distance between each pair of means. Furthermore, we computed our theoretical bounds for each component distribution and each coordinate of the corresponding mean. To get a bound that holds with high probability, we chose

$$\delta := \frac{1}{100 \cdot k(d+1)}.$$

Applying Theorem 6.3 for all k weight updates and Theorem 6.4 for all d coordinates in all k mean updates yields $k(d+1)$ bounds that hold with probability $1-\delta$. Using the union bound, the resulting bound for the Euclidean distance between the means holds with probability of at least $1-\frac{1}{100}$.

Finally, in the fourth type of tests, we compared the running times of the EM and SEM algorithm. Our analysis in Chapter 6.3.6 predicts a constant factor speedup in case of Gaussian mixture models. Furthermore, we motivated the overall speedup based on a factor k speedup for a part of the computations of the parameter updates. To investigate the dependency of the speedup on the number of component distributions, we started both algorithm with the GeoNames data set for several values of k between 1 and 100. To reduce the effects of single delayed runs and in case of the SEM algorithm to consider the probabilistic behavior, we averaged the running times of both algorithms over ten runs.

More precisely, we started both algorithms for each value of k with three different initial solutions. For each initial solution, we executed both algorithms three times.

The EM algorithm, as well as the SEM algorithm, compute their solution based on an initial parameter estimate. How to compute a good initial set of model parameters is not a subject of this thesis. Since we study the solutions of the SEM algorithm in relation to the solutions of the classical EM algorithm, we only have to assure that both algorithms are started with the same initial solutions. For our comparison of the EM and SEM algorithm, the quality of the initial solution is not critical. Therefore, we implemented a very simple initialization method that we used for all our experiments. To compute initial parameters $\theta = (w_1, \dots, w_k, \mu_1, \dots, \mu_k, \Sigma_1, \dots, \Sigma_k)$, we executed the following steps:

1. Draw k points $c_1, \dots, c_k$ from the input data set $X \subset \mathbb{R}^d$ uniformly at random.

2. Compute a partition $C_1, \dots, C_k$ of X into k subsets by assigning each $x \in X$ to the nearest c_i in terms of Euclidean distance (breaking ties arbitrarily).

3. For $i = 1, \dots, k$:
   a) Set the i-th weight to $w_i := \frac{|C_i|}{|X|}$.
   b) Set μ_i, Σ_i to the maximum likelihood estimate of a single Gaussian for the data set C_i (see Chapter 4.3.1).

For each data set, we created 10 sets of *initial model parameters*. Since the SEM algorithm is probabilistic, for each initial model we performed 100 different runs. To ensure that the results of our second and third type of test are not biased by an unlikely computation, we averaged the results of the different runs for each initial solution. Recall that the EM algorithm is deterministic. Therefore, we need to execute the EM algorithm only once.

7.4 RESULTS

In this section, we discuss the results of our four types of tests. The focus of the following presentation lies on the second and third type of test since these tests correspond to our analysis in Chapter 6. As mentioned above, we discuss the results for five particular data sets, two artificial data sets and three real world data sets. During our experiments we observed that the results for different initial solutions did not differ substantially. Therefore, we depict only results for some selected initial solutions that led to typical behavior of the algorithms.

Some characteristic results of our first type of tests are shown in Figure 23. The negative log-likelihood of the solutions computed by the EM algorithm is plotted as a dashed black line. Note that for each round of the EM algorithm we only have a single value to plot. For the SEM algorithm, we have to plot the results of 100 different runs of the algorithm. Therefore, we depict the results of the SEM algorithm similar to box plots. The dark gray line marks the median, the gray ribbon ranges from the lower to the upper quartile, and the light gray ribbon ranges from the minimum to the maximum of the negative log-likelihood of the computed solutions. For the majority of our experiments for the artificial as well as the real world data sets, the log-likelihood of the solutions of the EM and SEM algorithm almost coincide. To make the differences visible at all in Figure 23, we had to omit the first three rounds of the algorithms. For the Art1M data set, most of the initial solutions still lead to results as depicted in Figure 23a. However, for some initial solutions we observe small differences as depicted in Figure 23c. Furthermore, individual initial solutions lead to considerable differences as depicted in Figure 23e.

Generally speaking, for small values of n one can not expect that the SEM algorithm yields parameter estimates close to the EM computations. This is due to the law of large numbers, i. e., the influence of a single sampling step of the SEM algorithm is larger for smaller n. Indeed, for Art10K data set we observe differences between the log-likelihood of the computed solutions more often. Some examples are depicted in Figure 23b to Figure 23f. Note that the SEM results may lie above the EM results as well as below them and also both in the same run.

Some typical results of our second type of tests are shown in Figure 24 to Figure 28. For the interpretation of the results, we will relate the measured differences between the parameters computed by the EM and SEM algorithm to a simple upper bound derived from the dimension d and the largest spread Δ of the corresponding data set. That is, for a given data set, we assume that the spread in each dimension is at most Δ. Then, the Euclidean distance between the computed means is at most

See Chapter 6.3.2, Equation (28).

$$\Gamma_\mu := \Delta\sqrt{d}.$$

The Frobenius norm of the difference between the computed covariance matrices is at most

$$\Gamma_\Sigma := d\Delta^2.$$

Furthermore, recall that the weights lie in the interval $[0,1]$. That is, the difference between the computed weights is at most 1. In the following, we compute the fraction of Γ_μ and Γ_Σ which the computed means and covariance matrices differ by, respectively. The differences of the weights are always compared to 1.

For the artificial data sets, we have $d = 10$ and $\Delta \approx 40$. It follows that $\Gamma_\mu \approx 130$ and $\Gamma_\Sigma \approx 16\,000$. Regarding the Art1M data set (see Figure 24), we observe that the parameter vectors computed by the EM and SEM algorithm are very similar. More precisely, the weights differ by less than 0.002 and the means differ by less than $0.002 \cdot \Gamma_\mu$. The covariances even differ by less than $0.0002 \cdot \Gamma_\Sigma$. Moreover, we observe that after the first few rounds, the differences in the parameters of many components are even substantially smaller than those of the remaining components.

As mentioned in the discussion of our first type of tests, for small values of n one can not expect good parameter estimates. Indeed, for the Art10K data set (see Figure 25), the differences between the means and covariances are larger by a factor of 20. However, our first type of tests show that this does not necessarily result in lower likelihood of the computed parameters.

Recall that the two real world data sets corresponding to Figure 26 and Figure 27 are normalized. That is, $\Delta = 1$.

For the Forest Covertype data set (see Figure 26), we have $d = 10$. It follows that $\Gamma_\mu \approx 3.2$ and $\Gamma_\Sigma = 10$. Then, the weights differ by less than 0.004, the means differ by less than $0.004 \cdot \Gamma_\mu$, and the covariances differ by less than $0.0002 \cdot \Gamma_\Sigma$. That is, the results are similar to the results for the Art1M data set. One explanation for these good results is that the Forest Covertype data set is the largest of the real world data sets with a size of $n = 581\,012$.

For the ALOI data set (see Figure 27), we have $d = 27$. It follows that $\Gamma_\mu \approx 5.2$ and $\Gamma_\Sigma = 27$. For the weight updates, at least in the first couple of rounds, we get a difference of up to 0.2. The means differ by up to $0.009 \cdot \Gamma_\mu$ and the covariances differ by up to $0.0006 \cdot \Gamma_\Sigma$. These results are slightly worse than the results for the Forest Covertype data set. However, this can be explained by the smaller size of the ALOI data set with $n = 110\,250$ only. Furthermore, except for the difference of some weight updates during the first few rounds, the differences are still very small.

For the GeoNames data set (see Figure 28), we have $d = 3$ and $\Delta = 2$. It follows that $\Gamma_\mu \approx 3.5$ and $\Gamma_\Sigma = 12$. For the weights, we get a difference of less than 0.003. The means and the covariances differ by a fraction of less than $0.01 \cdot \Gamma_\mu$ and $0.0015 \cdot \Gamma_\Sigma$, respectively. However, for all but one component, the differences between the means and between the covariances are lower by a factor of 3 at least.

All three real world data sets have in common that after approximately 40 rounds, the differences in all parameters remain on a very small level. Furthermore, we point out that all results confirm that both algorithms compute almost the same parameter updates.

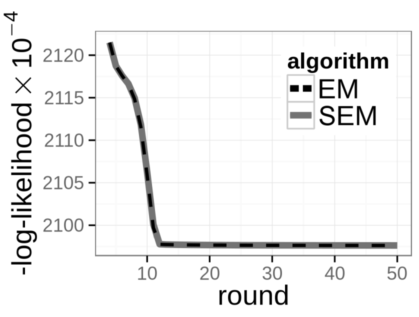
For our third type of tests, we depict a selection of results in Figure 29 to Figure 33. First, we observe that the actual difference measured in our experiments is significantly smaller than our high probability bound. Furthermore, the similar development of our bound and the actual difference indicates the accuracy of our analysis. Note that mostly, the largest differences exist during the first couple of rounds. This matches our observations from the second type of tests.

Just as for our second type of test, we relate the bound for the difference of the weight updates to the largest possible distance determined by the dimension and the spread of the data set. As mentioned in the description of our tests in Section 7.3, the presented bounds hold with probability $1 - \frac{1}{100}$. For the Art1M data set (cf. Figure 29), the results show that the bound is less than $0.013 \cdot \Gamma_\mu$. Again, due to the smaller size of the data set, the results for the Art10K data set are worse (cf. Figure 30). Here, we get bounds up to $0.12 \cdot \Gamma_\mu$. In Chapter 6.3.3, we already discussed the limited applicability of our bounds for small responsibilities r_j. Since smaller values of n result in smaller values of $r_j = n \cdot w_j^{EM}$, the larger bounds observed in Figure 30 are not surprising.

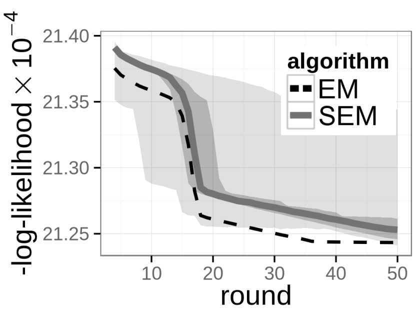
For the real world data sets, we observed a larger variety of developments of our high probability bound. This may arise from the fact that the real world data sets are less suitable for being modeled by Gaussian mixtures. To avoid tedious repetitions, we prefer to present a diverse choice of results rather than a characteristic selection. Nevertheless, all presented bounds for all real world data sets are at most $0.1 \cdot \Gamma_\mu$.

Finally, the results of our fourth type of tests are depicted in Figure 34 and Figure 35. Figure 34 shows that the running times of both algorithms linearly depend on the number of components k. The slope of the curve for the SEM algorithm is considerably smaller as predicted by our analysis in Chapter 6.3.6. Furthermore, the results show that for $k = 1$ the EM algorithm is faster than the SEM algorithm. This is not surprising since for $k = 1$ both algorithms compute the same updates, while the SEM processes the additional sampling step. To approximate the factor of the speedup, we did further tests for larger values of k. The results are depicted in Figure 35 and suggest a factor of up to 3 at least.

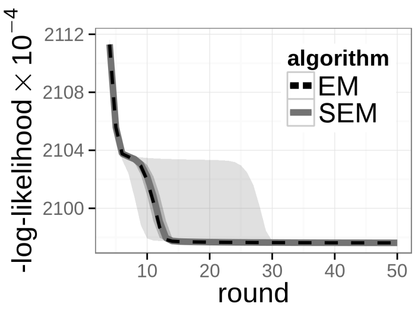
Note that although the theoretical and experimental running time analyses seem to match, we have to treat the results with care. On the one hand, our theoretical analysis simplifies the situation by only counting the multiplications. On the other hand, our implementation may be subject to several optimizations emerging from the linear algebra library, the compiler and the floating point hardware. Furthermore, these optimizations of the implementation possibly differ in their effectiveness for the two algorithms. Nevertheless, the measured speedup is real and thus, the SEM algorithm should be taken into account when it comes to the parameter estimation problem for Gaussian mixture models.



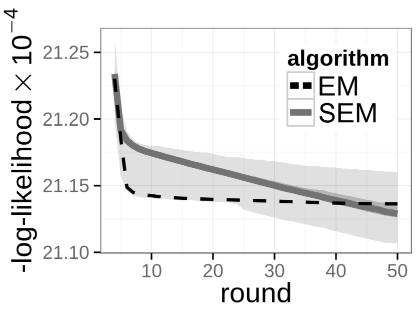
(a) Art1M data set



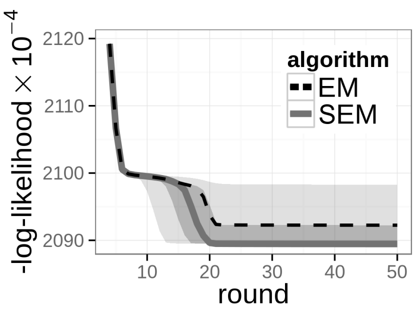
(b) Art10K data set



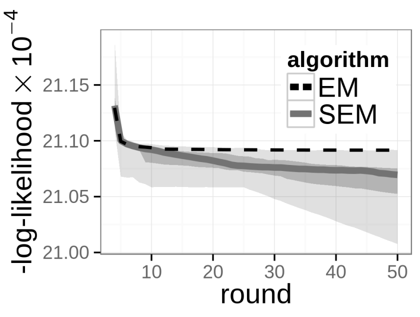
(c) Art1M data set



(d) Art10K data set

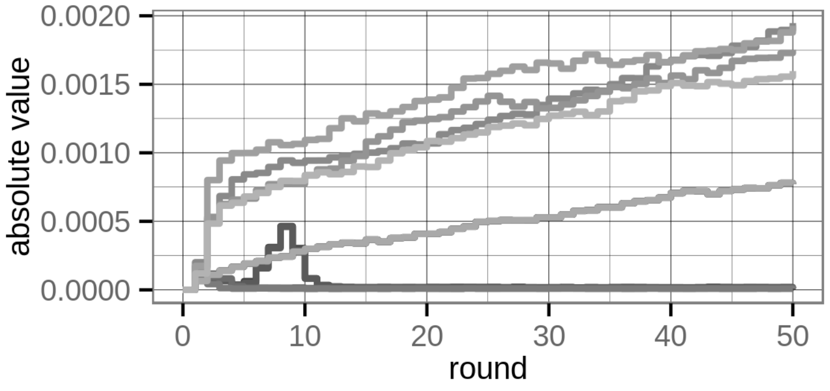


(e) Art1M data set

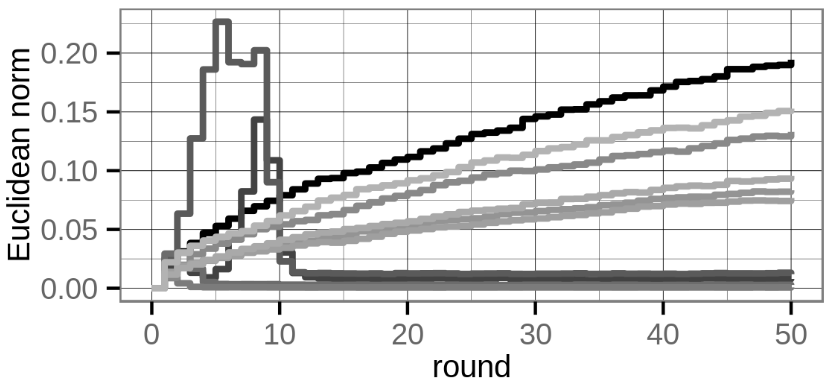


(f) Art10K data set

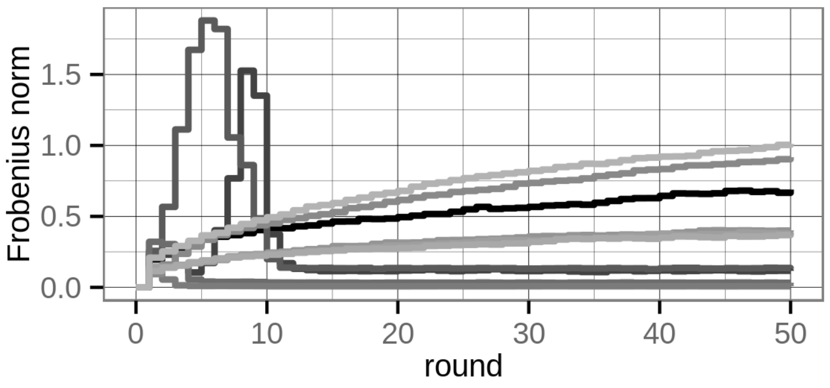
Figure 23: Negative log-likelihood of parameters computed by the EM and SEM algorithm for the selected artificial data sets and different initial solutions. Figure 23a, 23c, and 23e all depict results for the Art1M data set, but for different initial solutions. The same holds for Figure 23b, 23d, and 23f with the Art10K data set.



(a) Difference $w_k^{EM} - w_k^{SEM}$ between the weights.

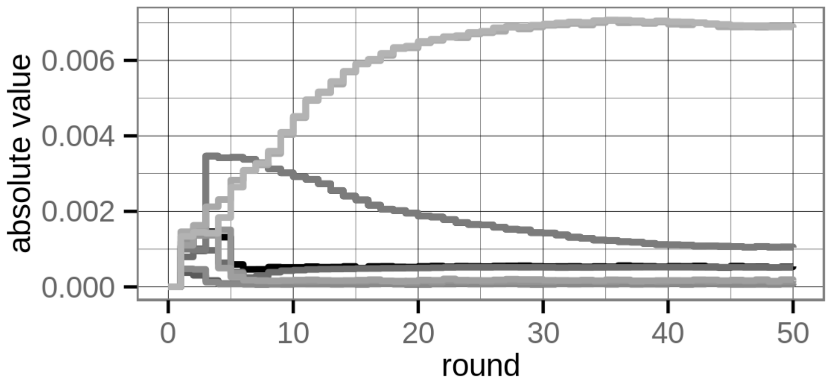


(b) Difference $\mu_k^{EM} - \mu_k^{SEM}$ between the means.

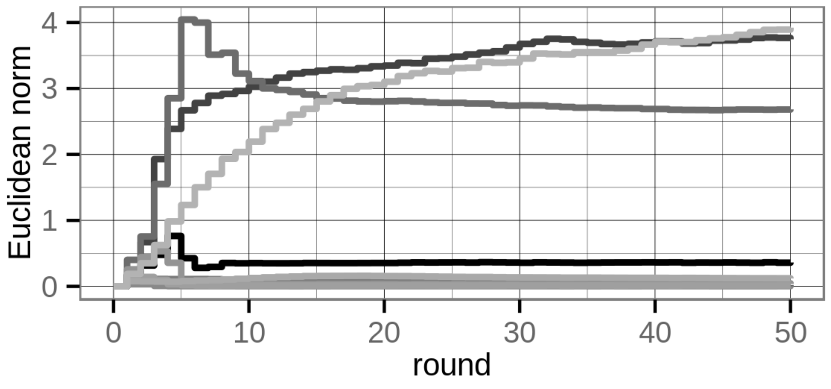


(c) Difference $\Sigma_k^{EM} - \Sigma_k^{SEM}$ between the covariances.

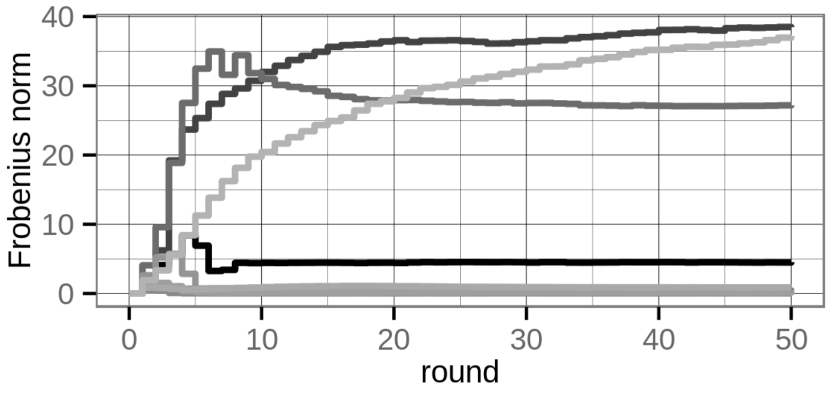
Figure 24: Comparison of intermediate solutions of the EM and SEM algorithm given the *Art1M* data set with $\Gamma_\mu \approx 130$ and $\Gamma_\Sigma \approx 16\,000$ for $k = 10$. All three figures depict the results for the same initial solution. Each figure consists of ten graphs, each depicting the difference for one component Gaussian averaged over 100 SEM runs.



(a) Difference $w_k^{EM} - w_k^{SEM}$ between the weights.

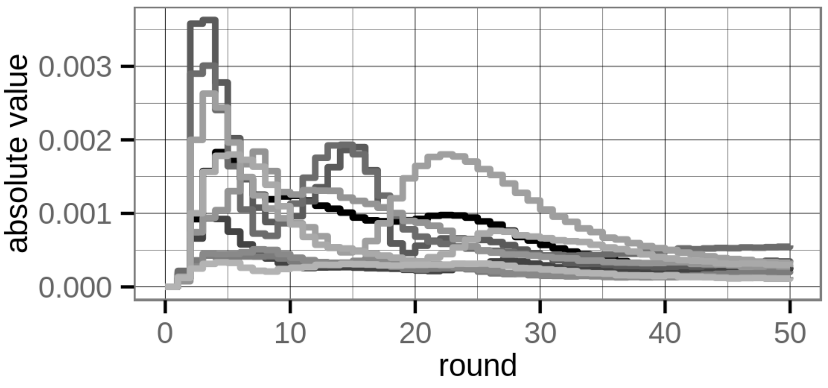


(b) Difference $\mu_k^{EM} - \mu_k^{SEM}$ between the means.

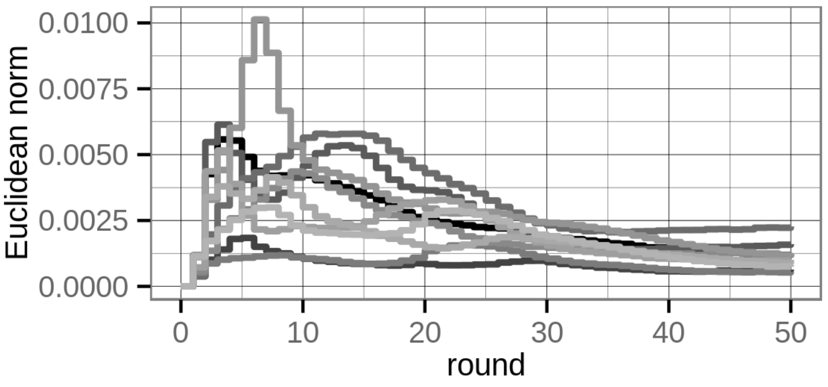


(c) Difference $\Sigma_k^{EM} - \Sigma_k^{SEM}$ between the covariances.

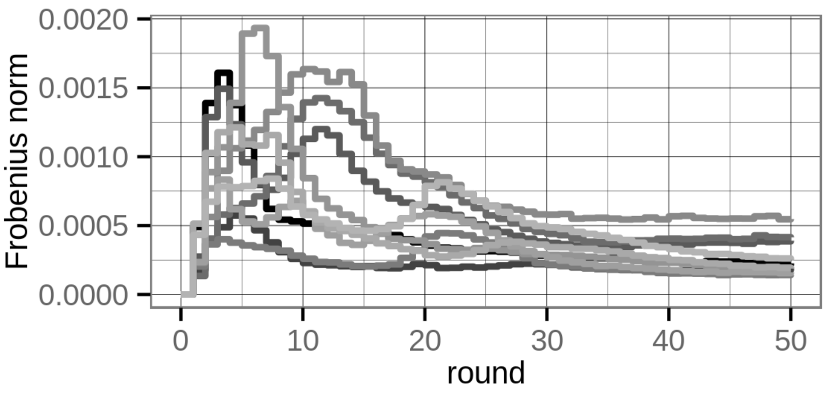
Figure 25: Comparison of intermediate solutions of the EM and SEM algorithm given the *Art10K* data set with $\Gamma_\mu \approx 130$ and $\Gamma_\Sigma \approx 16\,000$ for $k = 10$. All three figures depict the results for the same initial solution. Each figure consists of ten graphs, each depicting the difference for one component Gaussian averaged over 100 SEM runs.



(a) Difference $w_k^{EM} - w_k^{SEM}$ between the weights.

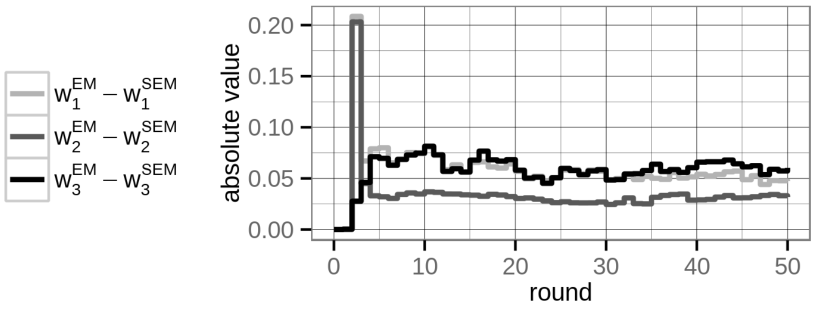


(b) Difference $\mu_k^{EM} - \mu_k^{SEM}$ between the means.

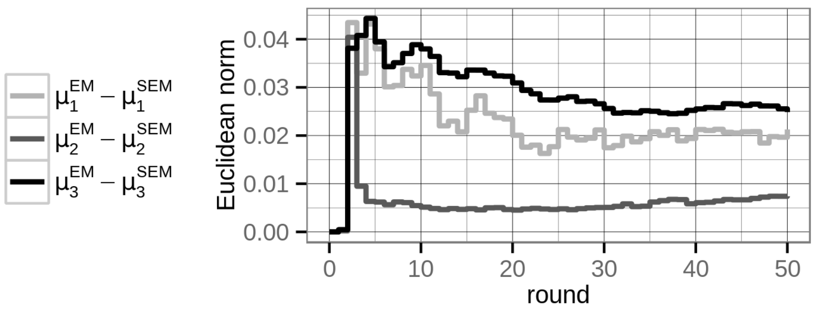


(c) Difference $\Sigma_k^{EM} - \Sigma_k^{SEM}$ between the covariances.

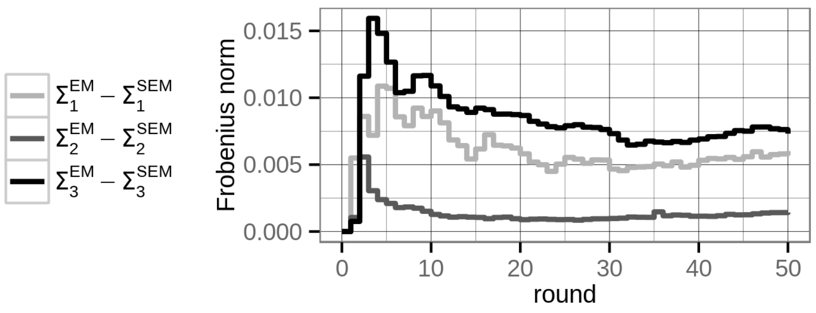
Figure 26: Comparison of intermediate solutions of the EM and SEM algorithm given the normalized *Forest Covertype* data set with $\Gamma_\mu \approx 3.2$ and $\Gamma_\Sigma = 10$ for $k = 10$. All three figures depict the results for the same initial solution. Each figure consists of ten graphs, each depicting the difference for one component Gaussian averaged over 100 SEM runs.



(a) Difference $w_k^{EM} - w_k^{SEM}$ between the weights.

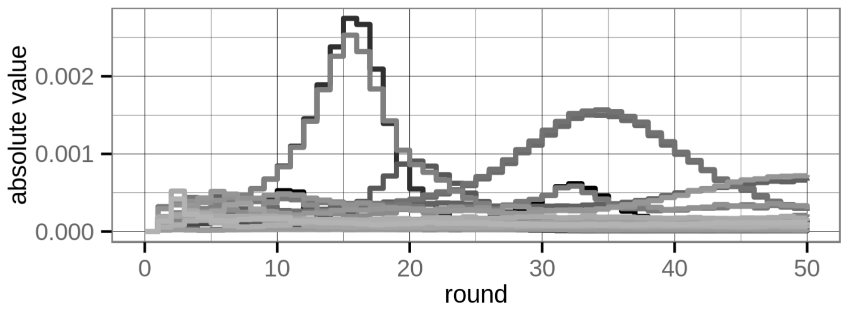


(b) Difference $\mu_k^{EM} - \mu_k^{SEM}$ between the means.

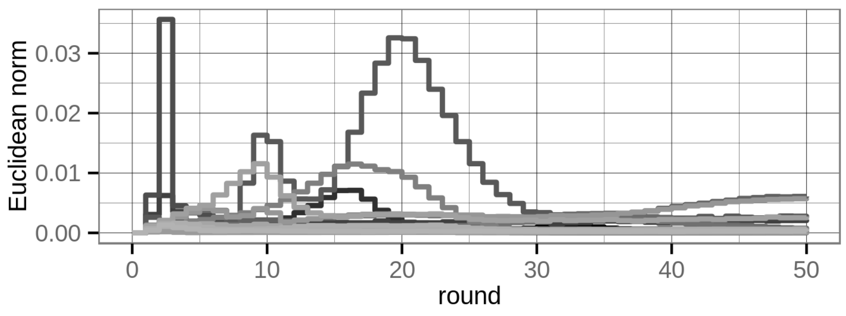


(c) Difference $\Sigma_k^{EM} - \Sigma_k^{SEM}$ between the covariances.

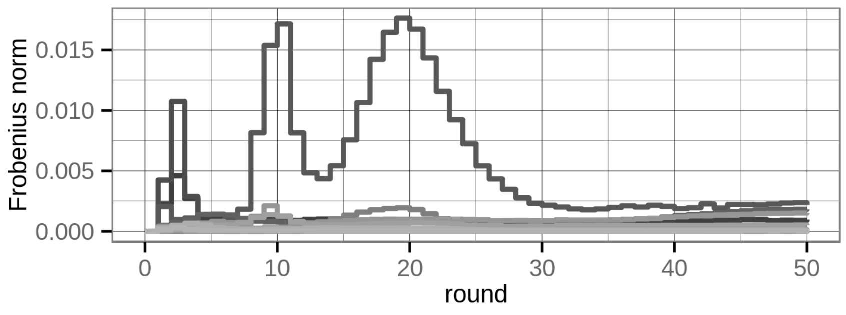
Figure 27: Comparison of intermediate solutions of the EM and SEM algorithm given the normalized *ALOI* data set with $\Gamma_\mu \approx 5.2$ and $\Gamma_\Sigma = 27$ for $k = 3$. All three figures depict the results for the same initial solution. Each figure consists of three graphs, each depicting the difference for one component Gaussian averaged over 100 SEM runs.



(a) Difference $w_k^{EM} - w_k^{SEM}$ between the weights.

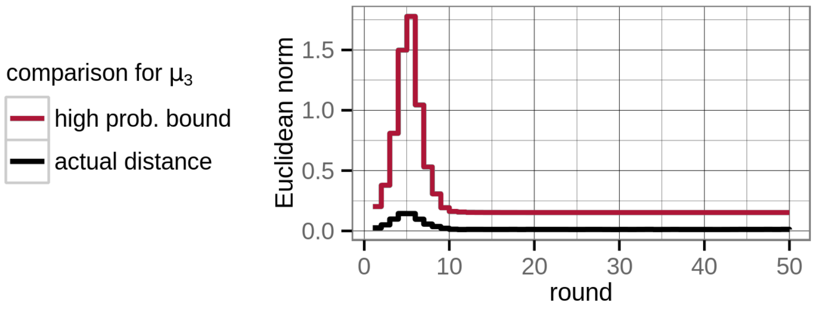


(b) Difference $\mu_k^{EM} - \mu_k^{SEM}$ between the means.

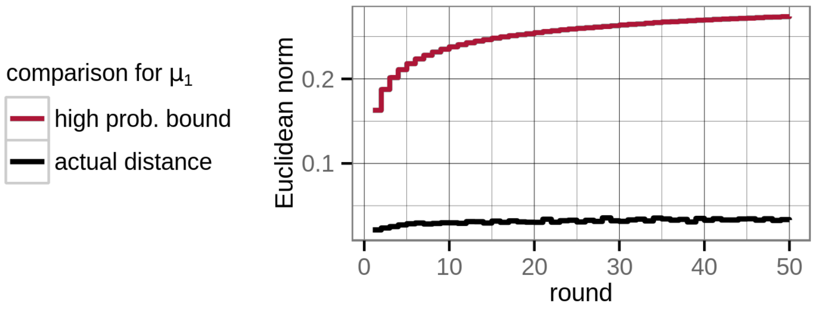


(c) Difference $\Sigma_k^{EM} - \Sigma_k^{SEM}$ between the covariances.

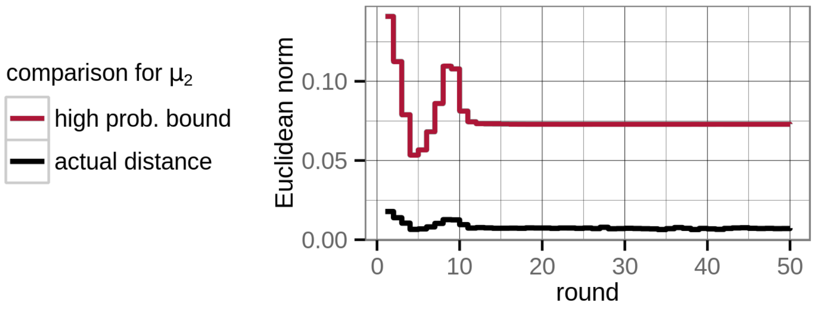
Figure 28: Comparison of intermediate solutions of the EM and SEM algorithm given the *GeoNames* data set with $\Gamma_\mu \approx 3.5$ and $\Gamma_\Sigma = 12$ for $k = 20$. All three figures depict the results for the same initial solution. Each figure consists of twenty graphs, each depicting the difference for one component Gaussian averaged over 100 SEM runs.



(a) This is one of the most frequent developments of the bound.

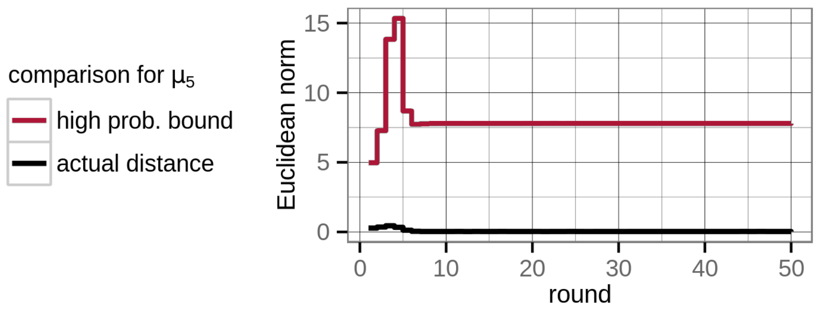


(b) Sometimes, the bound increases, albeit on a very low level.

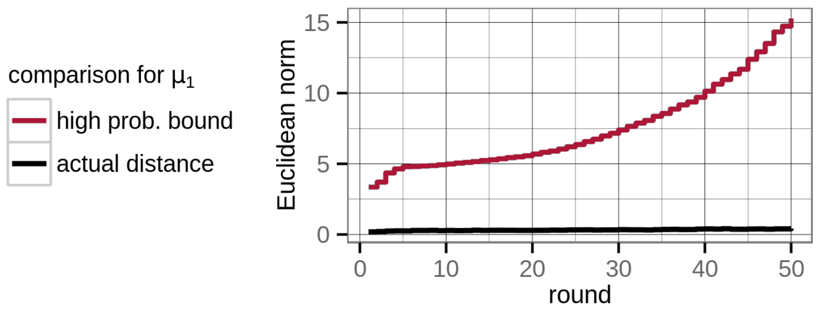


(c) Several ups and downs occur only rarely.

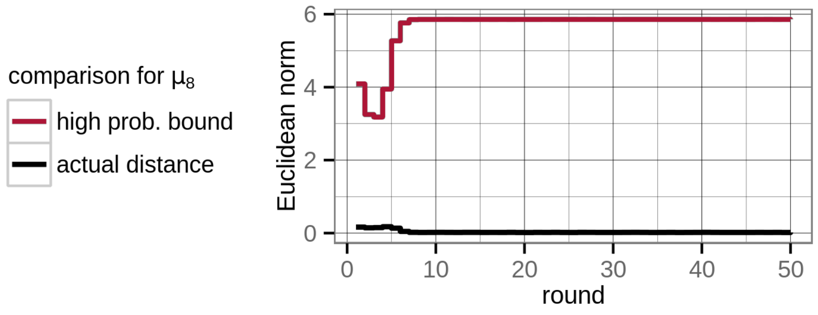
Figure 29: Experimental difference and the theoretical bound on the difference between means computed by the EM and SEM algorithm given the *Art1M* data set with $\Gamma_\mu \approx 130$ for $k = 10$. All three figures depict the results averaged over 100 SEM runs for the same initial solution but for the means of different component Gaussians.



(a) Similar to Figure 29a, but on a higher level.



(b) Although the bound is worsening, the actual differences remain small.



(c) Development of the bound contrary to the actual difference.

Figure 30: Experimental difference and the theoretical bound on the difference between means computed by the EM and SEM algorithm given the *Art10K* data set with $\Gamma_\mu \approx 130$ for $k = 10$. All three figures depict the results averaged over 100 SEM runs for the same initial solution but for the means of different component Gaussians.

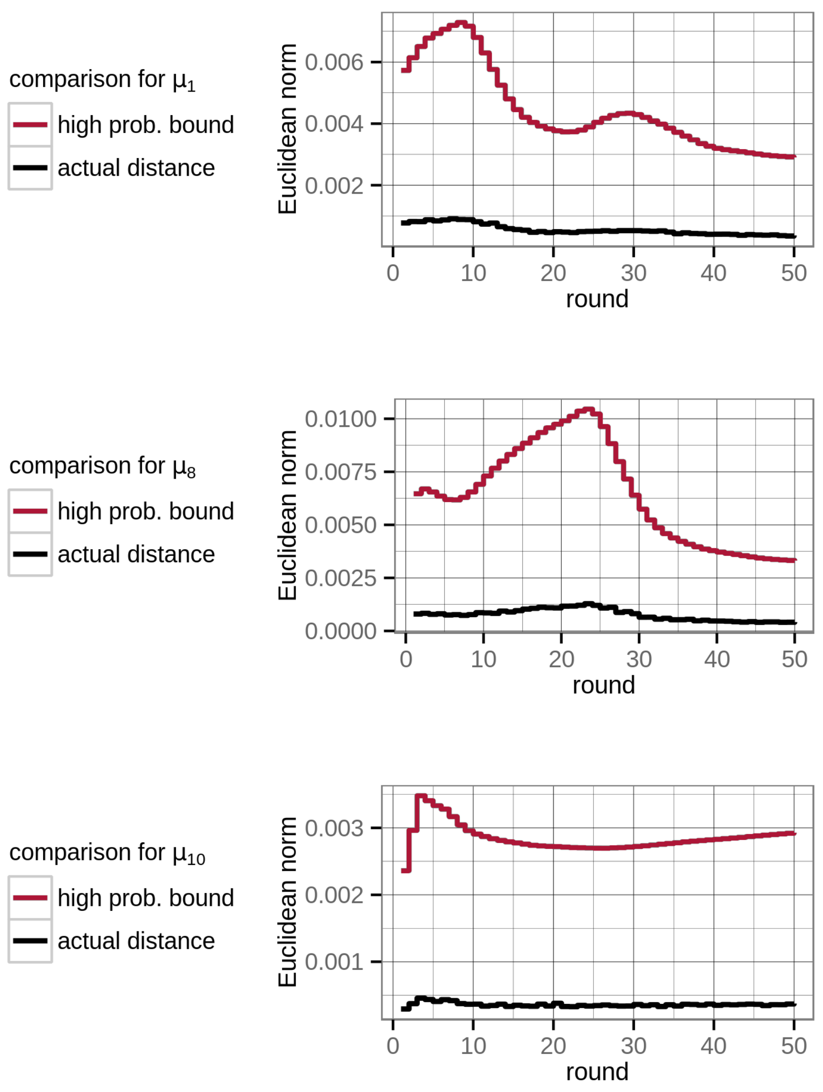


Figure 31: Experimental difference and the theoretical bound on the difference between means computed by the EM and SEM algorithm given the normalized *Forest Covertype* data set with $\Gamma_\mu \approx 3.2$ for $k = 10$. All three figures depict the results averaged over 100 SEM runs for the same initial solution but for the means of different component Gaussians.

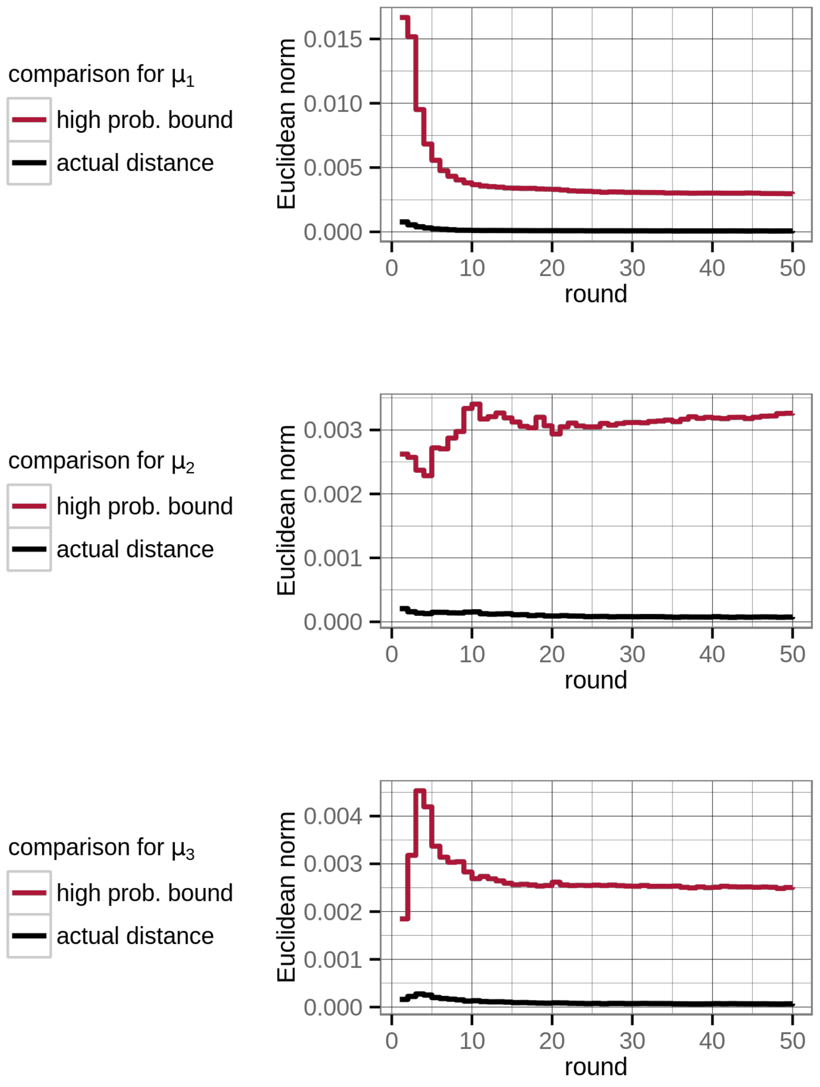


Figure 32: Experimental difference and the theoretical bound on the difference between means computed by the EM and SEM algorithm given the normalized *ALOI* data set with $\Gamma_\mu \approx 5.2$ for $k = 3$. All three figures depict the results averaged over 100 SEM runs for the same initial solution but for the means of different component Gaussians.

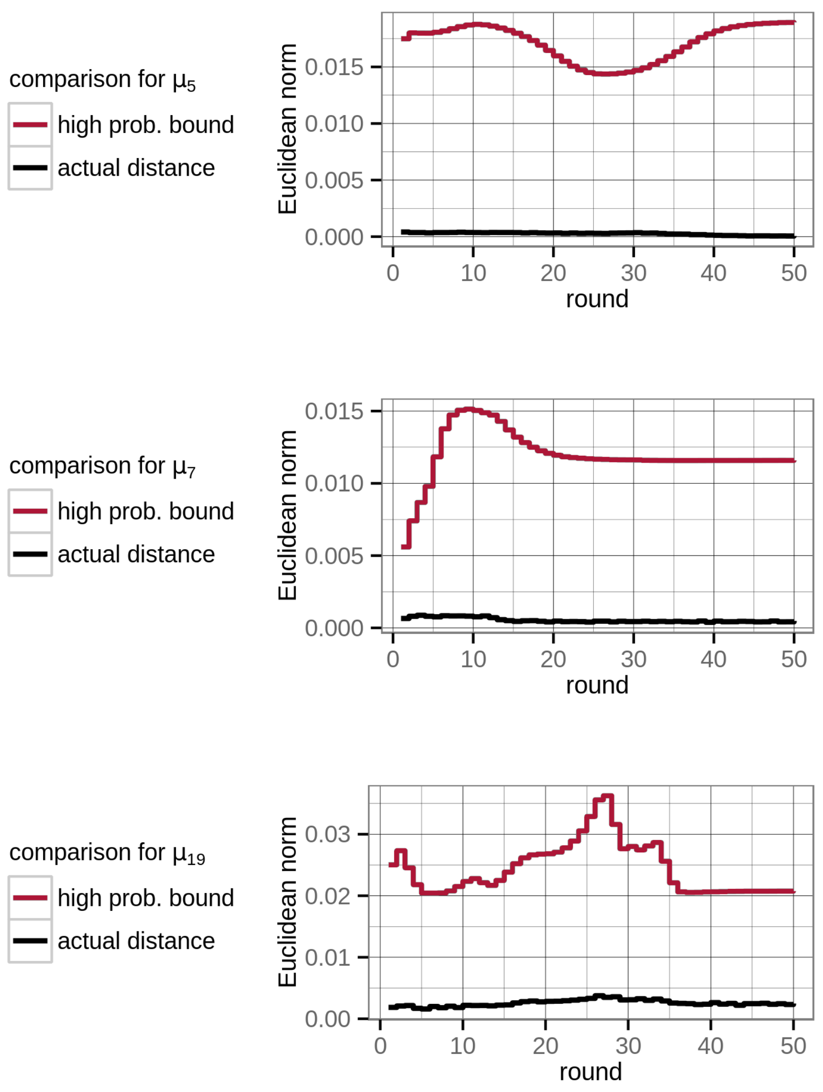


Figure 33: Experimental difference and the theoretical bound on the difference between means computed by the EM and SEM algorithm given the *GeoNames* data set with $\Gamma_\mu \approx 3.5$ for $k = 20$. All three figures depict the results averaged over 100 SEM runs for the same initial solution but for the means of different component Gaussians.

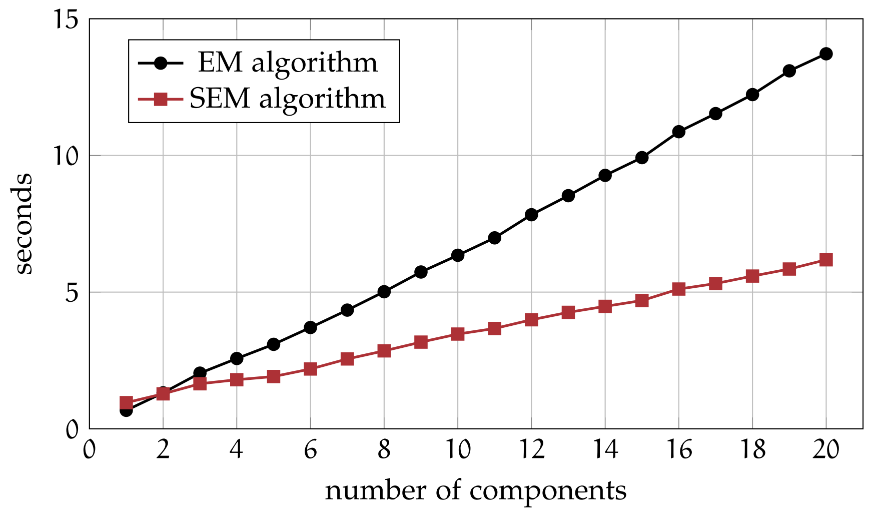


Figure 34: Average running times of the EM and SEM algorithm started on the GeoNames data set for different numbers of component Gaussians.

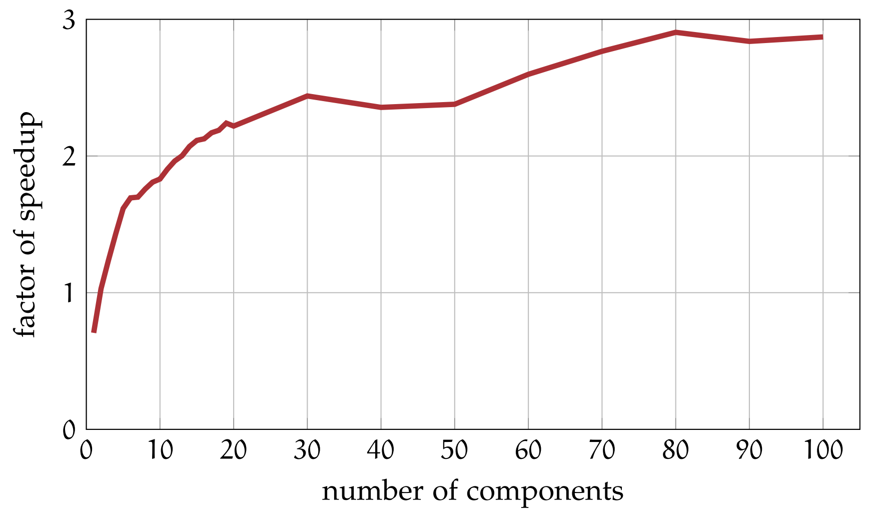


Figure 35: Average speedup of the SEM algorithm started on the GeoNames data set for different numbers of component Gaussians.

LIST OF FIGURES

Figure 1 Different measurements for the same cluster 15
Figure 2 The volume argument 19
Figure 3 $\mathrm{drad}(A \cup B) < \mathrm{cost}_{\mathrm{drad}}(\mathcal{C}_m) + \|p_A - p_B\|$ 22
Figure 4 Intermediate centers 26
Figure 5 Covering centers and clusters 27
Figure 6 Merging (Z, opt_k)-connected clusters 31
Figure 7 Congruent configurations 37
Figure 8 Merging clusters intersecting a ball of radius r 39
Figure 9 The two base cases of the induction 43
Figure 10 The inductive step 44
Figure 11 Partition of the optimal solution 44
Figure 12 Sketch of the set $W(x)$ 47
Figure 13 Part of the dendrogram for $W(x_i)$ 47
Figure 14 Part of the dendrogram for X 48
Figure 15 The Old Faithful data set 57
Figure 16 Density of a two-dimensional Gaussian 62
Figure 17 Points with constant Gaussian density 63
Figure 18 Density of a Gaussian mixture distribution 66
Figure 19 Decomposition of $\ln p_\theta(X)$ over the choice of q 74
Figure 20 Optimization of $\ln p_\theta(X)$ over the choice of θ 76
Figure 21 Projection of one of the artificial data sets 101
Figure 22 Projections of the real world data sets 103
Figure 23 Negative log-likelihood for artificial data sets 109
Figure 24 Differences for the Art1M data set 110
Figure 25 Differences for Art10K data set 111
Figure 26 Differences for the Forest Covertype data set 112
Figure 27 Differences for the ALOI data set 113
Figure 28 Differences for the GeoNames data set 114
Figure 29 Theoretical bound for the large artificial data set 115
Figure 30 Theoretical bound for the small artificial data set 116
Figure 31 Theoretical bound for the Forest Covertype data set 117
Figure 32 Theoretical bound for the ALOI data set 118
Figure 33 Theoretical bound for the GeoNames data set 119
Figure 34 Average running times for EM and SEM 120
Figure 35 Average speedup of the SEM algorithm 120

LIST OF ALGORITHMS

3.1 AgglomerativeDiscreteRadius(X) 20
3.2 AgglomerativeRadius(X) 24
3.3 AgglomerativeCompleteLinkage(X) 34

5.1 Classical EM algorithm 72

6.1 Probabilistic EM algorithm 82
6.2 SEM algorithm for mixtures 83

BIBLIOGRAPHY

[Achtert et al., 2012] Achtert, E., Goldhofer, S., Kriegel, H.-P., Schubert, E., and Zimek, A. (2012). Evaluation of clusterings – metrics and visual support. *Data Engineering, International Conference on*, 0:1285–1288.

[Ackermann, 2009] Ackermann, M. R. (2009). *Algorithms for the Bregman k-Median Problem*. PhD thesis, University of Paderborn.

[Ackermann et al., 2011] Ackermann, M. R., Blömer, J., Kuntze, D., and Sohler, C. (2011). Analysis of Agglomerative Clustering. In *Proceedings of the 28th International Symposium on Theoretical Aspects of Computer Science (STACS '11)*, pages 308–319. Dagstuhl Publishing.

[Ackermann et al., 2012] Ackermann, M. R., Blömer, J., Kuntze, D., and Sohler, C. (2012). Analysis of Agglomerative Clustering. *Algorithmica*.

[Amunts et al., 2013] Amunts, K., Lepage, C., Borgeat, L., Mohlberg, H., Dickscheid, T., Rousseau, M.-E., Bludau, S., Bazin, P.-L., Lewis, L. B., Oros-Peusquens, A.-M., Shah, N. J., Lippert, T., Zilles, K., and Evans, A. C. (2013). BigBrain: An Ultrahigh-Resolution 3D Human Brain Model. *Science*, 340(6139):1472–1475.

[Asuncion and Newman, 2007] Asuncion, A. and Newman, D. J. (2007). UCI machine learning repository. Available at: http://www.ics.uci.edu/~mlearn/MLRepository.html.

[Azevedo et al., 2009] Azevedo, F. A., Carvalho, L. R., Grinberg, L. T., Farfel, J. M., Ferretti, R. E., Leite, R. E., Filho, W. J., Lent, R., and Herculano-Houzel, S. (2009). Equal numbers of neuronal and non-neuronal cells make the human brain an isometrically scaled-up primate brain. *The Journal of Comparative Neurology*, 513(5):532–541.

[Bādoiu et al., 2002] Bādoiu, M., Har-Peled, S., and Indyk, P. (2002). Approximate clustering via core-sets. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, STOC '02, pages 250–257, New York, NY, USA. ACM.

[Baum et al., 1970] Baum, L. E., Petrie, T., Soules, G., and Weiss, N. (1970). A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains. *The Annals of Mathematical Statistics*, 41(1):164–171.

[Biernacki et al., 2003] Biernacki, C., Celeux, G., and Govaert, G. (2003). Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate Gaussian mixture models. *Computational Statistics & Data Analysis*, 41(3-4):561–575.

[Bilmes, 1997] Bilmes, J. (1997). A Gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models. Technical Report TR-97-021, ICSI.

[Bishop, 2006] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[Blömer et al., 2013] Blömer, J., Bujna, K., and Kuntze, D. (2013). A Theoretical and Experimental Comparison of the EM and SEM Algorithm. *CoRR: Computing Research Repository*, abs/1310.5034. arXiv:1310.5034 [cs.LG].

[Brewer et al., 2009] Brewer, G. J., Boehler, M. D., Pearson, R. A., DeMaris, A. A., Ide, A. N., and Wheeler, B. C. (2009). Neuron network activity scales exponentially with synapse density. *Journal of Neural Engineering*, 6(1):014001.

[Broder et al., 1997] Broder, A. Z., Glassman, S. C., Manasse, M. S., and Zweig, G. (1997). Syntactic clustering of the web. *Computer Networks and ISDN Systems*, 29(8–13):1157–1166. Papers from the Sixth International World Wide Web Conference.

[Celeux et al., 1995] Celeux, G., Chauveau, D., and Diebolt, J. (1995). On stochastic versions of the em algorithm. Technical Report 2514, Institut national de recherche en informatique et en automatique (INRIA).

[Celeux and Diebolt, 1985] Celeux, G. and Diebolt, J. (1985). The SEM algorithm: a probabilistic teacher algorithm derived from the EM algorithm for the mixture problem. *Computational Statistics Quarterly*, 2:73–82.

[Celeux and Diebolt, 1992] Celeux, G. and Diebolt, J. (1992). A stochastic approximation type EM algorithm for the mixture problem. *Stochastics and Stochastic Reports*, 41(1-2):119–134.

[Charikar et al., 1997] Charikar, M., Chekuri, C., Feder, T., and Motwani, R. (1997). Incremental clustering and dynamic information retrieval. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, STOC '97, pages 626–635, New York, NY, USA. ACM.

[Cover and Thomas, 2012] Cover, T. M. and Thomas, J. A. (2012). *Elements of information theory*. John Wiley & Sons.

[Dasgupta and Long, 2005] Dasgupta, S. and Long, P. M. (2005). Performance guarantees for hierarchical clustering. *Journal of Computer and System Sciences*, 70(4):555–569. Special Issue on COLT 2002.

[Dasgupta and Schulman, 2007] Dasgupta, S. and Schulman, L. (2007). A Probabilistic Analysis of EM for Mixtures of Separated, Spherical Gaussians. *Journal of Machine Learning Research*, 8:203–226.

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B: Statistical Methodology*, 39(1):1–38.

[Dias and Wedel, 2004] Dias, J. G. and Wedel, M. (2004). An empirical comparison of EM, SEM and MCMC performance for problematic Gaussian mixture likelihoods. *Statistics and Computing*, 14(4):323–332.

[Eisen et al., 1998] Eisen, M. B., Spellman, P. T., Brown, P. O., and Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, 95(25):14863–14868.

[Feder and Greene, 1988] Feder, T. and Greene, D. (1988). Optimal algorithms for approximate clustering. In *Proceedings of the twentieth annual ACM symposium on Theory of computing*, STOC '88, pages 434–444, New York, NY, USA. ACM.

[Florek et al., 1951] Florek, K., Lukaszewicz, J., Perkal, J., Steinhaus, H., and Zubrzycki, S. (1951). Sur la liaison et la division des points d'un ensemble fini. *Colloquium Mathematicae*, 2:282–285.

[Fortunato, 2010] Fortunato, S. (2010). Community detection in graphs. *Physics Reports*, 486(3-5):75 – 174.

[Fréchet, 1910] Fréchet, M. (1910). Les dimensions d'un ensemble abstrait. *Mathematische Annalen*, 68(2):145–168.

[Geusebroek et al., 2005] Geusebroek, J. M., Burghouts, G. J., and Smeulders, A. W. M. (2005). The amsterdam library of object images. *International Journal of Computer Vision*, 61(1):103–112.

[Golub and Van Loan, 2012] Golub, G. and Van Loan, C. (2012). *Matrix Computations*. Matrix Computations. Johns Hopkins University Press.

[Gonzalez, 1985] Gonzalez, T. F. (1985). Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38(0):293–306.

[Grimmett and Stirzaker, 2001] Grimmett, G. and Stirzaker, D. (2001). *Probability and Random Processes*. Probability and Random Processes. Clarendon Press.

[Hawkins, 2004] Hawkins, D. M. (2004). The problem of overfitting. *Journal of chemical information and computer sciences*, 44(1):1–12.

[Ip, 1994] Ip, E. H.-s. (1994). *A stochastic EM estimator in the presence of missing data – theory and applications*. PhD thesis, Stanford University.

[ISO, 2012] ISO (2012). *ISO/IEC 14882:2011 Information technology — Programming languages — C++*. International Organization for Standardization, Geneva, Switzerland.

[Johnson and Joram, 1984] Johnson, W. B. and Joram, L. (1984). Extensions of Lipschitz mappings into a Hilbert space. In *Conference in Modern Analysis and Probability*, volume 26 of *Contemporary Mathematics*, pages 189–206. American Mathematical Society.

[Koch et al., 2006] Koch, K., McLean, J., Segev, R., Freed, M. A., Berry, M. J., Balasubramanian, V., and Sterling, P. (2006). How much the eye tells the brain. *Current biology : CB*, 16(14):1428–1434.

[Kriegel et al., 2011a] Kriegel, H.-P., Kröger, P., Sander, J., and Zimek, A. (2011a). Density-based clustering. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1(3):231–240.

[Kriegel et al., 2011b] Kriegel, H.-P., Schubert, E., and Zimek, A. (2011b). Evaluation of multiple clustering solutions. In *Proceedings of the 2nd MultiClust Workshop (in conjunction with ECML PKDD)*.

[Leder, 2013] Leder, L. (2013). Nichtapproximierbarkeitsresultate zu Radius- und Durchmesserclustering unter Verwendung von L_p-Metriken. Bachelor's thesis. To appear.

[Lee et al., 2008] Lee, K., Kim, J., Kwon, K. H., Han, Y., and Kim, S. (2008). DDoS attack detection method using cluster analysis. *Expert Systems with Applications*, 34(3):1659–1665.

[Levchenko, 2013] Levchenko, K. (2013). Chernoff bound. Available at: http://www.cs.ucsd.edu/~klevchen/techniques/chernoff.pdf.

[Matsumoto and Nishimura, 1998] Matsumoto, M. and Nishimura, T. (1998). Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM Transactions on Modeling and Computer Simulation*, 8(1):3–30.

[McLachlan and Krishnan, 2008] McLachlan, G. J. and Krishnan, T. (2008). *The EM Algorithm and Extensions (Wiley Series in Probability and Statistics)*. Wiley-Interscience, 2 edition.

[McQuitty, 1957] McQuitty, L. L. (1957). Elementary Linkage Analysis for Isolating Orthogonal and Oblique Types and Typal Relevancies. *Educational and Psychological Measurement*, 17:207–209.

[Meyerhenke and Staudt, 2013] Meyerhenke, H. and Staudt, C. L. (2013). Engineering High-Performance Community Detection Heuristics for Massive Graphs. Accepted for full paper presentation at 42nd International Conference on Parallel Processing (ICPP).

[Mitzenmacher and Upfal, 2005] Mitzenmacher, M. and Upfal, E. (2005). *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, New York, NY, USA.

[Montgomery and Madison, 2004] Montgomery, J. M. and Madison, D. V. (2004). Discrete synaptic states define a major mechanism of synapse plasticity. *Trends in Neurosciences*, 27(12):744 – 750.

[Naszódi, 2010] Naszódi, M. (2010). Covering a set with homothets of a convex body. *Positivity*, 14(1):69–74.

[Pearson, 1894] Pearson, K. (1894). Contributions to the Mathematical Theory of Evolution. *Philosophical Transactions of the Royal Society of London. A*, 185:71–110.

[Pereira et al., 1993] Pereira, F., Tishby, N., and Lee, L. (1993). Distributional clustering of english words. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, ACL '93, pages 183–190, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Schaeffer, 2007] Schaeffer, S. E. (2007). Graph clustering. *Computer Science Review*, 1(1):27–64.

[Sneath and Sokal, 1973] Sneath, P. H. A. and Sokal, R. R. (1973). *Numerical taxonomy: the principles and practice of numerical classification*. W. H. Freeman.

[Sundberg, 1974] Sundberg, R. (1974). Maximum Likelihood Theory for Incomplete Data from an Exponential Family. *Scandinavian Journal of Statistics*, 1(2):49–58.

[Tan et al., 2005] Tan, P.-N., Steinbach, M., and Kumar, V. (2005). *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

[Turing, 1950] Turing, A. M. (1950). Computing Machinery and Intelligence. *Mind*, LIX:433–460.

[von Neumann, 1958] von Neumann, J. (1958). *The computer and the brain*. Mrs. Hepsa Ely Silliman memorial lectures. Yale University Press.

[Walker, 2012] Walker, R., editor (2012). *The Human Brain Project – A Report to the European Commission*. The HBP-PS Consortium, Lausanne.

[Webster, 1994] Webster, R. (1994). *Convexity*. Oxford Science Publications. Oxford University Press, USA.

[Wei and Tanner, 1990] Wei, G. C. G. and Tanner, M. A. (1990). A Monte Carlo Implementation of the EM Algorithm and the Poor Man's Data Augmentation Algorithms. *Journal of the American Statistical Association*, 85(411):699–704.

[Wu, 1983] Wu, C. F. J. (1983). On the Convergence Properties of the EM Algorithm. *The Annals of Statistics*, 11(1):95–103.