Statistical Learning with Similarity and Dissimilarity Functions

vorgelegt von
Dipl.-Math.
Ulrike von Luxburg
aus Tübingen

Von der Fakultät IV - Elektrotechnik und Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften
Dr. rer. nat.

genehmigte Dissertation

Promotionsausschuß:
Vorsitzender: Prof. Dr. F. Wysotzki
Berichter: Prof. Dr. S. Jähnichen
Berichter: Prof. Dr. K. Obermayer
Berichter: Prof. Dr. B. Schölkopf

Tag der wissenschaftlichen Aussprache: 24.11.2004

Berlin 2004
D83

Contents

Thanks!

For me, being a PhD student was a constant source of ups and downs,

... good times ...

... and bad times ...

times where I would have liked to throw in the towel, and times where I was very excited about my work. Finally I made it, and this would not have been possible without the support and inspiration I received by many people:

Bernhard Schölkopf guided me through my whole PhD time. He was always ready to give advice when I needed it and at the same time gave me the freedom and encouraged me to go my own way.

Alex Smola and Bob Williamson invited me to spend half a year at the Research School of Information Sciences and Engineering in Canberra. They introduced me to the field of machine learning, and I am grateful for their hospitality.

Most of all I am indebted to Olivier Bousquet, without whom this thesis never would have been completed. In countless discussions he explained me his view of learning theory, patiently answered my questions, and gave valuable hints and suggestions for my work. Whenever I came up with new ideas or results he shared my enthusiasm, always egging me further by asking exactly the right questions.

Doing a PhD is one thing, but doing it in such a stimulating environment as in our department is another thing. I would like to thank the members of the AGBS, all of whom were always generous to share their ideas and opinions and to give advice. In particular I am indebted to Matthias Hein, for many fruitful discussions and moral support; Malte Kuss, never lost for cheerful comments and suggestions; Felix Wichmann, always in the mood for a pleasant little chat at the coffee machine; and Jeremy Hill and Karin Bierig, my office mates.

Finally I would like to thank my family, especially Volker.

Chapter I

Introduction

This thesis explores statistical properties of machine learning algorithms from different perspectives. We will investigate questions arising both in the fields of supervised and unsupervised learning, dealing with diverse issues such as the convergence of algorithms, the speed of convergence, generalization bounds, and how statistical properties can be used in practical machine learning applications. All topics covered have the common feature that the properties of the similarity or dissimilarity function on the data play an important role and are the focus of our attention.

To set the scene for the rest of the thesis we want to pick up the terms "learning", "statistical", and "similarity and dissimilarity functions" from the title and briefly introduce their meaning in our context. This will be done on an informal level. Precise mathematical definitions will be given in the main part of the thesis, the idea being that all three main chapters can be read independently.

1.1 Learning

Learning is the process of inferring general rules from given examples. The examples are instances of some input space (pattern space), and the rules can consist of some general observation about the structure of the input space, or have the form of a functional dependency between the input and some output space. In this thesis we will be mainly concerned with two types of learning problems: classification and clustering. In both problems, the goal is to divide the input space into several regions such that objects within the same region "belong together" and "are different" from the objects in the other regions. The difference between the two problems is that classification is a supervised learning technique while clustering is unsupervised.

In classification we are given a set of training points $(x_i, y_i)_{i=1,\dots,n}$, each of them consisting of a pattern x_i and its label y_i. The goal is to infer a rule which can assign the correct label y to a new, previously unseen pattern x. We will introduce the standard mathematical framework for classification in Chapter III. In this framework it is well understood *what* we want to achieve, the question is only *how* we can

reach this goal efficiently under different conditions.

In clustering, our training data only consist of the training patterns $(x_i)_{i=1,\dots,n}$, without any label information. Here the goal is less well-defined. We want to discover "meaningful" classes in the data, but we have no information about how these classes might look or how many classes we actually have to find. Hence, for clustering both questions "what we want to achieve" and "how we do it" are interesting issues. These questions will be discussed in detail in Chapter II.

1.2 A statistical perspective on learning

Learning is often studied in a probabilistic setting. For supervised learning, the data space is given in form of a probability space $(\mathcal{X}\times\mathcal{Y}, \sigma(\mathcal{B}_\mathcal{X}\times\mathcal{B}_\mathcal{Y}), P)$ where $\mathcal{X}$ is the pattern space, $\mathcal{Y}$ the label space, $\mathcal{B}_\mathcal{X}$ and $\mathcal{B}_\mathcal{Y}$ are σ-algebras on $\mathcal{X}$ and $\mathcal{Y}$ (which we will omit in the following for simplicity), and P is a joint probability distribution of patterns and labels. We assume that the probability measure P is unknown, but that we can sample points from $\mathcal{X}\times\mathcal{Y}$ according to P. The training points $(x_i, y_i)_{i=1,\dots,n}$ are supposed to be drawn independently of each other according to the distribution P. The model for unsupervised learning works analogously. As in this case there is no label space $\mathcal{Y}$, we model the pattern space by a probability space $(\mathcal{X}, \mathcal{B}_\mathcal{X}, P)$ and again draw the training points $(x_i)_{i=1,\dots,n}$ independently according to P.

Due to the randomness in the generation of the training set it is natural to study properties of learning algorithms from a statistical point of view. Typically, one is interested in convergence issues on the one hand and the finite-sample performance on the other hand. For example, consider the following questions (which for simplicity are formulated for classification only):

- Does the classifier constructed by a given algorithm on a finite sample converge to a limit classifier if the sample size tends to infinity?
- In case of convergence, is the limit classifier the best one we can achieve? If not, how far is it away from the optimal solution?
- How fast does the convergence take place?
- How many sample points do we need to be able to construct a good classifier on the finite sample? Can we estimate the difference in performance between the finite sample classifier and the optimal solution, given only the finite sample?
- Is the result achieved on the finite sample stable, that is, does it change a lot if we randomly draw another training set or replace some of the training points by different ones?

Answers to some special cases of those questions are the core of this thesis. In Chapter II the focus will be on the convergence of clustering algorithms. For a certain class of algorithms, so-called spectral clustering algorithms, we will investigate

under which conditions convergence takes place and whether the limit clustering is a good clustering of the data space. In Chapter III we will derive a framework for classification on metric spaces. Among other topics, we will discuss the convergence and speed of convergence of the algorithms via generalization bounds. Finally, we will investigate in Chapter IV how certain statistical statements, namely generalization bounds in terms of compression coefficients, can be adopted for model selection purposes.

1.3 Similarity and dissimilarity

In both classification and clustering, in addition to the training points we need to have some information about the relation of the training points to each other. Often this additional information consists of similarity or dissimilarity measurements between the data points. A dissimilarity function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is usually understood to measure some kind of distance between points. In particular, dissimilarity functions are supposed to be increasing the more dissimilar two points get. The terms *dissimilarity* and *distance* are used synonymously. Special cases of dissimilarity functions are *metrics* (see below). *Similarity* functions $s : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (also called *affinity* functions) are in some sense the converse to dissimilarity functions: the similarity between two objects should grow if their dissimilarity decreases. In particular, a similarity function is supposed to increase the more similar the points are. *Kernel* functions are a special type of similarity functions (see below). In Section 1.5 we will define these terms more precisely and discuss their properties and relations to each other.

Spectral clustering is an algorithm which relies on a given similarity function on the training points. In Chapter II we will investigate which properties this similarity function has to satisfy to ensure the convergence of different spectral clustering algorithms. Chapter III deals with dissimilarities. We study in detail the structure of metric spaces and use our insights to construct a large margin classifier on metric spaces. Finally in Chapter IV, we study how parameters of a similarity function, such as a Gaussian kernel, can be determined from the training data.

One topic which will not be discussed in this thesis is the question how we choose a (dis)similarity function on the input space in the first place. Obviously, this choice will be crucial for the success of a learning algorithm, and it is important to pick a (dis)similarity function that reflects the relevant properties of the data. If such a (dis)similarity function is unknown, then one possible approach is to try and learn it from the training data. Approaches to do this have for example been discussed in Xing et al. (2003), Bie et al. (2003), Bousquet and Herrmann (2003), and Lanckriet et al. (2004). In this thesis, we will not discuss this question any further and just assume the (dis)similarity function to be given. In the last chapter, however, we will study how parameters of a given type of similarity functions can be determined.

1.4 Overview of the results

Now we want to give a short overview over the topics and results covered in the following chapters.

Chapter II: In this chapter we will study the convergence of a certain class of clustering algorithms, namely spectral clustering. Those algorithms rely on properties of graph Laplacian matrices, which are derived from the similarity matrix on the training points. The graph Laplacians are used either directly (unnormalized) or after a normalization step. Normalized or unnormalized spectral clustering then uses the coordinates of the first few eigenvectors of the normalized or unnormalized graph Laplacian, respectively, to obtain a partition of the training points. The first question we will investigate is whether the clustering constructed on a finite sample converges to some "limit clustering" of the whole input space if the sample size tends to infinity. Secondly, if convergence takes place we want to find out whether the limit clustering is actually a useful partition of the input space or not. The methods used to solve these questions originate in perturbation theory, numerical integration theory, and Hilbert-Schmidt operator theory. It will turn out that in all situations where normalized spectral clustering can be successfully applied, its clusterings converge to a limit clustering if we draw more and more data points. In the unnormalized case however, we need some strong additional conditions to guarantee the convergence of spectral clustering, and there exist examples where these conditions are not satisfied. Assuming convergence takes place, we then investigate the form of the limit clustering. It turns out that in both the normalized and the unnormalized case, if convergence takes place, then the limit partition has intuitively appealing properties and accomplishes the overall goal of clustering to group similar points in the same cluster and dissimilar points in different clusters. Some parts of the results in this chapter have been published in von Luxburg et al. (2004b), von Luxburg et al. (2004a).

Chapter III: Here the goal is to develop a framework for large margin classification in metric spaces. We want to find a generalization of linear decision functions for metric spaces and define a corresponding notion of margin such that the decision function separates the training points with a large margin. It will turn out that using Lipschitz functions as decision functions, the inverse of the Lipschitz constant can be interpreted as the size of a margin. In order to construct a clean mathematical setup we isometrically embed the given metric space into a Banach space and the space of Lipschitz functions into its dual space. With this construction it is possible to construct a large margin classifier in the Banach space which also has a geometrical meaning in the metric space itself. We call this classifier the Lipschitz classifier. The generality of our approach can be seen from the fact that several well-known algorithms are special cases of the Lipschitz classifier, among them the support vector machine, the linear programming machine, and the 1-nearest neighbor classifier. Then we proceed to analyze several properties of this algorithm. We

first prove several representer theorems. They state that there always exist solutions of the Lipschitz classifier which can be expressed in terms of distance functions to training points. Then we provide generalization bounds for Lipschitz classifiers in terms of the Rademacher complexities of some Lipschitz function classes. Finally we investigate the relationship of the Lipschitz classifier to other classifiers that can be obtained by embedding the metric space into different Banach spaces. The results of this chapter can also be found in von Luxburg and Bousquet (2003, 2004).

Chapter IV: In the last chapter we investigate connections between statistical learning theory and data compression on the basis of support vector machine (SVM) model selection, cf. von Luxburg et al. (2004c). Inspired by several generalization bounds we construct "compression coefficients" for SVMs which measure the amount by which the training labels can be compressed by a code built from the separating hyperplane. The main idea is to relate the coding precision to geometrical concepts such as the width of the margin or the shape of the data in the feature space. The compression coefficients derived combine well known quantities such as the radius-margin term R^2/ρ^2, the eigenvalues of the kernel matrix, and the number of support vectors. To test whether they are useful in practice we ran model selection experiments on benchmark data sets, and we found that compression coefficients are fairly accurate in predicting the parameters for which the test error is minimized.

1.5 General definitions and notation

In this section we want to introduce those definitions and notations which will be used in all following chapters. Some more specific definitions will be made in the respective chapters. For an overview over most of the symbols used see also the notation table on page 155. The input space will always be denoted by $\mathcal{X}$, the output space by $\mathcal{Y}$. The training points will be named $(x_i, y_i)_{i=1,\ldots,n}$ or $(X_i, Y_i)_{i=1,\ldots,n}$. We use the latter notation if we want to stress the fact that the training points are random variables (following the convention in probability theory to denote random variables with capital letters). The sample size will be always be denoted by n. Curly letters usually denote spaces: $\mathcal{X}$ the input space, $\mathcal{Y}$ the output space, $\mathcal{H}$ a Hilbert space, and $\mathcal{F}$ and $\mathcal{G}$ general function spaces. The symbols $\mathbb{1}$ and $\mathbb{0}$ denote the function (or vector) which is constant 1 or 0, respectively. The characteristic function of a set A is denoted $\mathbb{1}_A$. Dissimilarity functions and metrics will be denoted by d, similarity functions by s and k.

Dissimilarity functions

As in all three chapters similarity and dissimilarity functions play an important role we now want to introduce them formally and discuss several properties which are used in the machine learning and statistics literature to describe (dis)similarity functions.

Consider the following properties of distance functions (cf. Section 13.4 of Mardia et al., 1979, and Section 2 of Veltkamp and Hagedoorn, 2000):

(D1) $d(x, x) = 0$

(D2) $d(x, y) \geq 0$ (non-negativity)

(D3) $d(x, y) = d(y, x)$ (symmetry)

(D4) $d(x, y) = 0 \implies x = y$ (definiteness)

(D5) $d(x, y) + d(y, z) \geq d(x, z)$ (triangle inequality)

Conditions (D1) and (D2) are rather harmless and are in accordance with the intuitive meaning of "distance". A function which satisfies (D1) and (D2) is called a dissimilarity function or a distance function. Already the symmetry condition (D3) is not satisfied for all commonly used dissimilarity functions, a prominent example being the Kullback-Leibler divergence. Condition (D4) deals with invariances in the input space. Different points which have distance 0 from each other usually cannot be discriminated by a learning algorithm and are hence treated as being members of the same equivalence class. For this reason one should be very careful with dissimilarity functions not satisfying (D4). If a dissimilarity function satisfies the axioms (D1)-(D3) and (D5) it is called a semi-metric, and it is called a metric if (D1) - (D5) are satisfied. A space $(\mathcal{X}, d)$ will be called dissimilarity space or metric space, depending on whether d is a dissimilarity or a metric. A matrix $D := (d(x_i, x_j))_{i,j=1,...,n}$ of dissimilarities between points $x_1, ..., x_n \in \mathcal{X}$ is called *dissimilarity matrix* or *distance matrix*, independently of the specific properties of d.

An important issue in the context of distance functions is the question under which conditions a given dissimilarity space $(\mathcal{X}, d)$ can be embedded isometrically into Euclidean spaces $\mathcal{H}$. Here the goal is to find a mapping $\Phi : \mathcal{X} \to \mathcal{H}$ such that $d(x, y) = \|\Phi(x) - \Phi(y)\|$ is satisfied for all $x, y \in \mathcal{X}$. As distances in vector spaces always satisfy all axioms (D1) - (D5), isometrically embedding a dissimilarity space into a vector space is only possible if the dissimilarity function is a metric. This is a necessary condition, but it is not sufficient. A characterization of metric spaces which can be embedded isometrically into Hilbert spaces was given by Schoenberg (1938): A metric space $(\mathcal{X}, d)$ can be embedded isometrically into a Hilbert space if and only if the function $-d^2$ is conditionally positive definite, that is $-\sum_{i,j=1}^{l} c_i c_j d^2(x_i, x_j) \geq 0$ for all $l \in \mathbb{N}$, $x_i, x_j \in \mathcal{X}$, and for all $c_i, c_j \in \mathbb{R}$ with $\sum_i c_i = 0$. Such a metric is called a Euclidean metric. Contrary to embeddings into Hilbert spaces, isometric embeddings into certain Banach spaces can be constructed for arbitrary metric spaces. Such constructions will be discussed in detail in Section 2.

Another trick which is sometimes necessary is to transform a non-metric dissimilarity function into a proper metric. This can for example be done as follows:

From dissimilarities to metrics:

- Let d be a distance function and $x_0 \in \mathcal{X}$ an arbitrary point. Then $\tilde{d}(x,y) := |d(x,x_0) - d(y,x_0)|$ is a semi-metric on $\mathcal{X}$ (cf. Veltkamp and Hagedoorn, 2000).
- Let $(\mathcal{X}, d)$ a finite dissimilarity space such that d is symmetric and definite. Then the distance function

$$\tilde{d}(x,y) := \begin{cases} d(x,y) + c & \text{for } x \neq y \\ 0 & \text{for } x = y \end{cases}$$

  with $c \geq \max_{p,q,r \in \mathcal{X}} |d(p,q) + d(p,r) + d(q,r)|$ is a metric (Theorem 1 in Gower, 1986).
- If D is a dissimilarity matrix, then there exist constants h and k such that the matrices with the elements $\bar{d}_{ij} = (d_{ij}^2 + h)^{1/2}$ $(i \neq j)$ and $\tilde{d}_{ij} = d_{ij} + k$ $(i \neq j)$ are Euclidean (Theorem 7 in Gower, 1986).
- If d is a metric, so are $d + c$, $d^{1/r}$, $d/(d+c)$ for all $c > 0$ and $r \geq 1$ (Theorem 2 in Gower, 1986).
- Let $w : \mathbb{R} \to \mathbb{R}$ a monotonically increasing, continuous function which satisfies $w(x) = 0 \iff x = 0$ and $w(x+y) \leq w(x) + w(y)$. If $d(\cdot,\cdot)$ is a metric, then also $w(d(\cdot,\cdot))$ is a metric. Examples for w are $x/(1+x)$, $\tan^{-1}(x)$ or $\log(x)$ (Section 4.1. in Veltkamp and Hagedoorn, 2000).

Similarity functions

In the context of similarity functions consider the following properties (cf. Sections 13.4 and 14.2.3 of Mardia et al., 1979):

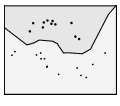
(S1) $s(x,x) > 0$

(S2) $s(x,y) = s(y,x)$ (symmetry)

(S3) $s(x,y) \geq 0$ (non-negativity)

(S4) $\sum_{i,j=1}^{n} c_i c_j s(x_i, x_j) \geq 0$ for all $n \in \mathbb{N}$, $c_i \in \mathbb{R}$, $x_i \in \mathcal{X}$ (positive definiteness)

None of these properties is an undisputable feature of what one intuitively understands under a similarity function, maybe with the exception of (S1). We call any function s which satisfies (S1) a *similarity function* or *affinity function*. As in the case of distances, symmetry is a convenient property, although it is not satisfied in all applications. The non-negativity is not satisfied for two standard examples: correlation coefficients and scalar products. Note however, that bounded similarity



function s can always transformed into non-negative similarity functions by adding a constant offset, that is by considering $\tilde{s}(x, y) := s(x, y) + c$ for some large enough constant c. Moreover, if s is a positive definite similarity function, then $\tilde{s}$ is still at least conditionally positive definite (cf. Schölkopf, 2001). In general, positive definiteness is a very strong requirement which is mainly satisfied by scalar products in Hilbert spaces. Similarity functions which are positive definite will be called *kernel functions*.

Machine learning algorithms are usually designed to deal with either similarities or dissimilarities. In general it is recommended to choose an algorithm which can deal with the type of data we are given, but sometimes it may become necessary to convert similarities into dissimilarities or vice versa. In some situations this can be done without loosing information, especially if the similarities and distances are defined by a scalar product in an Euclidean space. If this is not the case, several heuristics can be invoked. The general idea is to transform a similarity into a dissimilarity function or vice versa by applying a monotonically decreasing function. This is according to the general intuition that a distance is small if the similarity is large, and vice versa. Some ways how such a transformation can be done are listed in the following:

From similarities to dissimilarities:

- If the similarity function is a scalar product in a Euclidean space (i.e., it is positive definite), we can compute the corresponding metric by

$$d(x, y)^2 = \langle x - y, x - y \rangle = \langle x, x \rangle - 2\langle x, y \rangle + \langle y, y \rangle$$

- Assume that the similarity function is normalized, that is $0 \leq s(x, y) \leq 1$ and $s(x, x) = 1$ for all $x, y \in \mathcal{X}$. Then $d := 1 - s$ is a distance function (cf. Gower, 1985, Cox and Cox, 2001).

From dissimilarities to similarities:

- If the given distance function is Euclidean, we can compute a positive definite similarity function by

$$s(x, y) := \frac{1}{2}(d(x, 0)^2 + d(y, 0)^2 - d(x, y)^2)$$

  where 0 is an arbitrary point in $\mathcal{X}$ playing the role of an origin.

- If d is a dissimilarity function, then a non-negative decreasing function of d is a similarity function, for example $s(x, y) = \exp(-d(x, y)^2/t)$ for some parameter t or $s(x, y) = \frac{1}{1-d(x,y)}$.

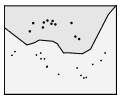
Chapter II

Convergence of Spectral Clustering on Random Samples

In this chapter we will study the problem of clustering from a theoretical point of view. In general, it seems very difficult to define "what clustering is" in a precise mathematical framework. But we will show that even if we cannot answer this general question, there are some questions which emerge naturally once we start to investigate specific clustering algorithms in a probabilistic framework. Perhaps the most important requirement for clustering algorithms will be that they work "consistently": the clustering constructed on finite samples drawn from some underlying distribution converges to a fixed limit clustering of the whole data space when the sample size tends to infinity. We will then investigate the question of consistency in detail for one certain class of algorithms, namely spectral clustering algorithms. It will turn out that for some versions of spectral clustering, convergence always takes place, while for some other versions this is not necessarily the case.

1 Clustering from a theoretical point of view

Intuitively, the "goal of clustering" is to form several groups among a given set of "objects" such that objects in the same group "belong together" and objects in different groups are "different from each other". Clearly, this formulation is too general to allow to describe clustering with one single mathematical framework. Instead, clustering is studied in many different settings and with many different objectives. This is due to the fact that clustering should discover previously unknown "structure" in given data, and the type of structure as well as the type of data varies from application to application. Contrary to classification, where the ultimate goal is to achieve as good results as the optimal Bayes classifier (cf. Section III.1), there is no such indisputable optimal solution for clustering. As a consequence it is difficult to assess the quality of a given clustering, and even more difficult to assess the quality of a given clustering algorithm.



There exist many attempts to formalize the problem of clustering by introducing systems of axioms to be satisfied by a clustering algorithm, see for example Jardine and Sibson (1971), Wright (1973), Hartigan (1975), or Puzicha et al. (2000). Usually, the axioms chosen are rather intuitive, such as independence under reordering or rotation of the sample points. The goal of defining such systems of axioms is to restrict the class of clustering algorithms to those which produce solutions consistent with all axioms. Kleinberg (2003) carries this to an extreme. He investigates a system of only three axioms: scale invariance, richness (every partition should in principle be possible), and consistency (shrinking distances of points within a cluster and expanding distances between points in different clusters should not affect the clustering). Even though all these axioms are intuitively very plausible and seem to be quite innocuous, Kleinberg can then show that these three axioms together are inconsistent. This means that there exists no clustering function that can satisfy all of them simultaneously.

The drawback of all those axiomatic approaches is that even though usually all axioms are rather intuitive, the particular choice of the set of axioms is a bit arbitrary and lacks justification. This emphasizes the fact that it is inherently difficult to formalize the problem of clustering. Consequently, in practice clustering is performed in many different frameworks, with many different objectives, and in most cases with more or less heuristic methods. In the following we want to divide the general question "what clustering is" into several subquestions which might be easier to answer.

1.1 The data space: probabilisic vs. deterministic

One important aspect that has to be clarified is how we assume that our training points have been generated. One approach is to assume that someone just gives us a fixed set of objects. We do not know where they come from or how they were generated, and we have no means of getting additional knowledge on the objects. An example where this situation occurs is image segmentation. We are just given the pixel values of some digital image, and the goal is to partition the image into different regions such as foreground and background. We call this situation the "deterministic setting". The converse assumption is to consider the objects as sample points generated by some unknown, underlying probability distribution P on some data space $\mathcal{X}$. In this case it is in principle possible to gain extra knowledge on the data by drawing more and more sample points. We refer to this framework as "the probabilistic setting". The main difference between both settings is what we consider to be the "full knowledge" about our data. In the first case, the finite sample *is* the full knowledge. In the probabilistic setting, the complete knowledge of the problem means to know the underlying distribution P. In this case, our finite sample only contains some part of the information, and in principle we can gain more information by drawing more and more sample points.

The most important question we have to answer when we want to do clustering is now:

1.2 What is the goal of clustering if we have full knowledge?

Let us first discuss the probabilistic case. A point of view which is often adopted in the probabilistic setting is that clustering should identify high-density regions which are separated from each other by low-density regions. Note that this definition implicitly requires some kind of distance d on the data space $\mathcal{X}$ in order to define what "density" means. A "density" is always taken with respect to some base measure of volume on the space $\mathcal{X}$ which plays the role of a uniform distribution. In $\mathbb{R}^d$ this is simply the Lebesgue measure. A density tells us how the given probability distribution deviates from the uniform one. One way to define a "uniform distribution" in arbitrary spaces is to require some distance function d on the space $\mathcal{X}$ and call a distribution (approximately) "uniform" if it assigns (approximately) the same volume to all ε-balls of the space. Note that this interpretation implies that by defining a metric on the data space we implicitly define the uniform distribution, and hence what we consider to be a totally unclustered space.

Another point of view which requires a similarity function k rather than a distance function is to look at clustering via diffusion processes. We define a diffusion process on the data space such that the transition probability between two points of the space is proportional to the similarity between two points, weighted by the probability distribution on the space: for some point $x \in \mathcal{X}$ and a measurable set $A \subset \mathcal{X}$ we define a transition kernel by $q(x, A) := \int_A k(x, y) dP(y)$. Then we define the diffusion process on $\mathcal{X}$ induced by this transition kernel. The goal of clustering is then to identify two (or several) clusters such that the probability of staying within the same cluster is high, and the probability of transitioning from one cluster to another one is low.

The deterministic case, where the given sample already contains all information we can get, can be seen as a special case of the probabilistic one with known probability distribution. We simply define our data space $\mathcal{X}$ to coincide with the sample, and set the sampling distribution P to be the uniform distribution assigning the weight $1/n$ to each of those points. Now we can use any of the methods above to define what "the goal of clustering" should be, either using a similarity or a dissimilarity function.

There are many more ways of defining what clustering should achieve, and there exists a huge literature suggesting various more or less heuristically derived criteria. Ultimately, it should depend on the application which one we choose, and here we do not want to judge these different definitions. But we want to stress that it is important to *define what we would like to achieve in clustering if we had full knowledge.*

1.3 Clustering with incomplete knowledge

Once we have identified what we want to achieve by clustering in the case of complete knowledge, we have to study how we can approximate this goal if we only have partial knowledge. This situation occurs if we assume a probabilistic setting and are given a finite sample from some larger data space drawn according to some unknown probability measure P. We then have incomplete knowledge insofar, as we only know some points of the (possibly infinite) input space and hence we only know the empirical distribution P_n instead of the true distribution P. Now we want to construct a clustering on the finite sample which comes close to the "true clustering" which we would construct if we knew the whole distribution. Ideally, we would like to minimize a loss function between the true clustering and the one constructed on the finite sample. But as clustering is an unsupervised problem, this is impossible. What we can do instead is to investigate clustering on the level of clustering algorithms. The above questions, placed in the context of a given clustering algorithm, are then the following:

- Do the clusterings constructed by the algorithm on finite samples converge to some limit clustering if we draw more and more sample points?
- Is this limit clustering actually the one we wanted to achieve in the first place?

Surprisingly, even though there exists a huge literature on clustering algorithms, those two questions are very seldom addressed. On the one hand this is due to the fact that studying convergence properties is very difficult for many clustering algorithms. But on the other hand it is also due to the fact that the two questions "What should we do in the complete knowledge case" and "How can we approximate our goal in the finite sample case" are often mixed up and not treated separately. In the following we want to review some algorithms where convergence has been studied.

1.4 Convergence of clustering algorithms

One class of clustering algorithms where convergence is well-established is model-based clustering (e.g., Zhong and Ghosh, 2003). It is assumed that the data points were generated by a mixture of Gaussians (or by some other parametric family of distributions), and the goal is to group those points together which were generated by the same Gaussian. In this setting, clustering reduces to the standard statistical problem of estimating the parameters of the model. The convergence of those statistical procedures is very well studied. Note however, that often model based approaches are too restrictive in practice. In a model-free approach, the analogue procedure would be to use density estimation techniques to identify high-density regions (e.g. Cuevas et al., 2001). Even though density estimation can be carried out consistently, there are two main disadvantages of this approach. Firstly, density

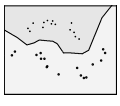
estimation is very difficult in high-dimensional spaces, and many applications definitely are high dimensional (for example in bioinformatics). Secondly, using density estimation to solve clustering violates the principle that we should never solve a more difficult problem as an intermediate step than the one we actually want to solve (cf. Section 1.9 of Vapnik, 1995).

Two very simple, but widely used clustering paradigms are k-centers and single linkage. Both have been studied in terms of their convergence properties.
Given n data points $x_1, ..., x_n$ in a vector space, the k-centers approach constructs k "empirical cluster centers" $c_1, ..., c_k$ that minimize

$$\sum_{i=1}^{n} \min_{j=1,...,k} d(x_i, c_j)^2.$$

In Pollard (1981) it has first been established that under some conditions (essentially, the data space being a Euclidean space and the true centers being unique) the empirical cluster centers converge almost surely to the "true" cluster centers which are the minimizers of $\int \min_{j=1,...,k} |x - c_j|^2 dP(x)$ when the number of training points increases. Subsequently, many authors have achieved similar results under weaker conditions, see Lember (2003) for an overview of the most recent results. Note however, that the k-means algorithm, which is the standard algorithm to construct empirical cluster centers, only approximates the empirical cluster centers and can get stuck in local minima. Hence, the convergence of the empirical cluster centers to the true cluster centers does not imply that the k-means algorithm converges to the true centers, and in fact it does not in general (cf. Bottou and Bengio, 1995). There exist many different algorithms which try to find k-centers empirically on a given finite sample of points, and some of them have theoretical guarantees. For instance, the algorithm in Hochbaum and Shmoys (1985) can be shown to be worse than the optimal solution at most by factor of 2, and the centers constructed by the algorithm proposed in Niyogi and Karmarkar (2000) can be shown to converge to the true centers (but here the true centers are determined by minimizing a different loss function than the least squares loss).

The class of linkage algorithms constructs a hierarchy of clusterings. All linkage algorithms start by assuming that each data point is one cluster. Then they recursively link the two clusters which are "closest" to each other. What "closest" means varies between the linkage algorithms: in single linkage, the distance between two clusters is defined as the smallest distance between its respective points. For complete linkage, the distance between two clusters is defined as the maximum distance between the respective points, and for average linkage it is some kind of average distance between the respective points. In Hartigan (1981), single linkage is shown to be "fractionally consistent": If there exist two disjoint high density regions which are separated by a region with sufficiently small density, then single linkage will asymptotically identify two clusters which come arbitrary close to the true clusters,



and in the limit, the distance between those two sets will be positive. But small clusters might not be correctly detected by single linkage. Hence, single linkage can be used to correctly identify distribution modes separated by deep valleys. In contrast, complete and average linkage are consistently misleading and asymptotically construct clusters which depend on the range of the data and not on the shape of the probability distribution (cf. Hartigan, 1981, 1985). These differences can be explained by the fact that single linkage mainly relies on short distances (i.e., local neighborhoods) while especially complete linkage relies on large distances (i.e., it mainly considers global information). The latter is clearly problematic, as local properties are often more important than the global ones (cf. Section III2.2). Moreover, most heuristic dissimilarity measures are only reliable on a local scale, and not on a global one.

2 Spectral clustering

Above we explained that for a given clustering algorithm there are two main issues to discuss: Does it converge, and if yes, is the limit clustering a reasonable clustering of the data space? In the remaining part of this chapter we want to discuss these questions for a special family of clustering algorithms, namely for spectral clustering. In general, spectral clustering proceeds by analyzing the properties of eigenvectors of certain matrices which are derived from similarity matrices on the data points. Therefore we first have to introduce the type of matrices which are most commonly used in spectral clustering, the so called graph Laplacians.

2.1 Graph Laplacians

In this section we want to introduce the unnormalized and normalized graph Laplacian on finite weighted graphs. Comprehensive treatments of properties of graph Laplacians can be found in in Chung (1997) for the normalized case and in Mohar (1991) for the unnormalized case.

Assume we are given a finite set $\mathcal{X} := \{X_1, ..., X_n\}$ of points and a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ that measures the pairwise similarities between those points. We will interpret the given data as a weighted, undirected graph as follows. The nodes correspond to the data points, and two nodes X_i and X_j are connected by an edge if $k(X_i, X_j) \neq 0$. The weight of the edge is then given by $k(X_i, X_j)$ (cf. Figure 1).

The matrix $K_n := (k(X_i, X_j))_{i,j=1,...,n}$ containing the pairwise similarities between the data points is called the similarity matrix (or affinity matrix). The degree d_i of a node X_i in the graph is the sum of the weights of all adjacent edges, that is $d_i := \sum_{j=1,...,n} k(X_i, X_j)$. The degree matrix D_n is defined as the diagonal matrix containing the degrees $d_1, ..., d_n$ on the diagonal. The *unnormalized graph Laplacian* is defined as the matrix $L_n = D_n - K_n$. There are two common ways of normalizing L_n. Either we consider $L'_n := D_n^{-1/2} L_n D_n^{-1/2}$ or $L''_n := D_n^{-1} L_n$. To

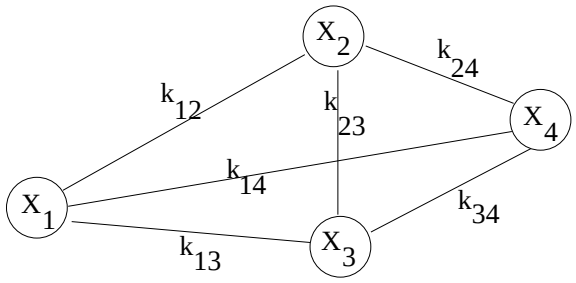


Figure 1: Interpretation of the data as a graph: data points X_i correspond to nodes, and the edge weights are given by the similarities $k_{ij} := k(X_i, X_j)$ of the adjacent points.

ensure that the normalizations are well defined we have to assume in both cases that $d_i > 0$ for all $i = 1, ..., n$. Defining the corresponding normalized similarity matrices $H'_n := D_n^{-1/2} K_n D_n^{-1/2}$ and $H''_n := D_n^{-1} K_n$ we can see that $L'_n = Id - H'_n$ and $L''_n = Id - H''_n$. The matrix H'_n has the advantage of being symmetric (if k is symmetric), while H''_n is a stochastic matrix. Below we will see that there is a close relationship between the eigenvalues and eigenvectors of the four matrices L'_n, H'_n, L''_n, and H''_n. Thus, properties about the spectrum of one of the matrices can be reformulated for the three other matrices as well. In particular, for studying convergence properties of spectral clustering it will make no difference whether we work with the normalization L'_n or L''_n. In the following we will call both L'_n and L''_n *normalized graph Laplacian*. Which of the two matrices we use will depend on the context.

Now we want to summarize some properties of the spectrum of normalized and unnormalized Laplacians. Here we make the assumption that k is non-negative and symmetric, as these are the standard assumptions in spectral clustering (this will be explained below).

Proposition 1 (Spectrum of L_n) *Let k be symmetric and non-negative. Then the following properties hold:*

1. *L_n is positive semi-definite.*
2. *The smallest eigenvalue of L_n is 0 with eigenvector $\mathbb{1}$. If k is strictly positive, the eigenvalue 0 has multiplicity one. In general, the multiplicity of the eigenvalue 0 equals the number of connected components of the graph.*
3. *The largest eigenvalue λ_n of L_n satisfies $\lambda_n \leq 2 \max_{i=1,...,n} d_i$.*

Proof. Part (1) follows directly from the fact that for each $v \in \mathbb{R}^n$

$$v^t L_n v = \sum_{i,j=1}^{n} v_i (v_i - v_j) k(X_i, X_j) = \frac{1}{2} \sum_{i,j=1}^{n} (v_i - v_j)^2 k(X_i, X_j) \geq 0.$$

Part (2): As k is symmetric, the matrix L_n is symmetric. Hence it possesses n real-valued eigenvalues $\lambda_1 \leq ... \leq \lambda_n$ (counted with multiplicity). For the constant one vector $\mathbb{1}$ we have $L_n\mathbb{1} = 0$, hence 0 is an eigenvalue of L_n. By part (1) it is clear that it is the smallest one. The statement about the multiplicity of the eigenvalue 0 can be found for example in Theorem 2.1. in Mohar (1991).

Part (3): It is always the case that the absolute value of the largest eigenvalue is bounded by the operator norm. If we consider the row sum norm $\|L_n\| := \max_{i=1,...,n} \sum_j |l_{ij}|$ (which is the operator norm of L_n with respect to the infinity norm on $\mathbb{R}^n$) it is easy to see that

$$\|L_n\| = \max_{i=1,...,n} (\sum_{j \neq i} |k_{ij}| + |d_i - k_{ii}|) = 2 \max_{i=1,...,n} \sum_{j \neq i} k_{ij} \leq 2 \max_{i=1,...,n} d_i,$$

hence $\lambda_n \leq 2 \max_{i=1,...,n} d_i$. ☺

Now we consider the normalized case. Here we assume that $d_i > 0$ for all i to ensure that the normalized matrices are well-defined.

Proposition 2 (Spectrum of L'_n, L''_n, H'_n, and H''_n) *Let k be symmetric and non-negative. Assume that $d_i > 0$ for each $i = 1, ..., n$. Then the following properties hold:*

1. *$v \in \mathbb{R}^n$ is eigenvector of L'_n with eigenvalue λ iff $D_n^{-1/2}v$ is eigenvector of L''_n with eigenvalue λ.*

2. *v is an eigenvector of L'_n with eigenvalue λ iff v is eigenvector of H'_n with eigenvalue $1 - \lambda$.*

3. *v is an eigenvector of L''_n with eigenvalue λ if v is an eigenvector of H''_n with eigenvalue $1 - \lambda$.*

4. *The eigenvalues of $H_n{}''$ satisfy $-1 \leq \lambda \leq 1$. The largest eigenvalue is 1 with the constant one vector $\mathbb{1}$ as eigenvector. Its multiplicity coincides with the number of connected components in the similarity graph. All eigenvectors with eigenvalue 1 are piecewise constant on the connected components.*

5. *L'_n is positive semi-definite.*

6. *The smallest eigenvalue of L'_n is 0 with eigenvector $\mathbb{1}$. Its multiplicity coincides with the number of connected components in the similarity graph. In particular, if k is strictly positive, it has multiplicity one.*

7. *The eigenvalues λ of L'_n satisfy $0 \leq \lambda \leq 2$.*

Proof. Part (1): Multiplying the eigenvalue equation $L'_n v = \lambda v$ from left with $D_n^{-1/2}$ shows that

$$D_n^{-1/2} L_n D_n^{-1/2} v = \lambda v \iff D_n^{-1} L_n D_n^{-1/2} v = D_n^{-1/2} \lambda v = \lambda D_n^{-1/2} v.$$

Part (2):

$$L'_n v = \lambda v \iff (Id - H'_n)v = \lambda v \iff H'_n v = (1-\lambda)v$$

Part (3):

$$L''_n v = \lambda v \iff (Id - H''_n)v = \lambda v \iff H''_n v = (1-\lambda)v$$

Part (4): Note that H''_n is a stochastic matrix, that is all rows have row sum 1. In particular $H''_n \mathbb{1} = \mathbb{1}$, hence $\mathbb{1}$ is an eigenvector of H''_n with eigenvalue 1. It is well known (e.g., Section 6.2.2. in Brémaud, 1999) that all eigenvalues λ of a stochastic matrix satisfy $|\lambda| \leq 1$. If k is symmetric, then H'_n is symmetric, hence its eigenvalues are real-valued. By Parts (2) and (1) this is also true for H''_n. The statement about the multiplicity of the eigenvalue 1 can be found for example in Section 6.1.1 of Brémaud (1999).
Assume the similarity graph has l connected components. When we order the data points according to their membership to the connected components, the matrix H''_n is a block diagonal matrix with blocks $A_1, ..., A_l$ on the diagonal. Each block A_i corresponds to one connected component of the graph. As all A_i are stochastic matrices, they all have eigenvalue 1 with eigenvector $\mathbb{1}$. As the dimension of the eigenspace is l by the statement about the multiplicity, the eigenspace of eigenvalue 1 of H''_n coincides with the span of the vectors $(\mathbb{1}, \mathbb{0}, ..., \mathbb{0})', ..., (\mathbb{0}, \mathbb{0}, ..., \mathbb{1})'$, where $\mathbb{0}$ and $\mathbb{1}$ correspond to the constant 0 and 1 vectors on the connected components. All these vectors are piecewise constant on the connected components, and so are their linear combinations.

Parts (5) and (6): As in Proposition 1.

Part (7): Follows form Parts (2) and (4).

☺

In the following we will number the eigenvalues of the Laplacians in increasing order, that is $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$. The term "first eigenvalue" hence refers to the smallest eigenvalue λ_1, and the "first eigenvector" to the corresponding eigenvector (analogously also for "second", "third",...). Note that we always count the eigenvalues *with* multiplicities. On the other hand, we will denote the eigenvalues of the normalized similarity matrices H'_n and H''_n by $\mu_1 \geq \mu_2 ... \geq \mu_n$ in decreasing order, hence the "first" eigenvalues correspond to the largest ones. In the light of Proposition 2 this seems to be the natural procedure.

2.2 Spectral clustering algorithms

Spectral clustering is a popular technique going back to graph partitioning algorithms (Fiedler, 1973; Donath and Hoffman, 1973). It has been applied to such diverse problems as load balancing (Van Driessche and Roose, 1995), parallel computations (Hendrickson and Leland, 1995), VLSI design (Hagen and Kahng, 1992), and image segmentation (Shi and Malik, 2000). There exist many different versions of spectral clustering. All algorithms have in common that they use eigenvectors of graph Laplacians or affinity matrices to derive a partition of the sample points. They differ in which matrix they use, which eigenvectors they use, and how they use the eigenvectors exactly. Overviews over some of the algorithms can be found in Weiss (1999) or Verma and Meila (2003). A nice survey on the history of spectral clustering can be found in Spielman and Teng (1996).

Let us explain the principle of spectral clustering by introducing an algorithm which is usually attributed to Ng et al. (2001), but which in fact already existed much earlier (see for example p. 9 in Bolla, 1991, or Sec. 3.4. in Shi and Malik, 2000). Assume we are given n data points $X_1, ..., X_n$ and want to partition them into l clusters. To this end, we compute the l smallest eigenvectors $v_1, ..., v_l \in \mathbb{R}^n$ of the normalized Laplacian L'_n (chosen to be orthogonal in case of repeated eigenvalues) and use them to form the $n \times l$-matrix Y containing the vectors $v_1, ..., v_l$ as columns. We now consider the n rows of the matrix Y to be new representations of our n data points. Formally this means that we identify the data point X_i with a point $Y_i := (y_{i1}, ..., y_{il}) \in \mathbb{R}^l$. We normalize the points Y_i to have Euclidean norm 1 in $\mathbb{R}^l$ and obtain the points $\tilde{Y}_i$. Finally, we use a simple clustering algorithm such as k-means to cluster the points $\tilde{Y}_i$ in $\mathbb{R}^l$.

We already mentioned that the standard assumptions for spectral clustering are that k is symmetric and non-negative. Now we can see why this is the case. The symmetry of k ensures that all eigenvalues of both the normalized and unnormalized Laplacian are real-valued. In particular, this is necessary to be able to order the eigenvalues. As we have already seen above, the non-negativity implies that the Laplacians are positive semi-definite. With these assumptions we make sure that the first eigenvalues and eigenvectors are the ones carrying the important information we need.

On the first glance it is not obvious why the spectral clustering algorithm explained above should work. What is the advantage of the new representation of the data points? To answer this question, for simplicity we assume that we only want to construct two clusters, that is $l = 2$. If the algorithm above works, the new representations $(\tilde{Y}_i)_i$ have to form two distinct clusters in $\mathbb{R}^2$ which are easy to identify by the k-means algorithm. This means that the coordinates of the points in the same cluster should be rather similar to each other. Luckily, in practice this is often the case. The reason is that if our data is clustered, then the second eigenvector v_2 of L'_n turns

out to be "piecewise constant". By this we mean that it has approximately the form $v_2 = (a_1, ..., a_n)'$ with $a_i \approx c_1$ or $a_i \approx c_2$, where c_1 and c_2 are some real numbers with opposite signs, e.g. $c_1 = +1$ and $c_2 = -1$. Then we cluster the data points according to the signs of a_i, that is we put the point X_i in cluster 1 if $a_i > 0$ and in cluster 2 if $a_i < 0$. In the next section we want to give an explanation why this procedure works.

2.3 Why does it work?

There are many articles where theoretical properties of spectral clustering are analyzed and explanations are given why the algorithm works in the finite sample case, for instance Guattery and Miller (1998); Kannan et al. (2000); Meila and Shi (2001); Shi and Malik (2000). We here want to give an explanation in terms of random walks, as it was introduced in Meila and Shi (2001). Assume we want to perform a discrete-time random walk on the similarity graph between our data points. We set the transition probabilities proportional to the similarities between the points: if two points are similar, then the probability that the random walk performs a step between those two points is high. The exact transition probabilities are given by the stochastic matrix H''_n. It can be shown that (normalized) spectral clustering tries to achieve the following: it wants to find two clusters such that the probability of staying within each of the clusters is high and the probability of changing from the one to the other cluster is low. Imagine the extreme case where the data contains two perfect clusters, that is the similarity between points in different clusters is 0. In this case, the probability to change between the two clusters is 0, and hence the stochastic matrix L''_n is not irreducible. It is a well-known fact (e.g., Section 6 of Brémaud, 1999) that in this case, the second eigenvalue of L''_n is 0, and the second eigenvector is constant with opposite signs on both parts. This can also be seen as a consequence of Proposition 2, which a similar argument as the one that will be given in Section 8.2. This means that spectral clustering recovers the correct solution by thresholding the coordinates of the second eigenvector. This reasoning also extends to the case of more than two clusters if we take into account more than the first two eigenvectors.

3 Mathematical background

The algorithm of Ng et al. (2001) we presented above is one out of many different spectral clustering algorithms, but it shows the general structure of most of them. We start with a matrix, either the normalized or unnormalized graph Laplacian. Then we look at its first eigenvectors and cluster our data points according to the values of the coordinates of these eigenvectors. Consequently, to study the question whether spectral clustering algorithms converge for growing sample size n, we have to investigate whether the first eigenvectors of the considered matrix "converge" or not. Studying this question is the main objective of this chapter. Before we can

start with the main work, we will have to recall several concepts about convergence of operators and spectral and perturbation theory. This will be done in the following section.

3.1 Basic spectral theory

In this section we want to recall some basic facts from spectral theory of bounded linear operators. Most of these facts can be found in any functional analysis book, for example in Rudin (1991), Conway (1985), or Taylor (1958). In Chatelin (1983) there is also a nice summary of spectral theory which introduces all we need.

Let E be a real Banach space and $T : E \to E$ a bounded linear operator. An eigenvalue of the operator T is a real or complex number λ such that $Tf = \lambda f$ holds for some element $f \in E$. f is called eigenvector or eigenfunction. Note that λ is an eigenvalue of T iff the operator $(T - \lambda)$ has a non-trivial kernel, that is $(T - \lambda)$ is not injective. The *resolvent* of T is defined as $\rho(T) := \{\lambda \in \mathbb{R};\ (\lambda - T)^{-1}$ exists and is bounded$\}$, and the *spectrum* of T as $\sigma(T) = \mathbb{R} \setminus \rho(T)$. If E is finite-dimensional, every non-invertible operator is not injective. Therefore, λ being in the spectrum implies that λ is an eigenvalue of T. This is not the case if E is infinite-dimensional. Here it can happen that an operator is injective, but nevertheless has no bounded inverse. Thus it is possible that the spectrum contains more than just the eigenvalues of T. In general, the spectrum of a bounded operator is a compact subset of the complex plane, and every possible compact subset can occur as the spectrum of some linear operator.

A part σ_{iso} of $\sigma(T)$ is called *isolated* if there exists some open neighborhood $U \subset \mathbb{C}$ of σ_{iso} such that $\sigma(T) \cap U = \{\sigma_{iso}\}$. Assume that the spectrum $\sigma(T)$ of some bounded operator T on a Banach space E consists of several isolated parts. For each isolated part of the spectrum, a *spectral projection* P_{iso} can be defined with the help of the operational calculus for bounded operators. The spectral projection P_{iso} is a linear projection such that its range is a T-invariant subspace and $\sigma(P_{iso}) = \sigma_{iso}$. For the exact definition of operational calculus and spectral projections we refer to Chapter 5 of Taylor (1958), Section VII.3. of Dunford and Schwartz (1957), Section 2.7. of Chatelin (1983), and Kato (Sec. 3.6.4. of 1966).

We want to stress that a spectral projection can only be defined for *isolated* parts of the spectrum. The reason lies in the fact that the spectral projection corresponding to some part σ_{iso} is defined, in the sense of the operational calculus, as a path integral over a path in the complex plane which encloses σ_{iso} and separates it from the rest of the spectrum.

Let λ be an isolated eigenvalue of $\sigma(T)$. Then the dimension of the range of the spectral projection P_λ corresponding to λ is called the *algebraic multiplicity* of λ. In case of a finite-dimensional Banach space, this corresponds to the multiplicity of the root λ of the characteristic polynomial (this is where the name "algebraic multi-

plicity" comes from). The *geometric multiplicity* is the dimension of the eigenspace corresponding to λ. By the construction of the spectral projections, the eigenspace of an eigenvalue λ is always contained in the range of its spectral projection. Consequently, the geometric multiplicity of an isolated eigenvalue is always smaller than or equal to the algebraic one. If the algebraic and geometric multiplicities of λ are finite and coincide, then the spectral projection is just the projection on the eigenspace of λ. In particular this is the case if the algebraic (and hence geometric) multiplicity is one. Such an eigenvalue is called a *simple* eigenvalue.

There are several ways to split the spectrum into different parts which have different meanings or interpretations. One partition which will be helpful in our context is the following. We define the *discrete spectrum* σ_{d} to be the part of $\sigma(T)$ which consists of all isolated eigenvalues of T with finite algebraic multiplicity, and the *essential spectrum* $\sigma_{\mathrm{ess}}(T) = \sigma(T) \setminus \sigma_{\mathrm{d}}(T)$ as the rest of the spectrum. The essential spectrum is always closed, and the discrete spectrum can only have accumulation points on the boundary to the essential spectrum. The importance of the partition in σ_{d} and σ_{ess} for spectral theory lies in the fact that the essential spectrum cannot be changed by finite-dimensional or compact perturbations of an operator.

Finally we want to mention some basic facts about projections in general, which we will apply to spectral projections later. Given a projection on a one-dimensional subspace, the vector spanning this subspace is only determined up to a change of sign. This is why we want to introduce convergence "up to a change of sign" with the following ad-hoc notation: $v_n - v \overset{+-}{\to} 0$ iff there is a sequence $(a_n)_n$ of signs $a_n \in \{+1, -1\}$ such that $a_n v_n - v \to 0$ (in the appropriate topology which will be clear from the context). Even more sloppy, we write $\|v_n - v\| \overset{+-}{\to} 0$ when we mean $v_n - v \overset{+-}{\to} 0$ in the norm topology.

Proposition 3 (Convergence of projections) *Let $(E, \|\cdot\|)$ be an arbitrary Banach space, $(v_n)_n$ and w vectors in E with norm 1, Pr_n the projection on v_n and Pr the projection on w. Assume that Pr_n converges pointwise to Pr. Then $\|v_n - w\| \overset{+-}{\to} 0$.*

Proof. By the pointwise convergence, we have $\mathrm{Pr}_n w \to \mathrm{Pr}\, w = w$, hence $\|\mathrm{Pr}_n w - w\| \to 0$. As Pr_n has finite range, it is in fact a continuous projection. Thus we can split the space E in a direct sum of the null space $\mathcal{N}(\mathrm{Pr}_n)$ and the range $\mathcal{R}(\mathrm{Pr}_n)$. In particular, we can write the vector w in the form $w = w_n^{\mathcal{R}} + w_n^{\mathcal{N}}$ where $w_n^{\mathcal{R}} \in \mathcal{R}(\mathrm{Pr}_n)$ and $w_n^{\mathcal{N}} \in \mathcal{N}(\mathrm{Pr}_n)$. Moreover, by the definition of Pr_n we know that $w_n^{\mathcal{R}} = a_n v_n$ for some $a_n \in \mathbb{R}$. Thus we have $\|\mathrm{Pr}_n w - w\| = \|a_n v_n - w\| \to 0$. This implies that $|a_n| \|v_n\| - \|w\| \to 0$, and as w and v_n are normalized this means $|a_n| \to 1$. This shows that v_n converges to w up to a change of sign, that is $\|v_n - w\| \overset{+-}{\to} 0$. ☺

3.2 Integral and multiplication operators

In this section we want to recall some basic facts on integral and multiplication operators. Let $(\mathcal{X}, \mathcal{B}, \mu)$ be a probability space and let $k \in L_2(\mathcal{X}\times\mathcal{X}, \mathcal{B}\times\mathcal{B}, \mu\times\mu)$. Then the operator

$$S : L_2(\mathcal{X}, \mathcal{B}, \mu) \to L_2(\mathcal{X}, \mathcal{B}, \mu),\ Sf(x) = \int_{\mathcal{X}} k(x, y)f(y)d\mu(y)$$

is called *integral operator* with kernel k. It is a bounded linear operator with operator norm $\|S\| \leq \|k\|_2 := \int\int k(x, y)dP(x)dP(y)$. In general, this inequality is not tight. Such integral operators form a subset of the compact operators, namely they are Hilbert-Schmidt operators (cf. Section 6.2. in Weidmann, 1980). On the class of Hilbert-Schmidt operators, a special norm is defined, the so called Hilbert-Schmidt norm (sometimes also called the Frobenius norm). This norm is defined as $\|S\|^2_{HS} := \sum_\alpha \|Se_\alpha\|^2$ where $(e_\alpha)_\alpha$ is an orthonormal basis of the Hilbert space $L_2(\mathcal{X})$. It can be shown that the definition of this norm is independent of the choice of the basis. Moreover it is true that $\|S\|^2_{HS} = \|k\|_2$. In particular this means that the operator norm (with respect to the underlying L_2-norm) is always less or equal than the Hilbert-Schmidt norm.

Let $\mathcal{X}$ be a compact space, and k a continuous function. Then the integral operator S can also be defined on the space $(C(\mathcal{X}), \|\cdot\|_\infty)$. There it has operator norm (with respect to the underlying $\|\cdot\|_\infty$-norm) $\|S\| \leq \|k\|_\infty := \sup_{x,y} |k(x, y)|$, and it is also compact.

Let $(\mathcal{X}, \mathcal{B}, \mu)$ be a probability space and let $d \in L_\infty(\mathcal{X}, \mathcal{B}, \mu)$. Define the *multiplication operator*

$$M_d : L_2(\mathcal{X}, \mathcal{B}, \mu) \to L_2(\mathcal{X}, \mathcal{B}, \mu),\ M_d f = fd.$$

This is a bounded linear operator with operator norm $\|M_d\| = \|d\|_\infty$ (cf. Example 2.2 in Section III.2 of Conway, 1985). The function d is called the multiplier function. If d is non-constant, the operator M_d is never compact. The multiplication operator M_d can also be defined on the space $C(\mathcal{X})$ if $\mathcal{X}$ is compact and d continuous. The following proposition recalls some well-known facts on the spectra of several types of operators.

Proposition 4 (Spectrum of some operators)

1. *Spectrum of a compact operator: Let T be a compact operator on a Banach space. Then $\sigma(T)$ is at most countable and has at most one limit point, namely 0. If $0 \neq \lambda \in \sigma(T)$, then λ is an eigenvalue with finite-dimensional eigenspace, and its algebraic and geometric multiplicities coincide.*

2. *Let S be an integral operator as defined above with symmetric kernel function k. Then all eigenvalues of S are real-valued.*

3. *Spectrum of a multiplication operator: For a bounded function $g \in L_\infty(P)$ consider the multiplication operator $M_g : L_2(P) \to L_2(P),\ f \mapsto gf$. M_g is a bounded linear operator whose spectrum coincides with the essential range of the multiplier g (i.e., the smallest interval $[l, u]$ such that $P(g(x) \in [l, u]) = 1$). The same is true if M_g is defined on the space $C(\mathcal{X})$ for some compact $\mathcal{X}$.*

4. *Let A be a bounded and V a compact linear operator on a Banach space. Then $\sigma_{ess}(A + V) = \sigma_{ess}(A)$ (cf. Theorem IV.5.35 in Kato, 1966 and Theorem 9.3 in Birman and Solomjak, 1987).*

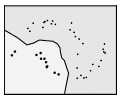
3.3 Some perturbation theory

The main tool to analyze the convergence of eigenvalues and eigenvectors of linear operators is perturbation theory. In the following we want to recall some definitions and facts from perturbation theory for bounded operators. The standard reference for perturbation theory in general is Kato (1966), for perturbation theory in Hilbert spaces we also recommend Birman and Solomjak (1987) and Weidmann (1980), and Bhatia (1997) for finite-dimensional perturbation theory. Many different types of convergence of operators and their consequences for the spectrum are studied in Chatelin (1983). For collectively compact perturbation theory of integral operators we refer to Anselone (1971).

Perturbation theory studies the question whether two operators which are "close" in some sense also have similar spectra. We will be especially interested in the question which type of operator convergence conserves eigenvalues and eigenvectors. In perturbation theory, the convergence of eigenvalues is often discussed in the following terms. $\sigma(T)$ is said to be *upper semi-continuous* if $T_n \to T$ (in some topology to be specified) implies that every converging sequence $(\lambda_n)_n$ with $\lambda_n \in \sigma(T_n)$ converges to some limit point λ which is in the spectrum of T. $\sigma(T)$ is said to be *lower semi-continuous* if $T_n \to T$ (in some topology to be specified) implies that every point in $\sigma(T)$ can be approximated by a sequence $(\lambda_n)_n$ with $\lambda_n \in \sigma(T_n)$.

$\sigma(T)$ is called continuous if it is both upper and lower semi-continuous.
Convergence of eigenvectors is not as easy to define as convergence of eigenvalues. The reason is that if the eigenspace of some eigenvalue has more than one dimension, then it contains infinitely many eigenvectors. Then it is difficult to define convergence of eigenvectors because we would have to specify which ones we are referring to. Instead, one usually studies convergence of the eigenspaces themselves in terms of spectral projections. In case of one-dimensional eigenspaces, Proposition 3 shows how this leads to convergence of the eigenvectors.

In general, to ensure that the spectral properties of a converging sequence of operators are preserved, we need to require rather strong types of convergence of operators, some of which we now want to introduce. For background reading we refer to Section 3.1. of Chatelin (1983).



Definition 5 (Convergence of operators) *Let E be an arbitrary Banach space, and B its unit ball. Let $(S_n)_n$ be a sequence of bounded linear operators on E.*

- $(S_n)_n$ converges pointwise*, denoted by $S_n \xrightarrow{p} S$, if $\|S_n x - Sx\| \to 0$ for all $x \in E$, where $\|\cdot\|$ denotes the norm on E.*
- $(S_n)_n$ converges in operator norm*, denoted by $S_n \xrightarrow{\|\cdot\|} S$, if $\|S_n - S\| \to 0$ where $\|\cdot\|$ denotes the operator norm.*
- $(S_n)_n$ *is called* collectively compact *if the set $\bigcup_n S_n B$ is relatively compact in E (with respect to the norm topology).*
- $(S_n)_n$ converges collectively compactly*, denoted by $S_n \xrightarrow{cc} S$, if it converges pointwise and if there exists some $N \in \mathbb{N}$ such that the operators $(S_n - S)_{n>N}$ are collectively compact.*
- *A sequence of operators* converges compactly*, denoted by $S_n \xrightarrow{c} S$, if it converges pointwise and if for every sequence $(x_n)_n$ in B, the sequence $(S - S_n)x_n$ is relatively compact.*

In general, both operator norm convergence and collectively compact convergence are strong types of convergence. They do not imply each other, but both imply compact convergence. Pointwise convergence is the weakest form of convergence and is implied by all the other ones.

Proposition 6 (Types of convergence)

1. $T_n \xrightarrow{cc} T \implies T_n \xrightarrow{c} T$.
2. $T_n \xrightarrow{\|\cdot\|} T \implies T_n \xrightarrow{c} T$.
3. $T_n \xrightarrow{cc} T \not\Rightarrow T_n \xrightarrow{\|\cdot\|} T$
4. $T_n \xrightarrow{\|\cdot\|} T \not\Rightarrow T_n \xrightarrow{cc} T$

Proof. Proofs for Parts (1) and (2), as well as counterexamples for Parts (3) and (4), can be found in Section 3.2. of Chatelin (1983). ☺

Operator norm convergence is sufficient to ensure the convergence of many spectral properties (Sec. IV.2.6, IV.3 Kato, 1966), but often it is too strong a requirement. This is the reason why in the context of integral operators, the notion of collectively compact convergence has been developed. Even more, it turns out that already compact convergence ensures the convergence of the spectral properties we are interested in. This is very convenient as compact convergence is implied both by collectively compact and operator norm convergence and hence allows to treat both in the same framework.

In general, what we can achieve under favorable conditions is is that *isolated parts* of the spectrum are upper semi-continuous (e.g., Sec. IV.3. of Kato, 1966). If these isolated parts are eigenvalues with finite multiplicity, then we also get lower semi-continuity (Theorems 3.16 and 2.23 Kato, 1966). These results are stated in detail in the following proposition:

Proposition 7 (Perturbation results for compact convergence) *Let E be an arbitrary Banach space and $(T_n)_n$ and T bounded linear operators on E. Let $T_n \xrightarrow{c} T$.*

1. Upper semi-continuity: *Let $\tau \subset \sigma(T)$ be an isolated part of $\sigma(T)$, that is there exists a neighborhood $M \subset \mathbb{C}$ of τ such that $M \cap \sigma(T) = \tau$. Let $\lambda_n \in \sigma(T_n) \cap M$ be a converging sequence with limit point λ. Then $\lambda \in \tau$.*

2. Lower semi-continuity: *Let $\lambda \in \sigma(T)$ be an isolated eigenvalue of T with finite algebraic multiplicity. Then there exists some neighborhood $M \subset \mathbb{C}$ of λ such that for large n, $\sigma(T_n) \cap M = \{\lambda_n\}$, and $(\lambda_n)_n$ converges to λ.*

3. Convergence of spectral projections: *Let P_n and P be the spectral projections associated to a converging sequence $\lambda_n \in \sigma(T_n)$ of isolated eigenvalues with finite multiplicity whose limit point λ is an isolated eigenvalue with finite multiplicity in $\sigma(T)$. Then $P_n \xrightarrow{p} P$.*

4. Convergence of eigenvectors: *Under the conditions of Part (3), if λ is a simple eigenvalue, so are λ_n for n large enough. Then the corresponding eigenfunctions converge up to a change of sign.*

Proof. See Proposition 3.18. and Sections 3.6. and 5.1. in Chatelin (1983), and Proposition 3. ☺

We will also need some results from finite-dimensional perturbation theory where the difference between eigenvalues and eigenprojections is bounded in terms of the operator norm of the perturbation.

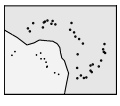
Proposition 8 (Finite-dimensional perturbation results) *Let A and B be two symmetric matrices in $\mathbb{R}^{n\times n}$, and denote by $\|\cdot\|$ the operator norm on $\mathbb{R}^{n\times n}$ (with respect to the Euclidean norm on $\mathbb{R}^n$).*

1. *Denote by $\rho(\sigma(A), \sigma(B))$ the Hausdorff distance between the spectra of A and B. Then*

$$\rho(\sigma(A), \sigma(B)) \leq \|A - B\|.$$

2. *Let $\mu_1 > ... > \mu_k$ be the eigenvalues of A counted without multiplicity, and $\mathrm{Pr}_1, ..., \mathrm{Pr}_k$ the projections on the corresponding eigenspaces. For $1 \leq r \leq k$ define the numbers*

$$\delta_r(A) := \min\{|\mu_i - \mu_j|;\ 1 \leq i < j \leq r+1\}.$$



Assume that $\|B\| \leq \delta_r(A)/2$. *Then for all* $1 \leq l \leq r$ *we have*

$$\|\operatorname{Pr}_l(A+B) - \operatorname{Pr}_l(A)\| \leq 4\frac{\|B\|}{\delta_r(A)}. \tag{1}$$

Both statements are also true with the Hilbert-Schmidt norm instead of the operator norm.

Proof. Part (1) can be found in Section VI.3 of Bhatia (1997), Part (2) in the appendix of Koltchinskii (1998) and in Lemma 5.2. of Koltchinskii and Giné (2000). ☺

4 Relating graph Laplacians to linear operators on $C(\mathcal{X})$

To distinguish between spectral clustering using the normalized or the unnormalized graph Laplacian we introduce the short terms "normalized spectral clustering" and "unnormalized spectral clustering".

To study the convergence of normalized or unnormalized spectral clustering we have to investigate whether the eigenvectors of the normalized or unnormalized Laplacians constructed on n sample points converge if $n \to \infty$. Here we face one technical problem: the size of the Laplacian matrix (both normalized and unnormalized) is $n \times n$, hence it grows if n increases. Similarly, the eigenvectors get longer and longer. The problem is now that to define convergence of operators, one usually requires that the operators are defined on the same space, and the same holds for convergence of vectors. As this is not satisfied in our case, we make the following construction. We relate each Laplacians matrix to some other operator such that all the operators are defined on the same space. Then convergence of these operators is well-defined, and we can study the convergence of their eigenvalues and eigenvectors. In this construction we have to ensure that the spectra of the Laplacians have a close relation to the ones of the operators.

In the unnormalized case, we will proceed by defining a sequence $(U_n)_n$ of operators which will be related to the matrices $(L_n)_n$. Each operator U_n will be defined on the space $C(\mathcal{X})$ of continuous functions on $\mathcal{X}$, independently of n. Moreover, we will ensure that the spectra of L_n and U_n are closely related. Then we can investigate the convergence of U_n and the convergence of its eigenvectors. Finally, we then have to transform this into statements about the eigenvectors of L_n. A similar approach works in the normalized case for L'_n.

In the whole Section 4, we assume that the data space $\mathcal{X}$ is a compact metric space and that the similarity function k is continuous.

4.1 Definition of the operators

In this section we introduce several linear operators on $C(\mathcal{X})$ corresponding to the matrices we are interested in. In general, we will proceed by identifying vectors $(v_1, ..., v_n)' \in \mathbb{R}^n$ with functions $f \in C(\mathcal{X})$ such that $f(X_i) = v_i$, and extending linear operators on $\mathbb{R}^n$ to deal with such functions rather than vectors. Let us start with the unnormalized Laplacian. Recall that L_n is defined as $D_n - K_n$ where D_n is the diagonal matrix containing the degrees $d_i = \sum_j k(X_i, X_j)$ and K_n is the similarity matrix. First we want to relate the degree vector $(d_1, ..., d_n)'$ to some functions in $C(\mathcal{X})$. To this end we define the *true and the empirical degree functions*

$$d(x) := \int k(x, y) dP(y) \in C(\mathcal{X}) \quad \text{and} \quad d_n(x) := \int k(x, y) dP_n(y) \in C(\mathcal{X}). \tag{2}$$

By definition, $d_n(X_i) = \frac{1}{n} d_i$, so the empirical degree function coincides with the degrees of the points X_i up to the scaling factor $1/n$. This factor comes from the hidden $1/n$ factor in the empirical distribution P_n. The function $d_n(\cdot)$ is the natural extension of the discrete degrees, which originally were defined on the data points only, to the whole space $\mathcal{X}$. By the law of large numbers it is clear that for $n \to \infty$, the empirical degree function $d_n(x)$ converges pointwise to the true degree function $d(x)$ almost surely. As we assumed that k is continuous, both d_n and d are continuous functions, and if k is bounded, then d_n and d are bounded.

We want to find an operator acting on $C(\mathcal{X})$ which behaves similar to the matrix D_n on $\mathbb{R}^n$. Let us analyze how the matrix D_n operates on some vector $f = (f_1, ..., f_n)' \in \mathbb{R}^n$. For each i we have $(D_n f)_i = d_i f_i$, that is the value of the vector f at coordinate i is multiplied by the value of d_i. If we now identify $\frac{1}{n} d_i$ with $d_n(X_i)$ and f_i with $f(X_i)$, then $\frac{1}{n} D_n$ can be interpreted as a multiplication operator. The linear operator on $C(\mathcal{X})$ corresponding to the matrix $\frac{1}{n} D_n$ will be the *empirical multiplication operator*

$$M_{d_n} : C(\mathcal{X}) \to C(\mathcal{X}),\ M_{d_n} f(x) := d_n(x) f(x). \tag{3}$$

To obtain its limit operator for $n \to \infty$ we replace the empirical measure P_n by the true one (this will be done properly below) and define

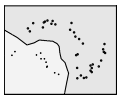
$$M_d : C(\mathcal{X}) \to C(\mathcal{X}),\ M_d f(x) := d(x) f(x). \tag{4}$$

We will call M_d the *true multiplication operator*. Next we have a look at the matrix K_n. Applying it to some vector $f \in \mathbb{R}^n$ yields $(K_n f)_i = \sum_j k(X_i, X_j) f_j$. This will be represented by the *empirical integral operator*

$$S_n : C(\mathcal{X}) \to C(\mathcal{X}),\ S_n f(x) := \int k(x, y) f(y) dP_n(y). \tag{5}$$

Its pointwise limit operator will be

$$S : C(\mathcal{X}) \to C(\mathcal{X}),\ S f(x) := \int k(x, y) f(y) dP(y) \tag{6}$$



and will be called the *true integral operator* (for proper convergence statements see below). Note that compared to the matrix K_n, the operator S_n has an extra scaling factor $1/n$ hidden inside the empirical distribution P_n. Consequently, the operator S_n corresponds to $\frac{1}{n}K_n$.

With these definitions, the operator corresponding to the unnormalized graph Laplacian $\frac{1}{n}L_n$ is the difference between the empirical multiplication and integral operators:

$$U_n : C(\mathcal{X}) \to C(\mathcal{X}),$$
$$U_n f(x) := M_{d_n} f(x) - S_n f(x) = \int k(x,y)(f(x) - f(y))dP_n(y)$$

and has the pointwise limit

$$U : C(\mathcal{X}) \to C(\mathcal{X}),$$
$$Uf(x) := M_d f(x) - Sf(x) = \int k(x,y)(f(x) - f(y))dP(y). \tag{7}$$

Now let us consider the case of the normalized Laplacian. We start with the symmetric normalization L'_n. We have already seen in Proposition 2 that the eigenvalues and eigenvectors of L'_n can be computed from the ones of H'_n. Therefore, no harm will be done by studying the convergence of the eigenvalues and eigenvectors of H'_n instead of L'_n. The matrix H'_n operates on some vector $f = (f_1, ..., f_n)'$ by $(H'_n f)_i = \sum_j \frac{k(X_i,X_j)}{\sqrt{d_i d_j}} f_j$. We can see that this is very similar to the behavior of the unnormalized similarity matrix K_n, the difference being that $k(X_i, X_j)$ is replaced by $k(X_i, X_j)/\sqrt{d_i d_j}$. So we will define the following *normalized empirical and true similarity functions*

$$h_n(x,y) := k(x,y)/\sqrt{d_n(x)d_n(y)}$$
$$h(x,y) := k(x,y)/\sqrt{d(x)d(y)} \tag{8}$$

and introduce the following three integral operators:

$$T_n : C(\mathcal{X}) \to C(\mathcal{X}),\ T_n f(x) = \int h(x,y)f(y)dP_n(y)$$
$$T_n' : C(\mathcal{X}) \to C(\mathcal{X}),\ T_n' f(x) = \int h_n(x,y)f(y)dP_n(y)$$
$$T : C(\mathcal{X}) \to C(\mathcal{X}),\ Tf(x) = \int h(x,y)f(y)dP(y). \tag{9}$$

Note that in case of these operators, the scaling factors $1/n$ which are hidden in P_n and d_n cancel each other (put in different terms, the matrix H'_n already contains a $1/n$ scaling factor, contrary to the case of the matrix K_n in the unnormalized case).

Therefore, contrary to the unnormalized case we do not have to scale the matrices H'_n and H_n with a factor $1/n$. So the operator T'_n corresponds directly to the matrix H'_n, while the operator T_n corresponds to the matrix $H_n := (h(X_i, X_j))_{i,j=1,\ldots,n}$. The reason to introduce T_n and H_n is a technical one. It will be easier to prove that T'_n converges to T in two steps using the operator T_n in between. We will show that T_n and T'_n "get close" and that T_n converges to T.

While we will always use the matrix H'_n and the operator T'_n to prove the convergence of spectral clustering, we will prefer to use the matrix H''_n to investigate some properties of the limit clustering. Therefore, analogously to above we introduce the normalized functions

$$\begin{aligned} g_n(x, y) &:= k(x, y)/d_n(x) \\ g(x, y) &:= k(x, y)/d(x) \end{aligned} \tag{10}$$

and the corresponding integral operators

$$\begin{aligned} &R_n'' : C(\mathcal{X}) \to C(\mathcal{X}),\ R_n''f(x) = \int g_n(x, y)f(y)dP_n(y) \\ &R : C(\mathcal{X}) \to C(\mathcal{X}),\ Rf(x) = \int g(x, y)f(y)dP(y). \end{aligned} \tag{11}$$

Finally, note that according to Section 3.2, all occurring integral operators (S_n, S, T_n, T'_n, T, R''_n, and R) are compact operators if the corresponding kernel functions (k, h_n, h, g_n, and g) are square-integrable. In the following, this will always be the case.

4.2 Relations between the spectra of the operators

The main point about all the constructions above is that they enable us to transfer the problem of convergence of the Laplacian matrices to the problem of convergence of a sequence of operators on $C(\mathcal{X})$. Now we want to establish the connections between the spectra of L_n and U_n, H'_n and ${T_n}'$, and H''_n and R''_n.

Proposition 9 (Spectrum of U_n)

1. *The spectrum of U_n consists of the compact interval* $\mathrm{rg}(d_n)$*, plus eventually some isolated eigenvalues with finite multiplicity. The same holds for U and* $\mathrm{rg}(d)$*.*
2. *If $f \in C(\mathcal{X})$ is an eigenfunction of U_n with arbitrary eigenvalue λ, then the vector $v \in \mathbb{R}^n$ with $v_i = f(X_i)$ is an eigenvector of the matrix $\frac{1}{n}L_n$ with eigenvalue λ.*
3. *Let $\lambda \notin \mathrm{rg}(d_n)$ be an eigenvalue of U_n with eigenfunction $f \in C(\mathcal{X})$, and $v_j := f(X_j)$. Then f is of the form*

$$f(x) = \frac{\frac{1}{n}\sum_j k(x, X_j)v_j}{d_n(x) - \lambda}. \tag{12}$$

4. *If v is an eigenvector of the matrix $\frac{1}{n}L_n$ with eigenvalue $\lambda \notin \mathrm{rg}(d_n)$, then f defined by equation (12) is an eigenfunction of U_n with eigenvalue λ.*

Proof. Part (1): By Proposition 4, the essential spectrum of the multiplication operator M_{d_n} consists of the range of the multiplier function d_n. As S_n is a compact operator, the essential spectrum of $U_n = M_{d_n} - S_n$ coincides with the essential spectrum of M_{d_n} (Part (3) of Proposition 4). Apart from the essential spectrum, an operator can only have some discrete spectrum consisting of isolated eigenvalues with finite multiplicity (Section 3.3).

Part (2): Note that for $v = (f(X_1), ..., f(X_n))'$ we actually have that $U_n f(X_i) = \frac{1}{n}(L_n v)_i$. This immediately shows that if f is an eigenfunction of U_n, then v is an eigenvector of $\frac{1}{n}L_n$ with the same eigenvalue.

Part (3): If λ is an eigenvalue of U_n and f the corresponding eigenfunction, we have that

$$U_n f(x) = f(x)d_n(x) - \frac{1}{n}\sum_j k(x, X_j)f(X_j) = \lambda f(x).$$

For $\lambda \notin \mathrm{rg}(d_n)$ this leads to the continuous function defined in equation (12).

Part (4): Define f as in equation (12). It is well-defined because v is an eigenvector of $\frac{1}{n}L_n$, and f is an eigenfunction of U_n with eigenvalue λ. ☺

This proposition establishes a one-to-one correspondence between the eigenvalues and eigenvectors of $\frac{1}{n}L_n$ and U_n, provided they satisfy $\lambda \notin \mathrm{rg}(d_n)$. The condition $\lambda \notin \mathrm{rg}(d_n)$ is needed to ensure that the denominator of equation (12) does not equal 0. But there is also a deeper reason for this condition which will play an important role later on. Note that the essential spectrum of U_n coincides with $\mathrm{rg}(d_n)$. An eigenvalue with $\lambda \in \mathrm{rg}(d_n)$ is thus part of the essential spectrum and is *not* isolated in the spectrum. As we have already seen in Section 3.3, the convergence of eigenvalues which are not isolated in the spectrum can in principle not be investigated with perturbation theory methods. In the end this will lead to the unsatisfactory situation that convergence of unnormalized spectral clustering can only be asserted if the first eigenvalues of the limit operator U are not contained in the range of the true degree function.

In the normalized case on the other hand, we are in a much more benign situation. Contrary to the operators U_n and U in the unnormalized case, the operators T_n', T_n, and T representing the normalized similarity matrices and their limits are compact integral operators, and no multiplication operators are involved. The advantage is that the essential spectrum of a compact operators consists only of the set $\{0\}$, and all other eigenvalues are isolated and have finite multiplicity – hence they satisfy

everything we need to apply perturbation theory methods. The next proposition shows a one-to-one relationship between the non-zero eigenvalues and eigenvectors of the normalized empirical similarity matrix H'_n and the ones of the operator T'_n.

Proposition 10 (Spectrum of T'_n)

1. *If $f \in C(\mathcal{X})$ is an eigenfunction of T'_n with arbitrary eigenvalue μ, then the vector $v \in \mathbb{R}^n$ with $v_i = f(X_i)$ is an eigenvector of the matrix H'_n with eigenvalue μ.*

2. *Let $\mu \neq 0$ be an eigenvalue of T'_n with eigenfunction $f \in C(\mathcal{X})$, and $v_j := f(X_j)$. Then f is of the form*

$$f(x) = \frac{1}{n\mu} \sum_j k(x, X_j) v_j. \tag{13}$$

3. *If v is an eigenvector of the matrix H'_n with eigenvalue $\mu \neq 0$, then f defined in by equation (13) is an eigenfunction of T'_n with eigenvalue μ.*

Proof. This can either be proved directly as Proposition 9, but it can also be seen as a special case of Proposition 9. Here we just have to observe that normalizing results in the constant degree function $\mathbb{1}$ and hence in this case, $\mathrm{rg}(d_n) = \{1\}$. Moreover, the eigenvalue $\lambda = 1$ of L'_n corresponds to the eigenvalue $\mu = 0$ of T'_n. ☺

Note that combining Propositions 10 and 2 yields the analogous statement for the operator R''_n and matrix H''_n.

5 Convergence in the unnormalized case

Our main result about the convergence of spectral clustering in the unnormalized case is the following theorem:

Theorem 11 (Convergence of unnormalized spectral clustering) *Let $\mathcal{X}$ be a compact metric space, $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a continuous function, $(X_i)_{i \in \mathbb{N}}$ a sequence of data points drawn iid from $\mathcal{X}$ according to the unknown probability distribution P, and L_n the unnormalized Laplacian matrix. Let d be the degree function introduced in equation (2) and $U : C(\mathcal{X}) \to C(\mathcal{X})$ the operator defined in equation (7). Let $\lambda \notin \mathrm{rg}(d)$ be an eigenvalue of U. Then there exists some neighborhood $M \subset \mathbb{C}$ of λ such that for large n, $\sigma(\frac{1}{n} L_n) \cap M = \{\lambda_n\}$, and $(\lambda_n)_n$ converges to λ a.s. Moreover, let $P_n : C(\mathcal{X}) \to C(\mathcal{X})$ be the spectral projection corresponding to $\sigma(U_n) \cap M$, and P the one corresponding to $\lambda \in \sigma(U)$. Then $P_n \xrightarrow{p} P$ a.s. If λ is a simple eigenvalue, then also the eigenvectors converge a.s. up to a change of sign: if v_n is the eigenvector of $\frac{1}{n} L_n$ with eigenvalue λ_n, $v_{n,i}$ its i-th component, and f the eigenfunction of U with eigenvalue $\lambda \notin \mathrm{rg}(d)$, then $\sup_{i=1,\dots,n} |v_{n,i} - f(X_i)| \xrightarrow{+-} 0$ a.s.*

Hence, the clustering constructed on the finite sample converges a.s. to a clustering of the whole data space $\mathcal{X}$.

Let us briefly discuss the assumptions of Theorem 20. The continuity of k and the compactness of $\mathcal{X}$ are technical assumptions. They do not seem indispensable, but they considerably reduce the technical level of the proofs. We will discuss these condition in more detail in Section 7 where we compare several convergence theorems.

An important assumption in Theorem 20 which is not automatically satisfied is that the second eigenvalue has multiplicity one. If this is not the case, Theorem 20 only asserts the convergence of the spectral projections, which then does not imply the convergence of the eigenvectors. But note that if this assumption is not satisfied, spectral clustering will produce more or less arbitrary results anyway, as the second eigenvector is no longer unique. It then depends on the actual implementation of the algorithm which of the infinitely many eigenvectors corresponding to the second eigenvalue is picked, and the result will often be unsatisfactory.

The crucial assumption in Theorem 11 is the condition that the eigenvalue must satisfy $\lambda \notin \mathrm{rg}(d)$. As we already explained above it ensures that the eigenvalue is isolated in the spectrum of U, and hence that perturbation theory can be applied. This fundamental assumption is always needed if one wants to prove convergence of the eigenvectors of U_n with perturbation theory methods. In case this assumption fails, it is *in principle* not possible to investigate the convergence of eigenvectors with perturbation theory methods. The reason is that the spectral projection corresponding to λ then corresponds to the whole part of the spectrum consisting of $\mathrm{rg}(d)$, not only to the eigenvalue λ. We could then prove the convergence of this spectral projection, but this does not allow any conclusions about the convergence eigenvector of λ. At the end of this section we will construct an example where the second eigenvalue indeed lies within $\mathrm{rg}(d)$. This means that there actually exist situations in which Theorem 11 cannot be applied, and hence unnormalized spectral clustering might not converge. Note however, that Theorem 11 only states sufficient conditions for convergence. In principle it is possible that the conditions are not necessary, and that unnormalized spectral clustering also converges if the condition $\lambda \notin \mathrm{rg}(d)$ is not satisfied.

5.1 Proof of Theorem 11

In this section we want to prove Theorem 11. Let us outline the steps of the proof. The first step consists in relating the eigenvectors of $\frac{1}{n}L_n$ to those of U_n. This has already been done in Proposition 9. The next step is to prove that U_n converges to U compactly. This will be done by considering the multiplication operator part M_{d_n} and the integral operator part S_n of U_n separately. It will turn out that the

multiplication operators M_{d_n} converge to M_d in operator norm (but not collectively compactly), and the integral operators S_n converge to S collectively compactly (but not in operator norm). The main ingredient in both parts will be the fact that the class of similarity functions with fixed first argument is not too large: it forms a Glivenko-Cantelli class. This makes it possible to extend some pointwise convergence results to uniform ones. Having established the convergence of the two parts of U_n, we can then show that U_n itself converges compactly to U. By the perturbation theory results of Section 3.3 this ensures the convergence of the spectral projections of isolated eigenvalues with finite multiplicity.

We start with the Glivenko-Cantelli properties.

Proposition 12 ($\mathcal{F}$ Glivenko-Cantelli class) *Let $\mathcal{X}$ be a compact metric space and $k : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ continuous. Then $\mathcal{F} := \{k(x,\cdot);\ x \in \mathcal{X}\}$ is a Glivenko-Cantelli class, that is*

$$\sup_{x\in\mathcal{X}} \left| \int k(x,y)dP_n(y) - \int k(x,y)dP(y) \right| \to 0 \ \text{a.s.}$$

The same holds for the class $\mathcal{F}\times g := \{k(x,\cdot)g(\cdot);\ x \in \mathcal{X}\}$ for an arbitrary $g \in C(\mathcal{X})$.

Proof. As k is a continuous function defined on a compact domain, it is uniformly continuous. In this case it is easy to construct, for each $\varepsilon > 0$, a finite ε-cover with respect to $\|\cdot\|_\infty$ of $\mathcal{F}$ from a finite δ-cover of $\mathcal{X}$. Hence $\mathcal{F}$ has finite $\|\cdot\|_\infty$-covering numbers. Then it is easy to see that $\mathcal{F}$ also has finite $\|\cdot\|_{L_1(P)}$-bracketing numbers (cf. van der Vaart and Wellner, 1996, p. 84). Now the statement for $\mathcal{F}$ follows from Theorem 2.4.1. of van der Vaart and Wellner (1996). The statement about $\mathcal{F}\times g$ can be proved similarly. ☺

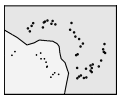
Note that a direct consequence of this proposition and the definition of the degree function in Equation 2 is that the empirical degree function d_n converges uniformly to the true degree function d, that is

$$\|d_n - d\|_\infty = \sup_{x\in\mathcal{X}} |d_n(x) - d(x)| = \left| \int k(x,y)dP_n(y) - \int k(x,y)dP(y) \right| \to 0 \text{ a.s.}$$

Next we establish the convergence of the integral operators S_n:

Proposition 13 (S_n converges collectively compactly to S a.s.) *Let $\mathcal{X}$ be a compact metric space, $k : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ a continuous function, and S_n and S as defined in equation (6). Then $S_n \xrightarrow{cc} S$ almost surely.*

Proof. It is clear that $S_n f(x) \to Sf(x)$ a.s. for each $x \in \mathcal{X}$ by the law of large numbers. By the Glivenko-Cantelli property in Proposition 12, this also holds uniformly over $x \in \mathcal{X}$, hence $S_n \xrightarrow{p} S$ a.s.
Next we prove that the operators $(S_n - S)_n$ are collectively compact a.s. As the



limit operator S is compact itself, it is enough to prove that $(S_n)_n$ are collectively compact a.s. (cf. Anselone, 1971, Sec.4.1). This will be proved using the Arzela-Ascoli theorem. First we fix the random sequence $(X_i)_i$ and hence the random operators $(S_n)_n$. It is easy to see that the operators $(S_n)_n$ are uniformly bounded:

$$\|S_n\| = \sup_{\|f\|_\infty \le 1} \|S_n f\|_\infty = \sup_{\|f\|_\infty \le 1} \sup_{x \in \mathcal{X}} \frac{1}{n} |\sum_j k(x, X_j) f(X_j)| \le \|k\|_\infty.$$

This implies that all functions in $\bigcup_n S_n B$ are uniformly bounded by

$$\sup_{n \in \mathbb{N}, f \in B} \|S_n f\|_\infty \le \|k\|_\infty.$$

To prove that the functions in $\bigcup_n S_n B$ are equicontinuous we have to bound the expression $|g(x) - g(x')|$ in terms of the distance between x and x', uniformly in $g \in \bigcup_n S_n B$. For fixed sequence $(X_i)_{i \in \mathbb{N}}$ and all $n \in \mathbb{N}$ we have

$$\begin{aligned} \sup_{f \in B, n \in \mathbb{N}} |S_n f(x) - S_n f(x')| &= \sup_{f \in B, n \in \mathbb{N}} |\int (k(x, y) - k(x', y)) f(y) dP_n(y)| \\ &\le \sup_{f \in B, n \in \mathbb{N}} \|f\|_\infty \int |k(x, y) - k(x', y)| dP_n(y) \le \|k(x, \cdot) - k(x', \cdot)\|_\infty. \end{aligned}$$

As k is a continuous function defined on a compact domain, it is in fact uniformly continuous. Hence the right hand side gets small uniformly in x and x' whenever the distance between x and x' gets small. This implies the equicontinuity of $\bigcup_n S_n B$. By the Arzela-Ascoli theorem we can now conclude that $\bigcup_n S_n B$ is relatively compact, hence for each fixed sequence $(X_i)_i$ the sequence $(S_n)_n$ is collectively compact. As this reasoning holds for all possible sequences $(X_i)_i$ we get the same statement for the random sequence $(S_n)_n$. ☺

Proposition 14 (U_n converges compactly to U a.s.) *Let $\mathcal{X}$ be a compact metric space, $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ continuous, and U_n and U as defined above. Then $U_n \xrightarrow{c} U$ a.s.*

Proof. We already know that $S_n \xrightarrow{cc} S$ a.s., hence by Proposition 7 we also get $S_n \xrightarrow{c} S$ a.s. For the multiplication operators we have operator norm convergence:

$$\begin{aligned} \|M_{d_n} - M_d\| &= \sup_{\|f\|_\infty \le 1} \|d_n f - df\|_\infty \le \|d_n - d\|_\infty \\ &= \sup_{x \in \mathcal{X}} \left| \int k(x, y) dP_n(y) - \int k(x, y) dP(y) \right|. \end{aligned}$$

The latter converges to 0 a.s. by the Glivenko-Cantelli properties of Proposition 12. As operator norm convergence implies compact convergence by Proposition 7, we also have $M_{d_n} \xrightarrow{c} M_d$ a.s. Finally, it is easy to see that the sum of two compactly

converging operators also converges compactly. Hence, $U_n \xrightarrow{c} U$ a.s. ☺

Now we have collected all ingredients to prove Theorem 11. In Proposition 9 we established a one-to-one correspondence between the eigenvalues $\lambda \notin \mathrm{rg}(d_n)$ of $\frac{1}{n}L_n$ and U_n, and we saw that the eigenvalues λ of U with $\lambda \notin \mathrm{rg}(d)$ are isolated and have finite multiplicity. In Proposition 14 we proved the compact convergence of U_n to U, which according to Proposition 7 implies the convergence of the spectral projections of isolated eigenvalues with finite multiplicity. For simple eigenvalues, this shows the convergence of the eigenvectors up to a change of sign according to Proposition 3. This proves Theorem 11.

5.2 Example for $\lambda \in \mathrm{rg}(d)$

Here we want to construct an example where the second eigenvalue of U satisfies $\lambda \in \mathrm{rg}(d)$ to show that the conditions in Theorem 11 can be violated. Let $\mathcal{X} = [1, 2] \subset \mathbb{R}$ and p be a piecewise constant probability density function on $\mathcal{X}$ with $p(x) = r$ if $4/3 \leq x < 5/3$ and $p(x) = (3 - r)/2$ otherwise, for some fixed constant $r \in [0, 3]$ (e.g., for $r = 0.5$ this density has two clearly separated high density regions). As similarity function we choose $k(x, y) := xy$. It is symmetric, positive on $\mathcal{X} \times \mathcal{X}$, and it has the advantage of being separated (i.e., it is a product of function depending either on x or on y).

The degree function in this case is

$$d(x) = \int_1^2 xyp(y)dy = x\left(\int_1^{4/3} y\frac{3-r}{2}dy + \int_{4/3}^{5/3} yr dy + \int_{5/3}^2 y\frac{3-r}{2}dy\right) = 1.5x$$

(independently of r) and has range $[1.5, 3]$ on $\mathcal{X}$. A function $f \in C(\mathcal{X})$ is eigenfunction with eigenvalue $\lambda \notin \mathrm{rg}(d)$ of U if the eigenvalue equation is satisfied:

$$Uf(x) = d(x)f(x) - x\int yf(y)p(y)dy \stackrel{!}{=} \lambda f(x). \tag{14}$$

Defining the real number

$$\beta := \int yf(y)p(y)dy \tag{15}$$

we can solve Equation (14) for $f(x)$ to obtain $f(x) = \frac{\beta x}{d(x)-\lambda}$. Plugging this into Equation (15) yields the condition

$$1 \stackrel{!}{=} \int \frac{y^2}{d(y) - \lambda}p(y)dy \tag{16}$$

Hence, λ is an eigenvalue of U if Equation (16) is satisfied. For our simple density function p, the integral in this condition can be solved analytically. It can then

been seen (for an illustration see Figure 2) that $g(\lambda) := \int \frac{y^2}{d(y)-\lambda} p(y)dy \stackrel{!}{=} 1$ is only satisfied for $\lambda = 0$, hence the only eigenvalue outside of $\mathrm{rg}(d)$ is the trivial eigenvalue 0.

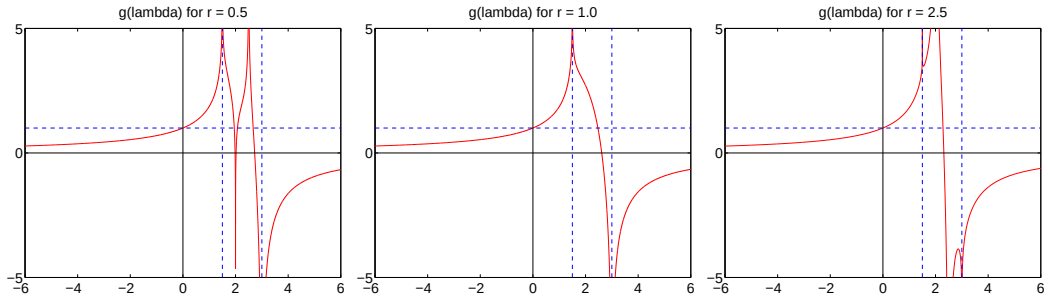


Figure 2: The three figures show $g(\lambda)$ for density parameters $r = 0.5$, $r = 1$, and $r = 2.5$. The vertical dotted lines show the range of $d(x)$, i.e. they mark the essential spectrum. We see that $g(\lambda) = 1$ only for $\lambda = 0$.

6 Convergence in the normalized case

As we have already mentioned earlier, the normalized case is more well-behaved than the unnormalized one as all operators are compact integral operators and there are no disturbing multiplication operators to deal with. With the same methods as above it is possible to prove the convergence of normalized spectral clustering. This is the content of Theorem 15, which will be presented and proved in Section 6.1. Moreover, in the normalized case a related result can be proved under different assumptions using different methods than the ones we already used in the unnormalized case. We will present this approach in Section 6.2.

As we already mentioned before, in the normalized case the eigenvalues and eigenvectors of both L'_n and L''_n are in a one-to-one relationship to the ones of the matrix ${H_n}'$. The smallest eigenvalues of the normalized Laplacians correspond to the largest eigenvalues of ${H_n}'$. In the following we will hence investigate the convergence of the largest eigenvectors of the matrix ${H_n}'$. We recall the definition of the matrix $H_n := (h(X_i, X_j))_{i,j=1,\ldots,n}$. Remember also that contrary to the normalized Laplacians L'_n and L''_n, the matrices ${H_n}'$ and H_n are usually not positive definite (unless the similarity function is positive definite). Its eigenvalues can be in the range between -1 and 1 as we have seen in Proposition 2.

6.1 Approach in $C(\mathcal{X})$

The first convergence theorem in the normalized case is of a similar form as the one in the unnormalized case:

Theorem 15 (Convergence of normalized spectral clustering I) *Let $\mathcal{X}$ be a compact metric space, $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a positive, continuous function, $(X_i)_{i \in \mathbb{N}}$ a sequence of data points drawn iid from $\mathcal{X}$ according to the unknown probability distribution P, and H'_n the normalized empirical similarity matrix. Let $T'_n, T_n, T : C(\mathcal{X}) \to C(\mathcal{X})$ be the integral operators defined in equation (9). Let $\mu \neq 0$ be an eigenvalue of T. Then there exists some neighborhood $M \subset \mathbb{C}$ of μ such that for large n, $\sigma(H'_n) \cap M = \{\mu_n\}$, and $(\mu_n)_n$ converges to μ a.s. Moreover, let $P'_n : C(\mathcal{X}) \to C(\mathcal{X})$ be the spectral projection corresponding to $\sigma(T'_n) \cap M$, and P the one corresponding to $\mu \in \sigma(T)$. Then $P'_n \xrightarrow{p} P$ a.s. If μ is a simple eigenvalue, then also the eigenvectors converge a.s. up to a change of sign: if v_n is the eigenvector of H'_n with eigenvalue μ_n, $v_{n,i}$ its i-th component, and f the eigenfunction of eigenvalue $\mu \neq 0$ of T, then $\sup_{i=1,\ldots,n} |v_{n,i} - f(X_i)| \xrightarrow{+-} 0$ a.s. Hence, normalized spectral clustering converges a.s. to a clustering of the whole data space $\mathcal{X}$.*

The assumptions in Theorem 20 are similar to those of Theorem 11. Again, the continuity of k and the compactness of $\mathcal{X}$ are technical assumptions and will be discussed in Section 7. The positivity assumption on k is one of the standard assumptions of spectral clustering. Together with the compactness of $\mathcal{X}$ it ensures that the degree function d is bounded away from 0, that is $d(x) > l > 0$ for some constant l. This prevents the normalized Laplacian from getting unbounded. For the requirement that the second eigenvalue has multiplicity one, the same remark as in the case of Theorem 11 applies: if this assumption is not satisfied, spectral clustering will produce more or less arbitrary results anyway, as the second eigenvector is no longer unique. It then depends on the actual implementation of the algorithm which of the infinitely many eigenvectors corresponding to the second eigenvalue is picked, and the result will often be unsatisfactory.

The main difference to Theorem 11 is that the disturbing condition $\lambda \notin \mathrm{rg}(d)$ has now been replaced by the rather harmless $\mu \neq 0$. This condition is for example always satisfied if k is a strictly positive function because then the similarity graph on the data points is always connected, and then the eigenvalue 0 has only multiplicity 1 according to Proposition 2.

Note that the eigenvalue $\mu = 0$ of T has been excluded in Theorem 15. The reason is that for compact operators in general, the eigenvalue 0 plays a special role as it might not be isolated. But this also makes sense with regard to the graph Laplacian and can be explained using Theorem 11. Assume that k was normalized from the beginning, that is $d(x) = \mathbb{1}$ (as it is the case for h after the normalization). Then the range of the degree function consists of the set $\{1\}$, and the only eigenvalue of U we cannot make any statement about according to Theorem 11 is the eigenvalue $\lambda = 1$ of U, which corresponds to the eigenvalue 0 of T. Hence excluding the eigenvalue 0 in Theorem 15 is not an artifact of the proof of Theorem 15.

Finally, note that even though Theorem 20 is stated in terms of the second eigenvalue and eigenvector, similar statements are true for higher eigenvalues, and also for spectral projections on finite dimensional eigenspaces with dimension larger than 1.

To summarize, all assumptions in Theorem 20 are already important for successful applications of spectral clustering on a finite sample. Theorem 20 now shows that with no additional assumptions, the convergence of normalized spectral clustering to a limit clustering on the whole data space is guaranteed.

Now we want to prove Theorem 15 similar to Theorem 11. Instead of dealing with multiplication and integral operators, we now only have to work with integral operators. On the other hand, because of the normalization we have to perform some additional steps compared to the unnormalized case.

The first step is again a Glivenko-Cantelli property, this time for the normalized similarity function h.

Proposition 16 ($\mathcal{H}$ Glivenko-Cantelli class) *Let $\mathcal{X}$ be a compact metric space and $k : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ continuous and positive. Then $\mathcal{H} := \{h(x,\cdot);\ x \in \mathcal{X}\}$ is a Glivenko-Cantelli class, that is*

$$\sup_{x\in\mathcal{X}} \left| \int h(x,y)dP_n(y) - \int h(x,y)dP(y)\right| \to 0 \text{ a.s.}$$

Proof. Because of the positivity of k and the compactness of $\mathcal{X}$, d is bounded away from 0, bounded from above, and continuous. Similarly, the function h is well-defined, bounded, and continuous. Hence, the proposition follows by the same proof as Proposition 12. ☺

Note that a direct consequence of this proposition is that

$$\|T_n f - Tf\|_\infty = \sup_x \left| \int h(x,y)f(y)dP_n(y) - \int h(x,y)f(y)dP(y)\right| \to 0 \text{ a.s.},$$

that is the pointwise convergence of T_n to T. Now we can prove the pointwise convergence of the empirical integral operator T_n' to the limit operator T.

Proposition 17 (T_n' converges pointwise to T a.s.) *Under the conditions of Proposition 16, $T_n' \xrightarrow{p} T$ a.s.*

Proof. For arbitrary $f \in C(\mathcal{X})$ we have

$$\|T_n' f - Tf\|_\infty \le \|T_n' f - T_n f\|_\infty + \|T_n f - Tf\|_\infty.$$

The term $\|T_n f - Tf\|_\infty$ on the right hand side converges to 0 a.s. by the Glivenko-Cantelli property of Proposition 16. To prove that $\|T_n' f - T_n f\|_\infty$ converges to 0 a.s.

it is enough to prove that $\sup_{x,y\in\mathcal{X}} |h_n(x,y) - h(x,y)| \to 0$ a.s. By Proposition 12 we know that $\sup_x |d_n(x) - d(x)| \to 0$ a.s., that is for each $\varepsilon > 0$ there exists some N such that for all $n > N$, $|d_n(x) - d(x)| \leq \varepsilon$ for all $x \in \mathcal{X}$. Then

$$|d_n(x)d_n(y) - d(x)d(y)| \leq |d_n(x)d_n(y) - d(x)d_n(y)| + |d(x)d_n(y) - d(x)d(y)| \leq 2(u+\varepsilon)\varepsilon,$$

which implies that

$$|\sqrt{d_n(x)d_n(y)} - \sqrt{d(x)d(y)}| \leq \sqrt{|d_n(x)d_n(y) - d(x)d(y)|} \leq \sqrt{2(u+\varepsilon)\varepsilon}.$$

By the positivity of k and the compactness of $\mathcal{X}$ we know that there exists some constant $l > 0$ such that $k(x,y) \geq l$ for all $x, y \in \mathcal{X}$. This leads to

$$\left| \frac{1}{\sqrt{d_n(x)d_n(y)}} - \frac{1}{\sqrt{d(x)d(y)}} \right| = \left| \frac{\sqrt{d_n(x)d_n(y)} - \sqrt{d(x)d(y)}}{\sqrt{d_n(x)d_n(y)}\sqrt{d(x)d(y)}} \right| \leq \frac{\sqrt{2(u+\varepsilon)\varepsilon}}{l^2}$$

for all $x, y \in \mathcal{X}$. Finally we obtain

$$\sup_{x,y\in\mathcal{X}} |h_n(x,y) - h(x,y)| \leq \sup_{x,y\in\mathcal{X}} \|k\|_\infty \left| \frac{1}{\sqrt{d_n(x)d_n(y)}} - \frac{1}{\sqrt{d(x)d(y)}} \right|$$

which converges to 0 a.s. by the calculation above and the boundedness of k. ☺

To ensure that during the convergence of T_n' to T the eigenvectors are preserved, we will again show collectively compact convergence.

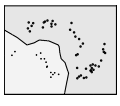
Proposition 18 (T_n' converges collectively compactly to T a.s.) *Under the conditions of Proposition 16, $T_n' \overset{cc}{\to} T$ a.s.*

Proof. As we already know that $T_n' \overset{p}{\to} T$ a.s. and as T is a compact operator, we only have to show that $(T_n')_n$ is collectively compact. This works analogously to the proof of Proposition 13 by the Arzela-Ascoli theorem. The functions in $\bigcup_n T_n' B$ are uniformly bounded as

$$\sup_{f\in B, n\in\mathbb{N}} \|T_n' f\|_\infty \leq \sup_{n\in\mathbb{N}} \|h_n\|_\infty \leq \|k\|_\infty \sup_{n\in\mathbb{N}} \|\frac{1}{d_n}\|_\infty \leq \|k\|_\infty \frac{1}{l}$$

where $l > 0$ is a lower bound on k as in the proof of the previous proposition. Now we have to show that there exists some $N \in \mathbb{N}$ such that $\bigcup_{n>N} T_n' B$ is equicontinuous. As in the proof of Proposition 13 it is easy to see that for fixed $x, x' \in \mathcal{X}$

$$\sup_{f\in B, n>N} |T_n' f(x) - T_n' f(x')| \leq \sup_{n>N} \|h_n(x,\cdot) - h_n(x',\cdot)\|_\infty.$$



Now we have to prove that the right hand side gets small whenever the distance between x and x' gets small.

$$\begin{aligned}
\sup_y |h_n(x,y) - h_n(x',y)| &= \sup_y \left| \frac{k(x,y)\sqrt{d_n(x')} - k(x',y)\sqrt{d_n(x)}}{\sqrt{d_n(x)d_n(x')d_n(y)}} \right| \\
&\leq \frac{1}{l^{3/2}} \sup_y \Big(|k(x,y)\sqrt{d_n(x')} - k(x',y)\sqrt{d_n(x')}| + \\
&\qquad + |k(x',y) + \sqrt{d_n(x')} - k(x',y)\sqrt{d_n(x)}| \Big) \\
&\leq \frac{1}{l^{3/2}} \Big(\|\sqrt{d_n}\|_\infty \|k(x,\cdot) - k(x',\cdot)\|_\infty + \|k\|_\infty |\sqrt{d_n(x)} - \sqrt{d_n(x')}| \Big) \\
&\leq \frac{1}{l^{3/2}} \Big(\sqrt{u}\|k(x,\cdot) - k(x',\cdot)\|_\infty + \|k\|_\infty \Big(|\sqrt{d_n(x)} - \sqrt{d(x)}| + \\
&\qquad + |\sqrt{d(x)} - \sqrt{d(x')}| + |\sqrt{d(x')} - \sqrt{d_n(x')}| \Big) \Big)
\end{aligned}$$

By the uniform continuity of k and d, the terms $\|k(x,\cdot) - k(x',\cdot)\|_\infty$ and $|\sqrt{d(x)} - \sqrt{d(x')}|$ only depend on the distance between x and x'. By the Glivenko-Cantelli properties of Proposition 12 we know that the term $|\sqrt{d_n(x)} - \sqrt{d(x)}| \leq \sqrt{|d_n(x) - d(x)|}$ converges to 0 uniformly in $x \in \mathcal{X}$ a.s. Hence, there exists some N such that for $n > N$ the terms $|\sqrt{d_n(x)} - \sqrt{d(x)}|$ and $|\sqrt{d_n(x')} - \sqrt{d(x')}|$ are smaller than ε independently of x and x'. These arguments together show that $\bigcup_{n>N} T_n' B$ is equicontinuous. This concludes the proof. ☺

Now the proof of Theorem 15 follows as in the unnormalized case: In Propositions 2 and 10 we established a one-to-one correspondence between the non-zero eigenvalues of T_n' and those of L_n'. By the compactness of T, the non-zero eigenvalues of T are isolated and have finite multiplicity. The positivity of k and the compactness of $\mathcal{X}$ imply that there exist some constants u and l such that $\infty > u > d(x) > l > 0$ holds for all $x \in \mathcal{X}$. Hence, proposition 18 proves the collectively compact convergence of T_n' to T. According to Proposition 7 this implies the convergence of the spectral projections of isolated eigenvalues with finite multiplicity. For simple eigenvalues, this shows the convergence of the eigenvectors up to a change of sign according to Proposition 3. This proves Theorem 11. ☺

6.2 Approach in $L_2(\mathcal{X})$

In this section we want to present a theorem which is slightly different from the one above and which is proved by different methods. Here we always assume that k is symmetric and bounded, but we do not need the assumptions that $\mathcal{X}$ is compact and k is continuous. We will consider the same linear operators as above, but define them on the spaces $L_2(P_n)$ and $L_2(P)$ instead of $C(\mathcal{X})$. These spaces have

the advantage of being Hilbert spaces, and the integral operators T_n and T are in then Hilbert-Schmidt operators. As such they have very nice properties, especially concerning their eigenvalues and eigenfunctions. These properties are the key for a series of theorems about convergence of eigenvalues and spectral projections of random matrices approximating integral operators in Koltchinskii and Giné (2000) and Koltchinskii (1998). Among other things, the authors show that the eigenvalues and spectral projections of a similarity matrix converge against those of the limit integral operator. An important tool in the proofs is to construct a Hilbert basis of eigenfunctions. With their help the authors show that the bilinear forms induced by the similarity matrices converge in some sense to the one induced by the integral operator.

To state these results more precisely we first have to introduce some more notation. For a function $f : \mathcal{X} \to \mathbb{R}$ denote its restriction to the sample points $X_1, ..., X_n$ by $\tilde{f}$. If f is square-integrable, then $\tilde{f} \in L_2(P_n)$. Let $h : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a symmetric, measurable similarity function such that $E(h^2(X, Y)) < \infty$. Similar to above we define the integral operators

$$\begin{aligned} T_n &: L_2(P_n) \to L_2(P_n),\ T_n f(x) = \int h(x, y) f(y) dP_n(y) \\ T'_n &: L_2(P_n) \to L_2(P_n),\ T'_n f(x) = \int h_n(x, y) f(y) dP_n(y) \\ T &: L_2(P) \to L_2(P),\ T f(x) = \int h(x, y) f(y) dP(y) \end{aligned} \tag{17}$$

now acting on the spaces $L_2(P_n)$ and $L_2(P)$ instead of $C(\mathcal{X})$. Note that the spaces $L_2(P_n)$ and $\mathbb{R}^n$ are isomorphic. This can be seen by identifying a vector $(v_1, ..., v_n)' \in \mathbb{R}^n$ with the function $f \in L_2(P_n)$ taking values $f(X_i) = v_i$. This also induces an isomorphism between the spaces of linear operators on $\mathbb{R}^n$ and those on $L_2(P_n)$. Especially for the matrix $H_n = (h(X_i, X_j))_{i,j=1,...,n}$ we have

$$(H_n v)_i = \sum_j h(X_i, X_j) v_j = \int h(X_i, y) f(y) dP_n(y) = T_n f.$$

and similarly $(H'_n v)_i = T'_n f$. This isomorphism also extends to the eigenvalues and eigenvectors of H_n and T_n, and H'_n and T'_n, hence they can be identified with each other. Condition $E(h^2(X, Y)) < \infty$ implies that the integral operator T is a Hilbert-Schmidt operator. Because of the symmetry of h, all eigenvalues of T are real-valued, and by the compactness of T, the point 0 is the only accumulation point in the spectrum. Let $\mu_1 \geq \mu_2 \geq ...$ denote the eigenvalues of T counted with multiplicity and $\Phi_1, \Phi_2, ...$ a corresponding set of orthonormal eigenvectors. By the Hilbert-Schmidt properties, the similarity function h can be written in the form $h(x, y) = \sum_{i=1}^{\infty} \mu_i \Phi_i(x) \Phi_i(y)$. To define the spectral projections, we have to take into account that each isolated eigenvalue leads to one spectral projection, no matter what its multiplicity is. Hence we get one spectral projection for each non-zero

eigenvalue, where the eigenvalues this time are counted *without* multiplicity. As spectral projections are only well defined if the eigenvalues are isolated, we only define the first r spectral projections (i.e., the ones belonging to the largest r eigenvalues, counted without multiplicity), where $r \in \mathbb{N}$ may be large but finite. In this way we make sure not to get into trouble if the eigenvalues get too close to the accumulation point 0. For $1 \leq k \leq r$ we denote by Pr_k the spectral projection corresponding to the k-th eigenvalue (counted without multiplicity). As T is a compact operator, the algebraic and geometric multiplicities of its non-zero eigenvalues coincide. Hence, the spectral projections are the projections on the eigenspaces.
Consider a sequence of Hilbert-Schmidt operators with $T_n \to T$, let $\mu_k \neq 0$ be the k-th largest eigenvalue of T. As it is isolated, there exists a neighborhood $M \subset \mathbb{C}$ such that $\sigma(T) \cap M = \{\mu_k\}$. By $\mathrm{Pr}_{k,n}$ we denote the spectral projections corresponding to $\sigma(T_n) \cap M$.
To measure the distance between two countable sets $A = (a_i)_{i\in\mathbb{N}}$, $B = (b_i)_{i\in\mathbb{N}}$, we introduce the minimal matching distance $\delta(A, B) := \inf_\pi \sum_{i=1}^{\infty} a_i - b_{\pi(i)}$, where the infimum is taken over the set of all permutations π of $\mathbb{N}$. Recall the notation $\tilde{f}$ for the restriction of a function $f : \mathcal{X} \to \mathbb{R}$ to the sample $X_1, ..., X_n$.

One of the theorems in Koltchinskii (1998), slightly reworded to fit in our framework, is the following:

Theorem 19 (Koltchinskii) *Let $(\mathcal{X}, \mathcal{B}, P)$ be an arbitrary probability space, $h : \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ a symmetric, measurable similarity function such that $E(h^2(X, Y)) < \infty$ and $E(|h(X, X)|) < \infty$, and T_n and T the integral operators defined above. Let Φ_i be the eigenfunctions of T as defined above. Then:*

1. *$\delta(\sigma(T_n), \sigma(T)) \to 0$ a.s.*
2. *Suppose that $\mathcal{G}$ is a class of measurable functions on $\mathcal{X}$ with a square-integrable envelope G with $\|G\|_{L_2(P)} \leq 1$, that is $|g(x)| \leq G(x)$ for all $g \in \mathcal{G}$. Moreover, suppose that for all $i \in \mathbb{N}$, the set $\mathcal{G}\Phi_i := \{g\Phi_i;\ g \in \mathcal{G}\}$ is a P-Glivenko Cantelli class. Let Pr_k and $\mathrm{Pr}_{k,n}$ be the spectral projections defined above. Then*

$$\sup_{f,g\in\mathcal{G}} \left| \langle \mathrm{Pr}_k(T_n)\tilde{f}, \tilde{g}\rangle_{L_2(P_n)} - \langle \mathrm{Pr}_k(T)f, g\rangle_{L_2(P)} \right| \to 0 \text{ a.s. for } n \to \infty.$$

There are several things to note here. First of all, this theorem states the convergence of the whole spectrum uniformly in the δ-metric. This is much stronger than the pointwise convergence of isolated eigenvalues which we used above. Secondly, the problem that the operators T_n and T are defined on different spaces has been overcome in a different way than the one we chose earlier. Instead of identifying each matrix with an operator on some fixed space (which was $C(\mathcal{X})$ in our case above), Koltchinskii compares bilinear forms. This is possible as the restriction $\tilde{f}$ of each P-integrable function $f : \mathcal{X} \to \mathbb{R}$ on the finite sample is a function in $L_2(P_n)$. This allows to compare the operators T_n and T via the bilinear forms they induce

on $L_2(P_n)$ and $L_2(P)$, and the same holds for the spectral projections. Convergence of the spectral projections can thus be stated in terms of convergence of the induced bilinear forms. Finally, note that the random matrices which are studied in Koltchinskii (1998) are supposed to have diagonal 0, which is not the case for our matrices L'_n. However, under the slightly stronger conditions we pose in Theorem 19 (in particular, $E(h^2) < \infty$), the results of Koltchinskii (1998) also hold with non-zero diagonal values and can be applied to our situation.

We now want to apply this theorem to our situation, where the function h will be the normalized similarity function as defined in equation (8). We can see that Theorem 19 establishes the convergence of the eigenvalues and spectral projections of T_n to those of T. The only remaining step will be to close the gap between T'_n and T_n. This will lead to the following theorem:

Theorem 20 (Convergence of normalized spectral clustering II) *Let $\mathcal{X}$ be an arbitrary metric space, $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a symmetric, bounded, measurable function, $(X_i)_{i \in \mathbb{N}}$ a sequence of data points drawn iid from $\mathcal{X}$ according to the unknown probability distribution P. Let d be the degree function introduced in equation (2) and assume that $d(x) > l > 0$ for all $x \in \mathcal{X}$. Let h be the normalized similarity function from equation 8, and assume that $E(h^2(X, Y))$ and $E|h(X, X)|$ are finite. Let H'_n be the normalized empirical similarity matrix, $T : L_2(P) \to L_2(P)$ the integral operator defined in equation (17), and $\mu \neq 0$ a simple eigenvalue of T. Then there exists some neighborhood $M \subset \mathbb{C}$ of μ such that for large n, $\sigma(H'_n) \cap M = \{\mu'_n\}$, μ'_n is a simple eigenvalue of H'_n, and μ'_n converges to μ a.s. Let Φ'_n and Φ be the corresponding eigenfunctions. They converge up to a change of sign, that is $\|\Phi'_n - \tilde{\Phi}\|_{L_2(P_n)} \stackrel{+-}{\to} 0$ a.s. Hence, normalized spectral clustering converges a.s. to a clustering of the whole data space $\mathcal{X}$.*

In this theorem, we do not require k to be continuous. For this reason, the Glivenko-Cantelli properties of Propositions 12 and 16 cannot be directly applied. This could be fixed by making assumptions on the covering numbers of $\mathcal{F}$, but in this approach we actually do not need the full strength of a Glivenko-Cantelli class. It is enough to show that the empirical and true degree functions approximate each other on the finite sample. This will be showed in the following proposition:

Proposition 21 (Degrees converge uniformly on sample) *Let $k : \mathcal{X} \times \mathcal{X}$ be bounded. Then $\max_{i=1,\ldots,n} |d_n(X_i) - d(X_i)| \to 0$ almost surely.*

Proof. With $M := \|k\|_\infty < \infty$ we have

$$\begin{aligned}\max_{i=1,\ldots,n} |d_n(X_i) - d(X_i)| &= \max_{i=1,\ldots,n} |\frac{1}{n} \sum_{j=1}^{n} k(X_i, X_j) - E_X k(X_i, X)| \\ &\leq \frac{2M}{n} + \frac{n-1}{n} \max_i |\frac{1}{n-1} \sum_{j \neq i} k(X_i, X_j) - E_X k(X_i, X)|.\end{aligned}$$

For fixed $x \in \mathcal{X}$, the Hoeffding inequality gives

$$P\left(|\frac{1}{n-1}\sum_{j\neq i} k(x, X_j) - E_X k(x, X)| > \varepsilon\right) \leq \exp(-M(n-1)\varepsilon^2).$$

As X_i is independent of the random variables X_j with $j \neq i$, the same is true conditionally on X_i if we replace x by X_i. Applying the union bound and taking expectations over X_i shows

$$\begin{aligned} &P\big(\max_{i=1,\dots,n} |\frac{1}{n-1}\sum_{j\neq i} k(X_i, X_j) - E_X k(X_i, X)| > \varepsilon\big) \\ &\leq \sum_{i=1}^{n} P\left(|\frac{1}{n-1}\sum_{j\neq i} k(X_i, X_j) - E_X k(X_i, X)| > \varepsilon \;\middle|\; X_i\right) \\ &\leq n\exp(-M(n-1)\varepsilon^2). \end{aligned}$$

This shows convergence of $\max_{i=1,\dots,n} |d_n(X_i) - d(X_i)| \to 0$ in probability. As the deviations decrease exponentially, the Borel-Cantelli lemma shows that this convergence also holds almost surely. ☺

The next proposition replaces the statement $T_n' \xrightarrow{p} T$ with a weaker statement which is only valid on the finite sample instead of the whole space $\mathcal{X}$:

Proposition 22 (Convergence of $T_n - T_n'$) *Let k be a bounded similarity function. Assume that there exist constants $u > l > 0$ such that $u \geq d(x) \geq l > 0$ for all $x \in X$. Then $\|T_n - T_n'\|_{L_2(P_n)} \to 0$ a.s. and $\|H_n - H_n'\|_2 \to 0$ a.s., where $\|\cdot\|_2$ denotes the operator norm for $n \times n$-matrices with respect to the Euclidean norm on $\mathbb{R}^n$.*

Proof. By the Cauchy-Schwartz inequality,

$$\begin{aligned} \|T_n - T_n'\|^2_{L_2(P_n)} &= \sup_{\|f\|_{L_2(P_n)}} \int \left(\int (h_n(x,y) - h(x,y)) f(y) dP_n(y)\right)^2 dP_n(x) \\ &\leq \sup_{\|f\|_{L_2(P_n)}} \int\int (h_n(x,y) - h(x,y))^2 dP_n(y) \int f^2(y) dP_n(y) \;\; dP_n(x) \\ &\leq \int\int (h_n(x,y) - h(x,y))^2 dP_n(y) dP_n(x) \\ &\leq \max_{i,j=1,\dots,n} |h_n(X_i, X_j) - h(X_i, X_j)|^2 \end{aligned}$$

By a similar argument as the one in the proof of Proposition 18 we can now show that the right hand side converges to 0 a.s., which shows that $\|T_n - T_n'\|$ converges to 0 almost surely. The statement for $\|H_n - H_n'\|$ follows by a similar argument. ☺

Now we can proceed to prove Theorem 20.

Proof of Theorem 20. Theorem 19 states that for large n, there exists a simple isolated eigenvalue μ_n of T_n such that $\mu_n \to \mu$. Let Pr_n and Pr be the corresponding spectral projections. Their induced bilinear forms converge in the sense described in Theorem 19. In particular, choosing $\mathcal{F} = \{\Phi\}$ in Theorem 19 shows that $\langle \mathrm{Pr}_n \tilde{\Phi}, \tilde{\Phi} \rangle \to \langle \mathrm{Pr}\, \Phi, \Phi \rangle$. Denote the eigenvector corresponding to μ_n by Φ_n. We get

$$\langle \Phi_n, \tilde{\Phi} \rangle^2 = \langle \langle \Phi_n, \tilde{\Phi} \rangle \Phi_n, \tilde{\Phi} \rangle = \langle \mathrm{Pr}_n \tilde{\Phi}, \tilde{\Phi} \rangle \to \langle \mathrm{Pr}\, \Phi, \Phi \rangle = \langle \Phi, \Phi \rangle = 1.$$

The eigenfunctions Φ and Φ_n are normalized to 1 in their respective spaces. By the law of large numbers, we also have $\|\tilde{\Phi}\|_{L_2(P_n)} \to 1$ almost surely. Hence, $\langle \Phi_n, \tilde{\Phi} \rangle \to 1$ or -1 implies the convergence of Φ_n to Φ up to a change of sign.

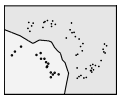
Now we have to compare μ'_n to μ_n and Φ'_n to Φ_n. In Proposition 22 we have seen that $\|{T_n}' - T_n\|_{L_2} \to 0$ and $\|H'_n - H_n\|_2 \to 0$ almost surely. As the eigenvalues and eigenvectors of ${T_n}'$ and T_n correspond to the ones of the finite-dimensional matrices H'_n and H_n, we can now apply finite-dimensional perturbation theory. Part (1) of Proposition 8 immediately shows that $|\mu_n - \mu'_n| \to 0$.

Let Pr'_n be the spectral projection corresponding to μ'_n. We want to use Part (2) of Proposition 8 to prove that $\| \mathrm{Pr}_n - \mathrm{Pr}'_n \| \to 0$ a.s. Choosing $A := H_n$ and $B := H'_n - H_n$, Proposition 8 states that for the spectral projection belonging to the r-th largest eigenvalue of H_n (counted without multiplicity) we have

$$\| \mathrm{Pr}_n - \mathrm{Pr}'_n \| \leq 4 \frac{\|H'_n - H_n\|}{\delta_r(H_n)}$$

where δ_r is defined as in Proposition 8. As we already showed above that $\|H'_n - H_n\|$ converges to 0, we can conclude $\| \mathrm{Pr}_n - \mathrm{Pr}'_n \| \to 0$ if we can ensure that the denominator on the right hand side is bounded away from 0 simultaneously for all n. But this is a consequence of the convergence of the eigenvalues. Intuitively, this is clear by the fact that $\delta_r(H_n)$ measures something like the smallest distance between the first $r+1$ eigenvalues of H_n, and for large n this converges to the smallest distance between the first $r+1$ eigenvalues of T. In detail, assume that the eigenvalue μ is the r-th largest eigenvalue of T (counted without multiplicity). It is clear that $\delta_r(T)$ is a finite, positive number as the $r+1$ largest eigenvalues of T are isolated. We already showed above that the eigenvalues of T_n converge to those of T. This convergence can be made uniformly for the first $r+1$ eigenvalues (as r is a finite number). As a consequence, we obtain $\sup_{n>N} |\delta_r(T) - \delta_r(T_n)| \leq \varepsilon$ for some large N. As we know that $\delta_r(T)$ is bounded away from 0, we can now conclude that also $\delta_r(T_n)$ is bounded away from 0, that is $0 < c < \inf_{n>N} \delta_r(T_n)$ for some constant c and some large N. Hence, $\| \mathrm{Pr}_n - \mathrm{Pr}'_n \|$ converges to 0 almost surely. This implies in particular that

$$\sup_{\|v\| \leq 1} \langle v, (\mathrm{Pr}_n - \mathrm{Pr}'_n) v \rangle \to 0$$



almost surely and thus also

$$\sup_{\|v\|\leq 1} |\langle v, \Phi_n\rangle^2 - \langle v, \Phi'_n\rangle^2| \to 0$$

almost surely. Since $|a^2-b^2| = |a-b||a+b|$, we can conclude that $\|\Phi'_n - \tilde{\Phi}\|_{L_2(P_n)} \stackrel{+-}{\to} 0$ a.s. ☺

Theorem 20 is slightly more general than Theorem 15, but its methods cannot be applied in the case of unnormalized spectral clustering. The reason is that the methods in the proof of Theorem 20 heavily rely on the fact that the operators under investigation are Hilbert-Schmidt operators, which is clearly not the case for the non-compact multiplication operators in the unnormalized case. In the next section we want to discuss the differences between Theorems 15 and 20 in more detail.

7 Mathematical differences between the two approaches

Here we want to discuss which of the assumptions in the convergence theorems are indispensable and which ones are just made for convenience. First we consider Theorems 11 and 15, which both are proved via the collectively compact convergence approach. Let us point out where we used the compactness assumption on $\mathcal{X}$. We defined the limit operators S, M_d, U and T on the space $C(\mathcal{X})$ of continuous functions. If $\mathcal{X}$ is not compact, then the integral operators might not be well defined because the integrals might diverge. Secondly, the Arzela-Ascoli theorem, which was used to prove the collectively compactness of the sequence of integral operators, requires $\mathcal{X}$ to be compact. A third couple of arguments using the compactness was about properties of continuous functions on a compact domain. Here the compactness allows to conclude that the continuous function k is already bounded and uniformly continuous. This was used in several steps, for instance in the proof of the Glivenko-Cantelli properties. Here, the compactness of $\mathcal{X}$ could be replaced by stronger conditions on k directly. All in all, the compactness of $\mathcal{X}$ is an assumption which is used in many places in the collectively compact approach. It does not seem impossible to find a way to prove Theorems 11 and 15 without this assumption, but this would certainly require a lot of technical work. One way how the compactness assumptions might be avoided is to define the operators on $L_2(P)$ instead of $C(\mathcal{X})$, define the empirical operators by discretizing the similarity function rather than the probability measure (as it will be indicated in Section 10), and making extra assumptions on k. In applications it would be desirable not to require compactness of the data space. For example, it is often assumed that the data are generated by Gaussian distributions on $\mathbb{R}^n$, which do not have a compact support. On the other hand, given a finite amount of data it seems sensible to make predictions only for the region of the space where the given training data lives, and this region can be assumed to have a finite diameter and be compact.

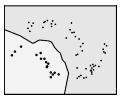
The second assumption in Theorems 11 and 15 is the continuity of the similarity function k. This assumption is more of a technical nature. It makes life much easier, but it is not really necessary. For instance, in the proof of the Glivenko-Cantelli properties, it could be replaced by statements about the covering numbers of the function class $\mathcal{F}$. Moreover, as long as k does not get too wild, the integral operators will still have the same properties as with continuous k, especially they will still have continuous eigenfunctions. In all practical applications, similarity functions are usually assumed to be continuous, hence this requirement does not restrict the application of the theorems.

Note that for the convergence of the eigenvectors it is not necessary to assume that k is symmetric or positive. In spectral clustering however, both assumptions are necessary: symmetry ensures that all eigenvalues are real-valued and hence can be ordered, and positivity ensures the positive semi-definiteness of the Laplacian L_n. Together, both assumptions ensure that the first eigenvectors of L_n are the relevant ones for spectral clustering.

In the normalized case we also assumed that the degree function is bounded away from 0. This seems to be a perfectly reasonable assumption, as otherwise the behavior of the normalized similarity function gets rather unpredictable even on a finite sample.

Now let us turn to the assumptions in Theorem 20. The assumption about the symmetry of k is inherited from Koltchinskii (1998), where it was needed to apply the classical perturbation inequalities of Lidskii and Wielandt. For non-symmetric matrices, similar results are difficult to obtain. Also the boundedness of k is an essential requirement which cannot be relaxed so easily. For instance, it is an essential ingredient in the proof of Proposition 21 which cannot be removed. Two assumptions which are not needed in the Koltchinskii approach are the continuity of k and the compactness of $\mathcal{X}$.

An important difference between both approaches is the fact that the limit operator in the collectively compact approach is defined on the space $C(\mathcal{X})$, while it is on $L_2(P)$ in the Koltchinskii approach. One first thing to note is that even in the space $L_2(P)$, all eigenfunctions of the integral operator T are continuous. This means that the sets of eigenvalues and eigenfunctions are the same in both cases. A disadvantage of $L_2(P)$ is that the functions in $L_2(P)$ are not pointwise defined. This is an unpleasant property as we are interested in the values of the functions at particular points. There seems to be no fundamental reason why one should prefer the space $L_2(P)$ instead of using $C(\mathcal{X})$ – the reasons why this is necessary in the Koltchinskii approach are due to the technical details of the proofs. Even more, for "nice" similarity functions as the Gaussian kernel the eigenfunctions are also "nice", for example differentiable. As there are reasons to believe that the discrete Laplacian matrices are closely related to the continuous Laplace operator on



$\mathbb{R}^n$ (Bousquet et al., 2004) or the Laplace-Beltrami operator on manifolds (Belkin, 2003), the space $C(\mathcal{X})$ of all continuous functions might still be too general to discover the relevant properties of spectral clustering. It might be desirable to work in some space of differentiable functions instead, where the norm on this space should be defined such that it is related to the graph Laplacian. Ideally, what we would like to achieve is to find a space that plays the same role for the graph Laplacians as the reproducing kernel Hilbert spaces play for kernel matrices. This will be an interesting topic for future research.

8 Interpretation of the limit partitions

In the sections above we showed that normalized spectral clustering on finite samples converges to some fixed limit clustering of the whole data space. Now we want to investigate whether this limit clustering partitions the data space $\mathcal{X}$ in a reasonable way. We start by showing that in an idealized setting where the data contains two "true" clusters, the limit clustering is able to recover these clusters. Then we will have a look at the general case.

8.1 An idealized clustering setting

Let us take a step back and reflect what we would like to achieve with spectral clustering. The overall goal in clustering is to find a partition of $\mathcal{X}$ into two (or more) disjoint sets $\mathcal{X}_1$ and $\mathcal{X}_2$ such that the similarity between points in the same set is high while the similarity between points from different sets is low. In this section we want to investigate whether this goal is achieved by spectral clustering if we have optimal conditions. To this end we introduce the "idealized clustering setting": We assume that the partition $\mathcal{X} = \mathcal{X}_1 \cup \mathcal{X}_2$ is an ideal partition of the space, that is the similarity function k satisfies $k(x_1, x_2) = 0$ for all $x_1 \in \mathcal{X}_1$ and $x_2 \in \mathcal{X}_2$, and $k(x_i, x_i') > 0$ for $x_i, x_i' \in \mathcal{X}_1$ or $x_i, x_i' \in \mathcal{X}_2$. Moreover, as in the last section we assume that k is continuous and $\mathcal{X}$ compact.

Now we want to study the form of the limit operators of spectral clustering under these idealized assumptions. Here, considering the operators on $L_2(P)$ is simpler than on $C(\mathcal{X})$, hence we start with the former. Let $\mathcal{X} = \mathcal{X}_1 \cup \mathcal{X}_2$ be a partition of the space $\mathcal{X}$ into two disjoint, measurable sets such that $P(\tilde{\mathcal{X}}_1 \cap \tilde{\mathcal{X}}_2) = 0$. As σ-algebra on $\mathcal{X}_i$ we use the restrictions $\mathcal{B}_i := \{B \cap \mathcal{X}_i;\ B \in \mathcal{B}\}$ of the Borel σ-algebra $\mathcal{B}$ on $\mathcal{X}$. Define the measures P_i as the restrictions of P to $\mathcal{B}_i$. Now we can identify the space $L_2(\mathcal{X}, \mathcal{B}, P)$ with the direct sum $L_2(\mathcal{X}_1, \mathcal{B}_1, P_1) \oplus L_2(\mathcal{X}_2, \mathcal{B}_2, P_2)$. Each function $f \in L_2(\mathcal{X})$ corresponds to a tuple $(f_1, f_2) \in L_2(\mathcal{X}_1) \oplus L_2(\mathcal{X}_2)$, where $f_i : \mathcal{X}_i \to \mathbb{R}$ is the restriction of f to $\mathcal{X}_i$. Let $S : L_2(P) \to L_2(P)$ be a linear operator. It can be identified with the matrix $\begin{pmatrix} S_{11} & S_{12} \\ S_{21} & S_{22} \end{pmatrix}$ acting on $L_2(\mathcal{X}_1, \mathcal{B}_1, P_1) \oplus L_2(\mathcal{X}_2, \mathcal{B}_2, P_2)$. In the idealized setting, the diagonal operators S_{ij} are the 0-operators and S corre-

sponds to $\begin{pmatrix} S_{11} & 0 \\ 0 & S_{22} \end{pmatrix}$.

In the case where we consider the linear operators on $C(\mathcal{X})$ rather than $L_2(P)$, the situation is a bit more complicated. The reason is that in general, functions (f_1, f_2) in $C(\mathcal{X}_1) \oplus C(\mathcal{X}_2)$ do not correspond to continuous functions on $\mathcal{X}$ as f_1 and f_2 cannot automatically be "glued together" continuously at the boundaries between $\mathcal{X}_1$ and $\mathcal{X}_2$. But we will see that in the idealized setting, this problem can be circumvented. This will be a consequence of the following proposition.

Proposition 23 (Two cases in idealized setting) *Let $(\mathcal{X}, dist)$ be a compact metric space and k continuous and non-negative. Let $\mathcal{X} = \mathcal{X}_1 \cup \mathcal{X}_2$ be a disjoint partition of $\mathcal{X}$ such that $k(x_1, x_2) = 0$ for all $x_1 \in \mathcal{X}_1$ and $x_2 \in \mathcal{X}_2$, and $k(x_i, x_i') > 0$ for $x_i, x_i' \in \mathcal{X}_1$ or $x_i, x_i' \in \mathcal{X}_2$. Then either the degree function d satisfies $\min_{x \in \mathcal{X}} d(x) = 0$, or $\mathcal{X}_1$ and $\mathcal{X}_2$ are two disjoint connected components of $\mathcal{X}$ (with respect to the topology on $\mathcal{X}$ induced by the metric) with positive distance.*

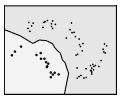
Be careful here not to mix up the topological connectedness of the space $\mathcal{X}$ (which is independent of k) and the idealized setting which states that the similarity between certain parts is 0.

Proof. As $\mathcal{X}$ is compact and k continuous, k is in fact uniformly continuous: for all $\varepsilon > 0$ there exists some $\delta > 0$ such that $\text{dist}(x_1, x_2) \leq \delta$ implies $|k(x_1, y) - k(x_2, y)| \leq \varepsilon$ for all $y \in \mathcal{X}$. Consider $x_1 \in \mathcal{X}_1$ and $x_2 \in \mathcal{X}_2$. For $y \in \mathcal{X}_1$, by the assumptions we know that $k(x_2, y) = 0$, hence by non-negativity $k(x_1, y) \leq \varepsilon$. Analogously, $y \in \mathcal{X}_2$ implies $k(x_1, y) = 0$ and $k(x_2, y) \leq \varepsilon$. Together, for all $y \in \mathcal{X}$ we have $\max\{k(x_1, y), k(x_2, y)\} \leq \varepsilon$, that is $\|k(x_1, \cdot)\|_\infty \leq \varepsilon$ and $\|k(x_2, \cdot)\|_\infty \leq \varepsilon$. Consequently, $d(x_1) \leq \varepsilon$ and $d(x_2) \leq \varepsilon$.

We can conclude that if for all $\delta > 0$ there exist $x_1 \in \mathcal{X}_1$ and $x_2 \in \mathcal{X}_2$ such that $\text{dist}(x_1, x_2) \leq \delta$, then for all $\varepsilon > 0$ there exist points $x \in \mathcal{X}$ such that $d(x) \leq \varepsilon$. By the compactness of $\mathcal{X}$ then $\min_x d(x) = \inf_x d(x) = 0$.

☺

This proposition states that if we assume that our data space is ideally clustered, then either there are points where the degree function is 0 or the clusters are well separated from each other. In case that $\mathcal{X}_1$ and $\mathcal{X}_2$ are disconnected in $\mathcal{X}$,this has some very nice consequences: it implies that $C(\mathcal{X})$ is isomorphic to $C(\mathcal{X}_1) \oplus C(\mathcal{X}_2)$. One direction of this isomorphism is trivial: each continuous function $f \in C(\mathcal{X})$ gives rise to two continuous functions $f_i := f|_{\mathcal{X}_i}$ on the two sets $\mathcal{X}_i$. But the other way round usually fails. This is due to the fact that "glueing together" two arbitrary functions f_i will usually not lead to a continuous function on $\mathcal{X}$. But in the special case that the $\mathcal{X}_i$ are disconnected from each other, this will be no problem. A function whose parts are continuous on the connected components of the space automatically is continuous on the whole space (there is nothing where the functions have to be glued together). This means that under the given conditions we have



$C(\mathcal{X}) \simeq C(\mathcal{X}_1) \oplus C(\mathcal{X}_2)$.

In case $d(x) = 0$, the range of the degree function has the form $[0, a]$ for some real number a, and in the idealized setting the second eigenvalue of L_n is $\lambda = 0$ (as the similarity graph has two connected components by the idealized assumptions). Hence, the second eigenvalue lies within the essential spectrum of the limit operator, and nothing can be said about the convergence of unnormalized spectral clustering. Moreover, normalized spectral clustering is not defined if $d(x) = 0$ as the normalization is not well defined (we would have to divide by 0 at some point).

Hence, in both the normalized and unnormalized case, the condition $d(x) = 0$ implies that no convergence statements can be derived in our framework. In case that $\mathcal{X}_1$ and $\mathcal{X}_2$ are two disjoint connected components of $\mathcal{X}$, both normalized and unnormalized spectral clustering are well defined and the convergence statements of Theorems 11, 15, and 20 apply. In the following we want to investigate the form of the limit operators in this case more closely.

8.2 Normalized limit operator on $L_2(P)$ in the idealized case

In this section it will be more convenient to consider the normalized similarity matrix H_n'' instead of H_n' as it is a stochastic matrix. Recall that the corresponding integral operators R_n'' and R are defined as in equation (11).

We have already seen above that under the given assumptions, the space $L_2(\mathcal{X}, \mathcal{B}, P)$ is isomorphic to the direct sum $L_2(\mathcal{X}_1, \mathcal{B}_1, P_1) \oplus L_2(\mathcal{X}_2, \mathcal{B}_2, P_2)$. Each function $f \in L_2(\mathcal{X})$ corresponds to a tuple $(f_1, f_2) \in L_2(\mathcal{X}_1) \oplus L_2(\mathcal{X}_2)$, where $f_i : \mathcal{X}_i \to \mathbb{R}$ is the restriction of f to $\mathcal{X}_i$. The operator R can now be identified with the matrix $\begin{pmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{pmatrix}$ acting on $L_2(\mathcal{X}_1, \mathcal{B}_1, P_1) \oplus L_2(\mathcal{X}_2, \mathcal{B}_2, P_2)$. We denote by d_i the restriction of d to $\mathcal{X}_i$ and by g_{ij} the restriction of g to $\mathcal{X}_i \times \mathcal{X}_j$. With these notations, the operators R_{ij} for $i, j = 1, 2$ are defined as

$$R_{ij} : L_2(\mathcal{X}_j) \to L_2(\mathcal{X}_i),\ R_{ij} f_j(x) = \int g_{ij}(x, y) f_j(y) dP_j(y).$$

Now we will assume that we are in the idealized case described above, that is $k(x_1, x_2) = 0$ for all $x_1 \in \mathcal{X}_1$ and $x_2 \in \mathcal{X}_2$, and $k(x_i, x_i') > 0$ for $x_i, x_i' \in \mathcal{X}_1$ or $x_i, x_i' \in \mathcal{X}_2$. This means that the operators R_{ij} for $i \neq j$ are 0, and then R has the form $\begin{pmatrix} R_{11} & 0 \\ 0 & R_{22} \end{pmatrix}$. Similar to Proposition 1.4, R then has eigenvalue 1 with multiplicity 2, and the corresponding eigenspace is spanned by the functions $(\mathbb{1}, \mathbb{0})$ and $(\mathbb{0}, \mathbb{1})$. Hence, all eigenfunctions corresponding to eigenvalue 1 are piecewise constant on the sets $\mathcal{X}_1, \mathcal{X}_2$, and the eigenfunction orthogonal to the function $(\mathbb{1}, \mathbb{1})$ has opposite sign on both sets. This means that thresholding the second eigenfunction will recover the true clustering $\mathcal{X}_1 \cup \mathcal{X}_2$.

Note that we can see the operator R as as describing a diffusion process on $\mathcal{X}$. The function g can be interpreted as a Markov transition kernel, and then the operator R describes a Markov diffusion process on $\mathcal{X}$. Hence, the limit clustering partitions the space into two sets such that diffusion only takes place within the sets, but not between them.

A similar reasoning also applies to the finite sample case. Here the block structure of the matrix H_n'' has already been investigated in Weiss (1999) and Ng et al. (2001). The discrete equivalent to a diffusion process is a random walk on the sample. We have already described the random walks interpretation of spectral clustering in Section 2.3. With the notations from above, we split the finite sample space $\{X_1, ..., X_n\}$ into the two sets $\mathcal{X}_{i,n} := \{X_1, ..., X_n\} \cap \mathcal{X}_i$, and define the operators $R_{ij,n}$ by

$$R_{ij,n} : L_2(\mathcal{X}_{j,n}) \to L_2(\mathcal{X}_{i,n}),\ R_{ij,n} f_j(x) = \int g_{ij,n}(x,y) f_j(y) dP_{j,n}(y).$$

According to the random walk interpretation, spectral clustering in the finite case is looking for a partition of the sample space such that the probability of staying within the same cluster is large while the probability of going from one cluster into another one is low, that is $R_{11,n}$ and $R_{22,n}$ are "large" and $R_{12,n}$ and $R_{21,n}$ are "small". In the special case $R_{12,n} = R_{21,n} = 0$, the true partition will be recovered.

So a similar interpretation of the partition constructed by spectral clustering holds both in the finite and in the limit case. We can interpret the operator R as describing a diffusion process on $\mathcal{X}$, and the goal is to partition the space into two sets such that diffusion mainly takes place within the sets, but not between them.

8.3 Normalized limit operator in $C(\mathcal{X})$ in the idealized case

Under the additional constraints that $\mathcal{X}$ is compact and k continuous, we can even make stronger statements. Again we assume that the partition $\mathcal{X} = \mathcal{X}_1 \cup \mathcal{X}_2$ is an ideal partition of the space, that is the similarity function k satisfies $k(x_1, x_2) = 0$ for all $x_1 \in \mathcal{X}_1$ and $x_2 \in \mathcal{X}_2$, and $k(x_i, x_i') > 0$ for $x_i, x_i' \in \mathcal{X}_1$ or $x_i, x_i' \in \mathcal{X}_2$. Moreover we assume that $\mathcal{X}_1$ and $\mathcal{X}_2$ are two disjoint connected components of $\mathcal{X}$ (we already discussed above that otherwise, convergence statements are impossible). As we have already seen that $C(\mathcal{X}) = \mathcal{C}(\mathcal{X}_1) \oplus \mathcal{C}(\mathcal{X}_2)$ in this case, we can proceed as in the space $L_2(P)$ and decompose R into the four operators R_{ij}. Additionally to the statements in the $L_2(P)$ case we can now get even stronger results.

Proposition 24 (Structural convergence in the normalized case) *In the idealized clustering setting on a disconnected space, and under the conditions of Theorem 15 we have $R_{ij,n} \xrightarrow{cc} R_{ij}$ almost surely, and the eigenvalues and eigenfunctions converge analogously to Theorem 15.*

Proof. This can be proved analogously to Theorem 15. ☺

This statement is much sharper than the convergence statement of R_n to R as it shows that for any fixed partition of $\mathcal{X}$, the *structure* of the operators is preserved when taking the limit. This means that a partition that has been constructed on the finite sample such that the diffusion between the two sets is small also keeps this property when we take the limit for $n \to \infty$.

8.4 Unnormalized limit operator in the idealized case

In this section we make the same assumptions as in Section 8.3. As above, we then have $C(\mathcal{X}) \simeq C(\mathcal{X}_1) \oplus C(\mathcal{X}_2)$. We compose the limit operator U into the four operators $(U_{ij})_{i,j=1,2}$ defined as

$$U_{ii} : L_2(\mathcal{X}_i) \to L_2(\mathcal{X}_i),\ U_{ii} f_i(x) = d_i(x) f_i(x) - \int k_{ii}(x,y) f_i(y) dP_i(y)$$
$$U_{ij} : L_2(\mathcal{X}_j) \to L_2(\mathcal{X}_i),\ U_{ij} f_j(x) = -\int k_{ij}(x,y) f_j(y) dP_j(y) \qquad (\text{ for } i \neq j).$$

We see that the off-diagonal operators U_{ij} for $i \neq j$ only consist of integral operators, whereas the multiplication operators only appear in the diagonal operators U_{ii}. Thus the operators U_{ij} for $i \neq j$ can also be seen as diffusion operators, and the same interpretation as in the normalized case is possible. If there exists a partition such that $k(x_1, x_2) = 0$ for all $x_1 \in \mathcal{X}_1$ and $x_2 \in \mathcal{X}_2$, then the second eigenfunction is constant on both parts (due to Proposition 1), and thresholding this eigenfunction will recover the "true" partition. Moreover, on the space $C(\mathcal{X})$ all four operators $U_{ij,n}$ converge compactly to U_{ij} almost surely. This means that the isolated parts of the spectra and the corresponding spectral projections converge. But here we run again into the same problem as in the Theorem 1: we do not know whether the eigenvalues of the diagonal operators U_{ii} are isolated or not, and we need the same assumptions as in Theorem 1 to conclude the convergence of the eigenspaces.

Thus, also in the unnormalized case the goal of spectral clustering is to find partitions such that the norms of the off-diagonal operators is small and the norms of the diagonal operators are large. This holds both in the discrete case and in the limit case, but only if the second eigenvalue of U is not inside the range of the degree function.

8.5 The general case

In practice, the similarity function k will usually be irreducible, that is there will exist no partition such that $k(x_1, x_2) = 0$ for all $x_1 \in \mathcal{X}_1$ and $x_2 \in \mathcal{X}_2$, and $k(x_i, x_i') > 0$

for $x_i, x_i' \in \mathcal{X}_1$ or $x_i, x_i' \in \mathcal{X}_2$. In the non-idealized case, the normalized operators R_{12} and R_{21} will not vanish. Then the goal will be to find a partition such that the norms of R_{12} and R_{21} are as small as possible, while the norms of R_{ii} should be reasonably large. If we find such a partition, then the operators $\begin{pmatrix} R_{11} & R_{12} \\ R_{21} & R_{22} \end{pmatrix}$ and $\begin{pmatrix} R_{11} & 0 \\ 0 & R_{22} \end{pmatrix}$ are close in operator norm. The difference between ideal case and non-ideal case can be measured in terms of the operator norms of the off-diagonal operators.

$$\left\| \begin{pmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{pmatrix} - \begin{pmatrix} T_{11} & 0 \\ 0 & T_{22} \end{pmatrix} \right\| = \left\| \begin{pmatrix} 0 & T_{12} \\ T_{21} & 0 \end{pmatrix} \right\| =$$

$$= \sup_{\|(f_1,f_2)\|_{(\mathcal{X},\mathcal{X})} \leq 1} \|T_{12} f_2 + T_{21} f_1\|_{L_2(\mathcal{X})} \leq \|T_{12}\|_{(\mathcal{X}_1,\mathcal{X}_2)} + \|T_{21}\|_{(\mathcal{X}_2,\mathcal{X}_1)}.$$

Here subscript $(\mathcal{X}_1, \mathcal{X}_2)$ indicates that the corresponding norm is the operator norm for the operator defined from $\mathcal{X}_1$ to $\mathcal{X}_2$.

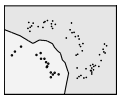
With perturbation theory as in Proposition 8 we can then measure how different the eigenvalues and eigenspaces in the idealized case and the non-idealized case are. In particular, if the difference is small, then the partition constructed by the non-idealized operator R will be approximately the same as the one constructed by the idealized operator $\begin{pmatrix} R_{11} & 0 \\ 0 & R_{22} \end{pmatrix}$, which is the partition $\mathcal{X}_1 \cup \mathcal{X}_2$.

A similar argument also holds in the unnormalized case.

9 Consequences for applications

9.1 Normalized or unnormalized Laplacian?

There is an ongoing debate whether for practical applications it is better to use the normalized or the unnormalized graph Laplacian. Articles using the normalized one include Van Driessche and Roose (1995), Shi and Malik (2000), Kannan et al. (2000), Ng et al. (2001), Meila and Shi (2001), while Barnard et al. (1995); Guattery and Miller (1998) use the unnormalized one. In cases where the degrees d_i of the different points are approximately constant (for example if similarities are based on a k-nearest neighbor procedure as in Belkin and Niyogi (2003b)), then there is not much difference in both approaches, but it does make a difference if the degrees have very different sizes. Comparing both approaches, Van Driessche and Roose (1995) and Weiss (1999) came to the conclusion that the normalized version should be preferred. On the other hand, there is a recent study (Higham and Kibble, 2004) which advocates for the unnormalized version in case only the second eigenvector is used.



The question whether one should use the normalized or the unnormalized Laplacian also arises in different applications of graph Laplacians to machine learning tasks, e.g. in the field of semi-supervised learning (Belkin and Niyogi, 2003b; Zhu et al., 2003; Zhou and Schölkopf, 2004). Here it also seems that the normalized Laplacian obtains better results than the unnormalized one.

Our convergence theorems support the conjecture that normalized Laplacian work at least as good as unnormalized ones. We showed that in the normalized case, convergence of spectral clustering is always guaranteed under the standard assumptions of spectral clustering, while we could not show this in the unnormalized case. On the other hand, if convergence takes place, then both the normalized and the unnormalized spectral clustering converge to an intuitively appealing limit clustering.

9.2 Basic sanity checks for the constructed clustering

In case one wants to use unnormalized spectral clustering, it makes sense to try to find out whether the assumptions of Theorem 11 are satisfied for the given data or not. As we do not know P, we cannot do this exactly, but we can try to make a sanity-check at least. To this end, we estimate the range of the degree function by the interval $[\min_{i=1,\ldots,n} d_i, \max_{i=1,\ldots,n} d_i]$. Then we check whether the second eigenvalue is inside or close to the boundaries of this interval or not. If this is the case, the result of unnormalized spectral clustering should be treated with care. If the second eigenvalue is far away from the empirical range, then we can hope that this will also be the case for the limit operator, and we know that then convergence holds.

According to the diffusion interpretation, it also seems possible to to construct a criterion to evaluate the goodness of the partition achieved by spectral clustering. For a good partition, the off-diagonal operators $R_{12,n}$ and $R_{21,n}$ should have a small norm compared to the norm of the diagonal matrices $R_{11,n}$ and $R_{22,n}$, which is easy to check in practical applications. It will be a topic for future investigations to work out this idea in detail.

10 Convergence of spectra of kernel matrices: why an often cited result does not apply

The techniques we used above to prove the convergence of the spectral properties of graph Laplacians can similarly be applied to prove that the eigenvalues and eigenvectors of kernel matrices $\frac{1}{n}K_n$ converge to those of the corresponding integral operator S. The term "kernel matrix" refers to a similarity matrix for a positive definite similarity function k, as it is used in kernel algorithms such as support vector machines. The convergence of spectra of kernel matrices has already attracted some

attention, cf. Williams and Seeger (2000), Shawe-Taylor et al. (2002), and Bengio et al. (2003). In general, the case of kernel matrices is a special case of the results obtained in Koltchinskii (1998) and Koltchinskii and Giné (2000), where even the distributions of the eigenvalues and spectral projections are investigated. This solves the convergence of spectral properties of kernel matrices completely.

However, all the mentioned studies about the convergence of kernel matrices also cite results from numerical integration theory as evidence for the convergence of eigenvalues of kernel matrices, especially Theorem 3.4 of Baker (1977) is frequently cited. In this section we want to show that Theorem 3.4 of Baker (1977) *does not apply* to the case of random kernel matrices. The main reason is that Theorem 3.4 of Baker is proved in a deterministic setting, and the proof does not carry over to the random case. This also explains why we had to prove the convergence of the spectral properties of the normalized graph Laplacian "by foot" as in the collectively compact approach or by using results of Koltchinskii (1998). The goal of this section is to explain this in detail.

Theorem 3.4 of Baker (1977) deals with the Nyström method of numerical integration. Let us first explain how this method works. For simplicity we assume (as Baker does) that $\mathcal{X}$ is a compact real interval, say $\mathcal{X} = [0, 1]$, and $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ a continuous similarity function. As above, consider the integral operator

$$S : C(\mathcal{X}) \to C(\mathcal{X}),\ Sf(x) = \int k(x, y) f(y) dP(y).$$

The aim of the Nyström method is to approximate this integral operator numerically such that the eigenvalues of the approximation converge to the true eigenvalues of S. There are two ways to perform this approximation.

The first approximation can be done by using a quadrature rule J_n. Instead of computing the integral analytically, we choose some points $x_1, ..., x_n$ in $\mathcal{X}$ and replace the integral $\int g(x)dP(x)$ by a sum $J_n(g) := \sum_{i=1,...,n} w_i g(x_i)$, where w_i are given weights depending on the quadrature rule and $g \in C(\mathcal{X})$ an arbitrary function. Clearly, a valid quadrature rule has to satisfy $J_n(g) \to \int g dP$ for all $g \in C(\mathcal{X})$. The simplest quadrature rule is the Riemann sum in case of a uniform distribution P, where the points x_i are chosen equidistant on $[0, 1]$ and the weights w_i are given by $1/n$. With the quadrature method, we discretize the operator S to the operator

$$\tilde{S}_n : C(\mathcal{X}) \to C(\mathcal{X}),\ \tilde{S}_n f(x) = \sum_{j=1,...,n} w_j k(x, x_j) f(x_j).$$

It can be shown similarly to Proposition 10 that the eigenvalues of $\tilde{S}_n$ are the same as the ones of the matrix $K_n W_n$, where W_n is the diagonal matrix containing the weights w_i of the quadrature rule on the diagonal, and $K_n = (k(x_i, x_j))_{i,j=1,...n}$ with the x_i chosen above. So the question is whether the eigenvalues of $K_n W_n$ converge

to the ones of S or not. For deterministic quadrature rules, Baker proves that they do, by using the second discretization we will describe below. This is also plausible by the results we already obtained in the last sections. As a special case consider the Riemann sum. Here we have $w_i = 1/n$ for all i, and hence $K_n W_n = \frac{1}{n} K_n$. We already showed that the eigenvalues of $\frac{1}{n} K_n$ converge to the ones of S (even though we proved this for random points X_i, but it is even simpler in the case of deterministic x_i).

The other way to approximate S is not to discretize the integral operator, but the kernel function k. Here we partition the space $\mathcal{X}$ into n sets $A_1, ..., A_n$, denote the characteristic functions of A_i by $\mathbb{1}_{A_i}$, for each i choose one representative point $x_i \in A_i$, and define

$$k_n(x, y) := \sum_{i,j=1,...,n} k(x_i, x_j) \mathbb{1}_{A_i}(x) \mathbb{1}_{A_j}(y).$$

The function k_n is piecewise constant on each of the sets $A_i \times A_j$, and its value on $A_i \times A_j$ is the value of $k(x_i, x_j)$ at the two representative points. Let $\hat{S}_n$ be the discretized integral operator where k has been replaced by k_n:

$$\hat{S}_n : C(\mathcal{X}) \to C(\mathcal{X}),\ \hat{S}_n f(x) = \int k_n(x, y) f(y) dP(y).$$

Both ways of discretizing are very similar. We choose representative points in $\mathcal{X}$ and evaluate the integral at these representative points. Note however that in the first method, we only discretize with respect to y (via the discretization of the integral), while in the second method we discretize in x and y. The trick in Baker (1977) is now to connect both discretizations with each other. For a partition $A_1, ..., A_n$ of the space $\mathcal{X}$, $w_j := P(A_j)$, and $x_i \in A_i$ we consider the quadrature rule given by the x_i and w_i, and the discretized kernel with the x_i and A_i. It can be shown that then the eigenvalues of the quadrature rule operator $\tilde{S}_n$ converge to the ones of S if $\sup_{x,y \in \mathcal{X}} |k_n(x, y) - k(x, y)| \to 0$. A necessary condition for the latter is in particular that $\max_{i=1,...,n} \operatorname{diam}(A_i) \to 0$ a.s., where diam denotes the diameter of a bounded set.

The connection to the convergence of the eigenvalues of kernel matrices is now as follows. Consider the *random* quadrature rule which approximates an integral with respect to P by the integral with respect to the empirical distribution P_n, that is we approximate the integral operator S by $\tilde{S}_n f(x) = \frac{1}{n} \sum_j k(x, X_j) f(X_j)$, where X_i are the iid data points drawn from P. By the law of large numbers, this is clearly a valid integral rule, and the weights w_i are given by $1/n$. By the discussion above, the eigenvalues of $\tilde{S}_n$ are the ones of the matrix $K_n W_n$, which is simply the matrix $\frac{1}{n} K_n$ we are interested in. The proof described in Baker now proceeds by showing that $\sup_{x,y \in \mathcal{X}} |k_n(x, y) - k(x, y)| \to 0$. In case of deterministic quadrature

rules this is possible, but in our random case we come upon a problem. To define the appropriate function k_n we have to construct sets A_i such that $P(A_i) = \frac{1}{n}$ and such that the randomly drawn points X_i are in the sets A_i (note that in general the sets A_i will not be connected even in the simple case $\mathcal{X} = [0, 1]$, which might already hint that there might be problems). To prove that k_n converges to k in the sense above, we additionally have to show that $\max_{i=1,\ldots,n} \operatorname{diam}(A_i) \to 0$. Here the problem is the max in the expression. We have to construct the sets A_i in such a way that $X_i \in A_i$, $P(A_i) = \frac{1}{n}$, and $\max_{i=1,\ldots,n} \operatorname{diam}(A_i) \to 0$. I believe that this is impossible in general, in particular if our space $\mathcal{X}$ has a difficult geometric structure. In the deterministic case, however, the quadrature rule and the sets A_i are just chosen such that all these properties hold (consider the Riemann sum as a simple example).
Conversely, another idea is to start with a kernel function k_n which we can control and then construct the corresponding quadrature rule. For example, we can choose the sets A_i as the neighborhoods of the data points X_i, that is we define

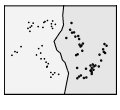
$$A_i := \{x \in \mathcal{X};\ \operatorname{dist}(x, X_i) < \operatorname{dist}(x, X_j) \text{for all j } \neq i\}.$$

Then it is possible to prove that $\max_i \operatorname{diam}(A_i) \to 0$ a.s., and consequently also that $\sup_{x,y} |k_n(x, y) - k(x, y)| \to 0$ a.s. But the quadrature rule corresponding to k_n is now the rule $J_n(g) = \sum w_j g(X_j)$ where $w_j = P(A_j)$. Hence, $\tilde{S}_n$ has the eigenvalues given by the matrix $K_n W_n$. So by proving that k_n converges to k we can prove that the eigenvalues of $K_n W_n$ converge to the ones of S. But we are actually interested in the eigenvalues of the matrix $\frac{1}{n} K_n$. So to prove that the eigenvalues of our kernel matrices converge to the ones of the integral operator we additionally have to show that $\|K_n W_n - \frac{1}{n} K_n\| \to 0$ a.s. It can be seen that a necessary condition for this is that $\max_i |w_i - \frac{1}{n}| = \max_i |P(A_i) - P_n(A_i)|$ converges to 0. But this, I believe, is wrong. We can see that the condition $\max_i |P(A_i) - P_n(A_i)| \to 0$ has a Glivenko-Cantelli flavor, and related problems have been studied in the context of data based histogram rules and nearest neighbor density estimation (cf. Lugosi and Nobel, 1996). We need to show that the difference between the empirical and the true measure of a certain family of sets converges to 0 uniformly over all sets. In general, this is only true if more and more data points end up being in all of the sets, which is not the case here (each A_i contains only one data point by definition).

To conclude, the proof of Theorem 3.4 of Baker (1977) cannot easily be carried over from deterministic quadrature rules to random ones. We do not claim that it is impossible, we just point out the difficulties one would have to solve on the way. So before one can cite Theorem 3.4 of Baker (1977) in the context of random kernel matrices, one first has to explain how these difficulties can be overcome.

11 Discussion

The take-home message of this chapter is the following: under the standard assumptions of spectral clustering (k is continuous, symmetric, and positive) all three



convergence Theorems 11, 15, and 20 apply. In the normalized case, we then can conclude that spectral clustering converges, while in the unnormalized case, convergence can only be guaranteed under the additional assumption that the interesting eigenvalues are not in the range of the degree function. In case that convergence takes place, the limit clustering achieves the objective of clustering, namely to divide the space such that the within-similarity in the clusters is high and the between-similarity is low.

There are many open questions related to spectral clustering which have not been addressed in our work so far. The most obvious one is what happens in the unnormalized setting if the second eigenvalue is within $\mathrm{rg}(d)$. As Theorem 11 only contained sufficient conditions it is not clear whether in this case convergence can also take place or not. To solve this question one would either have to construct an example where the algorithm does definitely not convergence, or to prove with other means that we always get convergence even if the condition $\lambda \in \mathrm{rg}(d)$ is satisfied.

Also very important is the question about the speed of convergence and the concentration of the limit results. Results in this direction would enable us to make confidence predictions about how close the clustering on the finite sample is to the "true" clustering proposed by the limit operator.

This immediately raises a second question: Which relations are there between the limit clustering and the geometry of the data space? For certain similarity functions such as the Gaussian kernel $k_t(x, y) = \exp(-\|x - y\|^2/t)$, it has been established that there is a relationship between the operator T and the Laplace operator on $\mathbb{R}^n$ (Bousquet et al., 2004) or the Laplace-Beltrami operator on manifolds (Belkin, 2003). Can this relationship also be extended to the eigenvalues and eigenfunctions of the operators?

There are also more technical questions related to our approach. The first one is the question which space of functions is the "natural" space to study spectral clustering. The space $L_2(P)$ is a large space and is likely to contain all eigenfunctions we might be interested in. On the other hand, for "nice" similarity functions the eigenfunctions are continuous or even differentiable, thus $L_2(P)$ might be too general to discuss relevant properties such as relations to continuous Laplace operators. Moreover, we want to use functions which are pointwise defined, as we are interested in the value of the function at specific data points. But of all spaces, the functions in L_p-spaces do not have this property.

Another question concerns the type of convergence results we should prove. In this work, we fixed the similarity function k and considered the limit for $n \to \infty$. As a next step, the convergence of the limit operators with respect to some kernel parameters, such as the kernel width t for the Gaussian kernel, can be studied as in the works of Bousquet et al. (2004) and Belkin (2003). But it seems more appropriate

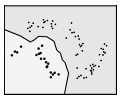
to take limits in t and n simultaneously. This might reveal other important aspects of spectral clustering, for example how the kernel width should scale with n.

Chapter III

Classification in Metric Spaces Using Lipschitz Functions

The goal of classification is to learn how to distinguish between objects from two or several categories. Contrary to clustering, classification is a supervised learning task where we are given a set of training points consisting of patterns with labels. These training points are used to construct a classifier which assigns class memberships to all patterns of the data space. To achieve this, we require some kind of information about the relationships between the training points. In most cases, this information either consists of similarities or dissimilarities between pairs of training points. As a lot of attention has already been paid to similarity based classifiers such as support vector machines, we want to focus on distance based classification.

Rather than performing classification on a metric space directly, our approach will be to embed the given metric space into some vector space and use the additional linear structure to construct a good classifier. In the first sections we will introduce the problem of classification in general (Section 1) and review several ways of dealing with distances from an embedding point of view (Section 2). In Section 3 we introduce the concept of a large margin classifier and explain why it is desirable to use them. In the main part of this chapter, our goal will be to develop a framework for large margin classification in metric spaces. We want to find a generalization of linear decision functions for metric spaces and define a corresponding notion of margin such that the decision function separates the training points with a large margin. This will be accomplished by embedding the given metric space isometrically into a certain Banach space called Arens-Eells space, and simultaneously embedding the space of Lipschitz functions on the metric space into the dual of the Arens-Eells space (Section 5). Performing large margin classification in the Arens-Eells space will result in an algorithm called the Lipschitz classifier. It will turn out that using Lipschitz functions as decision functions, the inverse of the Lipschitz constant can be interpreted as the size of a margin (Section 6). To analyze the resulting algorithm, we then prove several representer theorems (Section 7). They state that there always exist solutions of the Lipschitz classifier which can be expressed in terms of



distance functions to training points. In Section 8 we provide generalization bounds for Lipschitz classifiers in terms of the Rademacher complexities of some Lipschitz function classes. As the embedding in the Arens-Eells space is not the only possible isometric embedding of a metric space into a Banach space, we investigate relationships between the Lipschitz classifier and classifiers constructed by different embeddings in Section 9. We will show that certain embeddings can be seen as a special case of the Lipschitz classifier, and also several well-known algorithms such as the support vector machine, the linear programming machine, and the 1-nearest neighbor classifier can be interpreted in the framework of the Lipschitz classifier.

1 The standard classification framework

Let $\mathcal{X}$ be some arbitrary set of patterns and $\mathcal{Y}$ some discrete set denoting class memberships of patterns, for instance $\mathcal{Y} = \{-1, +1\}$ in the two-class case. Assume that the space $\mathcal{X} \times \mathcal{Y}$ is endowed with an unknown probability measure P. The goal is then to "learn" a decision function $f : \mathcal{X} \to \mathcal{Y}$ which can predict the label $y \in \mathcal{Y}$ for each point $x \in \mathcal{X}$ "as accurately as possible". Our information consists of a set of n training points $(x_i, y_i)_{i=1,\dots,n}$ which are drawn iid from $\mathcal{X} \times \mathcal{Y}$ according to P. The accuracy of a decision function f is measured with respect to some predefined loss function $\ell(x, y, f(x))$. Popular choices for loss functions in classification are the 0-1-loss $\ell(x, y, f(x)) = \mathbb{1}_{f(x)=y}$, the hinge loss $\ell(x, y, f(x)) = \max\{0, 1 - yf(x)\}$, the squared hinge loss $\ell(x, y, f(x)) = (\max\{0, 1 - yf(x)\})^2$, the least square loss $\ell(x, y, f(x)) = (1 - yf(x))^2$, or the clipped hinge loss $\ell(f(x), y) = 1$ if $yf(x) \leq 0$, $1 - yf(x)$ if $0 \leq yf(x) \leq 1$, and 0 if $yf(x) \geq 1$. For a given loss function, the expected error (also called risk) of a classifier f is defined as

$$\mathcal{R}(f) = \int \ell(x, y, f(x)) dP(x, y).$$

The overall goal of classification is to construct a decision function f which minimizes the risk $\mathcal{R}(f)$. If P was known, it would be clear how the best classifier would look like. It would classify the points according to whether $P(y = 1|x)$ is smaller or larger than $1/2$. This classifier is called the Bayes classifier, and its risk is called the Bayes risk. But in practice we do not know P, and hence we neither can construct the Bayes classifier, nor can we compute the risk $\mathcal{R}(f)$. Instead we often consider the empirical risk

$$\mathcal{R}_{\text{emp}}(f) = \int \ell(x, y, f(x)) dP_n(x, y) = \frac{1}{n} \sum_{i=1}^{n} \ell(x_i, y_i, f(x_i)).$$

For every decision function f, the empirical risk can be computed from the training data. Hence, we can try to find a decision function which has low empirical error, and hope that its true error will also be small. As minimizing the empirical risk often leads to overfitting, it is often better to use a regularization approach. Here

we minimize the regularized risk functional

$$\mathcal{R}_{\text{reg}}(f) = \mathcal{R}_{\text{emp}}(f) + \lambda\Omega(f)$$

where, Ω is a regularization functional (also called regularizer) and λ is some tradeoff constant. Usually the regularizer measures how "well-behaved" a function is, for instance by measuring its variation. By minimizing the regularized risk instead of the empirical risk we artificially shrink the hypothesis class. From a theoretical point of view this has the advantage that classifiers chosen from a small hypothesis class have less tendency to overfit than those chosen from a large function class. This can be seen from generalization bounds proved with methods of statistical learning theory. Those bounds often have the form $\mathcal{R}(f) \leq \mathcal{R}_{\text{emp}}(f) + C(\mathcal{F})$, where $\mathcal{F}$ is the hypothesis class and $C(\mathcal{F})$ a term measuring the "capacity" of this function class. The measures of capacity we will consider later are the Rademacher average R_n and the related maximum discrepancy $\tilde{R}_n$. For an arbitrary class $\mathcal{F}$ of functions, they are defined as

$$R_n(\mathcal{F}) := E\left(\frac{1}{n}\sup_{f\in\mathcal{F}}|\sum_{i=1}^{n}\sigma_i f(X_i)|\right)$$

$$\tilde{R}_n(\mathcal{F}) := E\left(\frac{1}{n}\sup_{f\in\mathcal{F}}|\sum_{i=1}^{n}(f(X_i) - f(X_i'))|\right)$$

where σ_i are iid Rademacher random variables (i.e., $\Pr(\sigma_i = +1) = \Pr(\sigma_i = -1) = 1/2$), X_i and X_i' are iid sample points according to the (unknown) sample distribution, and the expectation is taken with respect to all occurring random variables. It holds that $R_n(\mathcal{F}) \geq \frac{1}{2}\tilde{R}_n(\mathcal{F})$. Sometimes we also consider the conditional Rademacher average $\hat{R}_n$ given by

$$\hat{R}_n(\mathcal{F}) := E\left(\frac{1}{n}\sup_{f\in\mathcal{F}}|\sum_{i=1}^{n}\sigma_i f(X_i)|\,\middle|\, X_1, ..., X_n\right)$$

where the expectation is taken only conditionally on the sample points $X_1, ..., X_n$. Let $\mathcal{F}$ be a class of bounded functions and ℓ the clipped hinge loss function from above. Then the following bound can be proved with techniques using McDiarmid's concentration inequality, symmetrization, and the contraction property of Rademacher averages (cf. Bartlett and Mendelson, 2002, Anthony (2002), and Chapter 3 of Devroye and Lugosi, 2001):

Theorem 1 (Rademacher error bound) *With probability at least $1-\delta$ over the iid drawing of n sample points, every $f \in \mathcal{F}$ satisfies*

$$\mathcal{R}(f) \leq \mathcal{R}_{\text{emp}}(f) + 2R_n(\mathcal{F}) + \sqrt{\frac{8\log(2/\delta)}{n}}.$$

A similar bound can be obtained with the maximum discrepancy (see Bartlett and Mendelson, 2002).

2 Different ways of dealing with dissimilarities for classification

Assume that additionally to the training points, we are given a dissimilarity function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ which measures some kind of distance between the points in $\mathcal{X}$. In the following we want to give a short overview over some approaches how this distance information can be used for classification.

The easiest and most straightforward way for dissimilarity-based classification is the k-nearest neighbor classifier k-NN. It classifies an arbitrary point $x \in \mathcal{X}$ by a majority vote among the k closest training points. This classifier has the advantage that it is easy to implement and easy to understand. Moreover, if one suitably increases k with the sample size n, then it is consistent: if $k \to \infty$ and $k/n \to 0$, then the error of k-NN converges to the Bayes risk. If we simply fix k and let $n \to \infty$, then the error of k-NN is bounded by twice the Bayes risk. On the other hand, k-NN does not work well in high-dimensional spaces or for very noisy training data. For a comprehensive discussion of statistical properties of k-NN we refer to Sections 5 and 11 in Devroye et al. (1996).

Another elegant classifier which works for arbitrary dissimilarity spaces $\mathcal{X}$ is the Linear Programming machine (LP machine) introduced in Graepel et al. (1999b). It constructs a decision function of the form

$$f(x) = \sum_{i=1}^{n} a_i d(x, x_i) + b$$

which minimizes the regularized risk with respect to the regularizer $\Omega(f) = \sum_i |a_i|$. This can be done by solving a linear program. This algorithm is easy to implement, achieves sparse solutions, and works well in practice (Graepel et al., 1999b).

Both k-NN and the LP-machine do not require the data space $\mathcal{X}$ to have any structure, apart from possessing a dissimilarity function. But for many methods it is convenient or even necessary to work in a richer space $\mathcal{X}$. For example, to construct linear classifiers we need to have a vector space, preferably a Hilbert space structure. If $\mathcal{X}$ does not possess the structure we need, we somehow have to impose this structure on $\mathcal{X}$, or have to transfer the problem to another space $\tilde{\mathcal{X}}$ which has the structure we want. During this transformation we have to make sure that we do not loose important information about the original problem. One common way of dealing with this problem is to embed the given space $\mathcal{X}$ into a vector space V such that the original distances are preserved as well as possible. There are many different algorithms which (approximately) solve this problem. They differ in which requirements they have for the dissimilarity function, which information they want to preserve (local or global distances, approximately or exact) and in which space they want to embed. In the following we want to give an overview over some of

those methods.

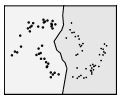
2.1 Globally isometric embeddings into a Hilbert space

The vector space possessing the most "structure" is the Hilbert space. In general it is not possible to embed arbitrary metric spaces *isometrically* into Hilbert spaces. Recall from the introduction that a metric space $(\mathcal{X}, d)$ can be embedded isometrically into a Hilbert space if and only if the metric $-d^2$ is conditionally positive definite, that is $-\sum_{i,j=1}^{l} c_i c_j d^2(x_i, x_j) \geq 0$ for all $l \in \mathbb{N}$, $c = (c_1, ..., c_l) \in \mathbb{R}^l$ with $\sum_i c_i = 0$, and $(x_i)_{i=1,...,l} \in \mathcal{X}$. Below we refer to this condition as Schoenberg condition. A classical method for embedding a finite metric space into an Euclidean space is multidimensional scaling (MDS), cf. Cox and Cox (2001). Given an $n \times n$ distance matrix D, the goal is to find points $p_1, ...p_n \in \mathbb{R}^m$ such that the Euclidean distances between the points coincide with the distances in D, that is $\|p_i - p_j\|_2 = d_{ij}$. This is only possible if the dissimilarity function satisfies the Schoenberg condition. In this case, MDS uses the relation $d_{ij}^2 = \langle p_i - p_j, p_i - p_j \rangle = \langle p_i, p_i \rangle + \langle p_j, p_j \rangle - 2\langle p_i, p_j \rangle$ to derive an inner product matrix K which is consistent with the given distance matrix. The coordinates of the points can then be computed by decomposing $K = XX^t$ and identifying the columns of X with the points p_i. The dimension m of the Euclidean space can be chosen to be the rank of the similarity matrix K.

If the distance function is not Euclidean, then a set of points with $\|p_i - p_j\|_2 = d_{ij}$ does not exist. In this case, the goal is to find a set of points in a Euclidean space such that their distances are approximately the ones given in D. This is usually done by defining a loss function which is then minimized by gradient descent. One problem in this case is that it is not clear how the dimension m should be chosen. The loss usually decreases for increasing dimension as in high-dimensional spaces we have more freedom of choosing points. One heuristic to choose a reasonable dimension is then to choose the smallest dimension such that the loss does not decrease significantly if we further increase the dimension.

2.2 Locally isometric embeddings into a Hilbert space

While in MDS we try to preserve all pairwise distances as well as possible, other approaches suggest only to preserve local distances. Here the intuition is that the data has a manifold structure that should be recovered by the embedding. In this case, the information we want to preserve consists of the intrinsic distances between points along the manifold. In small neighborhoods, those distances are close to the distances measured by d, but for points which are far away from each other this is not true any more. Now the goal is to find an embedding of $\mathcal{X}$ into a low-dimensional space $\mathbb{R}^m$ which preserves the local distances. There are several approaches to this problem. The Isomap algorithm of Tenenbaum et al. (2000) relies on classical MDS,



but replaces the original distance matrix D by a new matrix $\tilde{D}$ which contains the intrinsic geodesic distances between the points. The matrix $\tilde{D}$ is computed as follows. First, we construct the neighborhood graph corresponding to D, consisting of the nodes x_i and connecting each node x_i with its k-nearest neighbors as measured by D (or alternatively, with all points in a ε-neighborhood). Then the length of the shortest path between two nodes x_i and x_j in this graph is defined to be the new distance $\tilde{d}_{ij}$. Finally, classical MDS is used to find a configuration of points in the Euclidean space which approximates the distances in $\tilde{D}$.

A similar goal is pursued with the Locally Linear Embedding algorithm (LLE) of Roweis and Saul (2000). For simplicity let us assume that the training points x_i are points in a high-dimensional space, and we want to map them to points p_i in a low-dimensional space which has the "true" intrinsic dimension of the data. First we look at the points in the high dimensional space. We approximate each point x_i by a linear combination of its k neighbors, that is we want to find weights w_{ij} such that $x_i \approx \sum_{j=i_1,\dots,i_k} w_{ij} x_j$, where the sum goes over the k nearest neighbors of x_i. Secondly we then try to find a set of points p_i in a low dimensional space such that the linear relationships are preserved, that is we minimize the loss $|p_i - \sum_{j=i_1,\dots,i_k} w_{ij} p_j|$, where the weights are fixed to the values determined in the first step. This can be done effectively by solving an eigenvector problem.

A third algorithm falling into the same category is the Laplacian Eigenmap, cf. Belkin and Niyogi (2003a). As in Isomap, we first construct the neighborhood graph corresponding to the given distance matrix. Then we transform the dissimilarities d_{ij} to similarities w_{ij}, compute the graph Laplacian matrix L on the similarity graph and consider the $n \times m$ matrix M containing the first generalized eigenvectors of L. The rows of this matrix are the coordinates of the representation points $x_i \in \mathbb{R}^m$. (For details about graph Laplacians and their eigenvalues we refer to Chapter II). The authors show that this embedding preserves the local distance information in some sense. This embedding is also similar to the one proposed in Ng et al. (2001) for spectral clustering, which we already discussed in Chapter II.

2.3 Isometric embeddings in Banach spaces

If the Schoenberg condition is not satisfied for some space $(\mathcal{X}, d)$, then it cannot be embedded isometrically into a Hilbert space. Instead of relaxing the isometry condition, we can also relax the requirement that the target space should be a Hilbert space. Below we will see that it is always possible to embed metric spaces isometrically into Banach spaces. A Banach space is nearly "as nice" as a Hilbert space, and in this chapter we will study many properties of embeddings in Banach spaces. Some of those embeddings, namely the one into the predual of the space of Lipschitz functions and the Kuratowski embedding into the space of continuous functions, will be discussed in depth in Sections 5 and 9. For embeddings in other

Banach spaces such as L_1-spaces we refer to Watson (1999).

2.4 Isometric embeddings in pseudo-Euclidean spaces

If the distance function d on $\mathcal{X}$ fails to satisfy the triangle inequality, then even embedding $\mathcal{X}$ isometrically into a Banach space is no longer possible, as a norm in a Banach space by definition always satisfies the triangle inequality. In case the distance function is still symmetric and satisfies $d(x,x) = 0$ for all $x \in \mathcal{X}$, it is possible to embed the space $\mathcal{X}$ into a pseudo-Euclidean space such that the dissimilarities are preserved. Pseudo-Euclidean spaces (cf. Goldfarb, 1985) consist of a direct sum of two Hilbert spaces. In both spaces, we first define the norm in the standard way using the scalar product, but then we flip the sign of the norm in one of the two spaces. As a consequence, the norm on the whole space can have attain negative values and does not satisfy the triangle inequality. Embeddings into pseudo-Euclidean spaces have been considered by several authors (Graepel et al., 1999a; Pekalska et al., 2001), but in my opinion they are no good choice. The reason is that the geometry in pseudo-Euclidean spaces does not behave as in standard Euclidean spaces and has very strange properties which are difficult to understand. Hence, just transferring algorithms which were designed for Euclidean spaces to pseudo-Euclidean spaces is not very promising. Instead I believe that in this case, algorithms which directly work on the dissimilarity space should be preferred.

3 Large margin classifiers

Assume the training patterns $(x_i)_{i=1,\dots,n}$ are points in some Hilbert space $\mathcal{H}$ and are separated by a hyperplane

$$h_{(\omega,b)} = \{x \in \mathcal{H};\ \langle x, \omega\rangle + b = 0\}.$$

Then the geometrical margin of the hyperplane $h_{(\omega,b)}$ with respect to the training patterns is defined as

$$\rho(\omega, b) := \min_{i=1,\dots,n} d(x, h_{(\omega,b)}) := \min_{i=1,\dots,n} \min_{h \in h_{(\omega,b)}} \|x_i - h\|$$

and measures the minimal distance of the training points to the separating hyperplane. The large margin principle states that if the training data can be separated by linear hyperplanes, then we should choose the hyperplane that has the largest margin. A classifier which implements the large margin principle is called a large margin classifier. A typical example for large margin classifiers is the support vector machine (cf. Schölkopf and Smola, 2002). If the training points are separable, we choose the hyperplane according to the following optimization problem:

$$\max \rho(\omega, b) \text{ subject to } d(x_i, h_{\omega,b}) \geq \rho,\ y_i(\langle \omega, x_i\rangle + b) \geq 0.$$

The two constraints ensure that all training points lie outside the margin and that they are correctly separated by the hyperplane. The standard formulation of the

SVM exploits the fact that we can express the margin as the inverse of the length of the normal vector ω of the separating hyperplane, that is $\rho = 1/\|\omega\|$. Moreover, we can also merge the two constraints into a single one and get

$$\min \|\omega\|^2 \text{ subject to } y_i(\langle \omega, x_i \rangle + b) \geq 1. \tag{1}$$

This is called the hard margin support vector machine. The term "hard margin" refers to the fact that we try to separate the training points perfectly. In the likely case that the training points cannot be separated by a linear hyperplane because the two classes overlap, we relax the hard margin condition. In this case, we try to find a hyperplane that classifies most training points correctly, has a large margin with respect to most of the training points, but is allowed to ignore that some training points are within the margin or on the wrong side of the hyperplane. Such a classifier will be called a soft margin classifier. The standard formulation in case of the SVM is

$$\min \|\omega\|^2 + C \sum_{i=1}^{n} \xi_i \text{ subject to } y_i(\langle \omega, x_i \rangle + b) \geq 1 - \xi_i,\ \xi_i \geq 0$$

where C is a trade-off parameter, sometimes called the soft margin parameter. The slack variables ξ_i measure how much the training points x_i violate the hard margin principle.

There are several reasons why it is believed that large margin classifiers are good classifiers. Intuitively, the classification of the training patterns is more robust against noise in the input patterns if the separating hyperplane is "far away" from the input patterns. Adding noise to the positions of the patterns then does not change their classification by the hyperplane. Secondly, there are several generalization bounds which bound the true risk of the classifier by the sum of the empirical risk and a capacity term involving the margin of the classifier. The most widely known bound of this type is the radius-margin bound (cf. Section 10.3 of Vapnik, 1998 or Bartlett and Shawe-Taylor, 1999). Here the important quantity in the capacity term is R^2/ρ^2, where R is the radius of the smallest sphere enclosing the training patterns and ρ is the margin of the separating hyperplane. The mathematical reason why those bounds hold true is that selecting only large margin hyperplanes considerably reduces the capacity of the hypothesis class. An overview over many large margin bounds can be found for example in Section 4.3 of Cristianini and Shawe-Taylor (2000). Another intuitive argument in favor of large margin classifiers will be discussed in Chapter IV. There we will see that selecting a large margin classifier satisfies what is known as Ockham's razor: the large margin classifier is in some sense "the simplest" classifier we can choose.

Finally, we can establish a connection between large margin classification and regularization in the case of soft margin SVMs. The idea is to interpret the term $\sum_i \xi_i$ as empirical error $\mathcal{R}_{\text{emp}}$ and the margin $\|\omega\|^2$ as regularizer Ω. This connection

becomes interesting if we apply the "kernel trick" for SVMs. Instead of assuming that our training patterns x_i are elements of a Hilbert space, the space $\mathcal{X}$ is allowed to be an arbitrary set. The the only "structure" we require on this space is that it possesses a similarity function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ which is positive definite. Then we embed the data space $\mathcal{X}$ into a Hilbert space $\mathcal{H}$ by a so called feature mapping $\Phi : \mathcal{X} \to \mathcal{H}$. This embedding can be constructed in such a way that the similarity is preserved: the inner product $\langle \cdot, \cdot \rangle_{\mathcal{H}}$ in $\mathcal{H}$ satisfies $\langle \Phi(x), \Phi(y) \rangle_{\mathcal{H}} = k(x, y)$. With this embedding we implicitly impose a vector space structure on the formerly unstructured data space. Now we have the advantage that we can construct a linear classifier in the Hilbert space $\mathcal{H}$. It is an important aspect of the kernel trick that the solution of the large margin SVM in the feature space has a very special form. Let $h_{(\omega,b)}$ be the large margin hyperplane constructed in the Hilbert space with the training set $(\Phi(x_i), y_i)_{i=1,\dots,n}$. It can be shown that the normal vector ω of the large margin hyperplane has the form $\omega = \sum_i a_i y_i \Phi(x_i)$ for non-negative coefficients a_i. The fact that the solution of the large margin optimization problem can always be expressed as a linear combination of the training points $\Phi(x_i)$ is called the representer theorem. It has the consequence that the decision function given by a hyperplane $h_{(\omega,b)}$ has a special form:

$$\langle \omega, \Phi(x) \rangle_{\mathcal{H}} = \langle \sum_i y_i a_i \Phi(x_i), \Phi(x) \rangle_{\mathcal{H}} = \sum_i y_i a_i \langle \Phi(x_i), \Phi(x) \rangle_{\mathcal{H}} = \sum_i y_i a_i k(x_i, x).$$

The last equality is a consequence of the construction: we chose the Hilbert space $\mathcal{H}$ and the embedding Φ in such a way that the similarities were preserved. Interpreted as function on the input space, the decision function hence is of the form $f(x) = \sum_{i=1}^n y_i a_i k(x, x_i) + b$ where a_i are the coefficients in the linear expansion of ω and b the offset of the hyperplane. The correspondence between hyperplanes and functions of the form $f(x) = \sum_{i=1}^n a_i k(x, x_i)$ also works the other way round. A given function $f(x) = \sum_{i=1}^n y_i a_i k(x, x_i) + b$ with non-negative a_i corresponds to the linear hyperplane $h_{(\sum_i a_i y_i \Phi(x_i), b)}$ with margin

$$\| \sum_i a_i y_i \Phi(x_i) \|_{\mathcal{H}}^2 = \langle \sum_i a_i y_i \Phi(x_i), \sum_i a_i y_i \Phi(x_i) \rangle = \sum_{i,j=1}^n a_i a_j y_i y_j k(x_i, x_j),$$

and the slack variables ξ_i can be recovered by $\xi_i = \max\{0, 1 - y_i f(x_i)\}$.

Now let us come back to the connections between large margin classification and regularization. According to the large margin principle, SVMs construct a linear decision boundary in a Hilbert space $\mathcal{H}$ such that the training points are separated with a large margin and the sum of the margin errors is small. The objective function which is minimized in this process is given by $\|\omega\|_{\mathcal{H}}^2 + C \sum_i \xi_i$. By the representer theorem we know that ω will be a linear combination of training points of the form $\omega = \sum_i a_i y_i \Phi(x_i)$ and the decision function will have the form $f(x) = \sum_i a_i y_i k(x_i, x) + b$. Now we can interpret SVMs from a regularization point of view.

For a given kernel function k we choose the set $\mathcal{F} := \{\sum_{i=1}^{l} c_i k(p_i, \cdot)+b;\ l \in \mathbb{N},\ c_i, b \in \mathbb{R},\ p_i \in \mathcal{X}\}$ as hypothesis space. The empirical error of a function $f \in \mathcal{F}$ will be defined as $\mathcal{R}_{\text{emp}}(f) = \sum_i \xi_i$ with $\xi_i = \max\{0, 1 - y_i f(x_i)\}$, and as regularizer we choose $\Omega(f) = \| \sum_i c_i \Phi(x_i)\|_{\mathcal{H}}^2 = \sum_{ij} c_i c_j k(x_i, x_j)$. Then we construct our decision function as minimizer of the regularized risk functional $\mathcal{R}_{\text{reg}}(f) = \mathcal{R}_{\text{emp}}(f) + \lambda\Omega(f)$. By the representer theorem we know that the minimizer of the regularized risk functional can always be chosen such that the points p_i are elements of the training set, that is the solution will have the form $f(x) = \sum_i c_i k(x_i, x)+b$. With $a_i := c_i/y_i$, we can recover the SVM solution $f(x) = \sum_i a_i y_i k(x_i, x) + b$.

The advantage of the regularization formulation is that it is more intuitive than the Hilbert space formulation. Moreover, we can now try to find out whether the regularizer Ω has a geometrically plausible interpretation on the function space $\mathcal{F}$. Indeed, for many kernel functions k this is the case. In can be seen (cf. Smola et al., 1998) that for many kernel functions k, the regularizer Ω on $\mathcal{F}$ measures some kind of smoothness of the function f. This gives us one more explanation why large margin classifiers work well, at least in the context of SVMs: choosing functions with a large margin in $\mathcal{H}$ corresponds to choosing very smooth functions in $\mathcal{F}$, which is a very sensible thing to do in many applications.

4 Large margin classification on metric spaces

In the last section we explained the concept of large margin classifiers in Hilbert spaces. The decision function was a linear hyperplane, and the margin the distance of the closest training point to the hyperplane. Now we want to generalize the concept of large margin classification to more general spaces than Hilbert spaces. In the last section we have already seen how this can be done for an input space $\mathcal{X}$ which possesses a positive definite similarity function k. We first embed $\mathcal{X}$ into a Hilbert space $\mathcal{H}$ such that the given similarity k on $\mathcal{X}$ is preserved, and then construct a large margin classifier in $\mathcal{H}$. Now we want to consider the situation where $\mathcal{X}$ is a metric space. Instead of a positive definite similarity function we are given a metric d on $\mathcal{X}$. According to the principle described above, the easiest way to construct a large margin classifier would now be to embed the metric space $\mathcal{X}$ into a Hilbert space such that the distances are preserved, and then construct a large margin classifier in this Hilbert space. But here we come across a problem which we already mentioned above. In general, it is not possible to embed metric spaces isometrically into a Hilbert space. Now we have two options. Either we relax the condition that the embedding should be an isometry. Instead, we could use one of the methods described in Section 2 to embed $\mathcal{X}$ approximately isometrically into some Hilbert space, or even only preserving local distances. The other option is to relax the condition that we want to embed the metric space into a Hilbert space. A space which is only a bit weaker than a Hilbert space is a Banach space. It is also a complete vector space, but only possesses a norm instead of a scalar product. So we still have linear structure to define hyperplanes, and we still have a norm to

measure the margin in this space.

This second option is the one we will follow now, and as in the support vector machine case our construction will have close connections to regularization.

From the regularization point of view, a convenient set of functions on a metric space is the set of Lipschitz functions, as functions with a small Lipschitz constant have low variation. Thus it seems desirable to separate the different classes by a decision function which has a small Lipschitz constant. We will construct the large margin classifier which corresponds to this regularization approach. To this end, we embed the metric space $(\mathcal{X}, d)$ in a Banach space $\mathcal{B}$ and the space of Lipschitz functions into its dual space $\mathcal{B}'$. Remarkably, both embeddings can be realized as isometries simultaneously. By this construction, each $x \in \mathcal{X}$ will correspond to some $m_x \in \mathcal{B}$ and each Lipschitz function f on $\mathcal{X}$ to some functional $T_f \in \mathcal{B}'$ such that $f(x) = T_f m_x$ and the Lipschitz constant $L(f)$ is equal to the operator norm $\|T_f\|$. In the Banach space $\mathcal{B}$ we can then construct a large margin classifier such that the size of the margin will be given by the inverse of the operator norm of the decision functional. The basic algorithm implementing this approach is

$$\text{minimize } \mathcal{R}_{\text{emp}}(f) + \lambda L(f)$$

in regularization language and

$$\text{minimize } L(f) + C \sum\nolimits_i \xi_i \text{ subject to } y_i f(x_i) \geq 1 - \xi_i,\ \xi_i \geq 0$$

in large margin language. In both cases, $L(f)$ denotes the Lipschitz constant of the function f, and the minimum is taken over a subset of Lipschitz functions on $\mathcal{X}$. To apply this algorithm in practice, the choice of this subset will be important. We will see that by choosing different subsets we can recover the SVM, the linear programming machine, and even the 1-nearest neighbor classifier. In particular this shows that all these algorithms are large margin algorithms. So the Lipschitz framework can help to analyze a wide range of algorithms which do not seem to be connected at the first glance.

5 Lipschitz function spaces

In this section we introduce several Lipschitz function spaces and their properties. For a comprehensive overview we refer to Weaver (1999).
A function $f : \mathcal{X} \to \mathbb{R}$ on a metric space $(\mathcal{X}, d)$ is called a Lipschitz function if there exists a constant L such that $|f(x) - f(y)| \leq L d(x, y)$ for all $x, y \in \mathcal{X}$. The smallest constant L such that this inequality holds is called the Lipschitz constant of f, denoted by $L(f)$. For convenience, we recall some standard facts about Lipschitz functions:

Lemma 2 (Lipschitz functions) *Let $(\mathcal{X}, d)$ be a metric space, $f, g : \mathcal{X} \to \mathbb{R}$ Lipschitz functions and $a \in \mathbb{R}$. Then $L(f+g) \leq L(f) + L(g)$, $L(af) \leq |a|L(f)$ and $L(\min(f,g)) \leq \max\{L(f), L(g)\}$, where $\min(f,g)$ denotes the pointwise minimum of the functions f and g. Moreover, let $f := \lim_{n\to\infty} f_n$ the pointwise limit of Lipschitz functions f_n with $L(f_n) \leq c$ for all $n \in \mathbb{N}$. Then f is a Lipschitz function with $L(f) \leq c$.*

For a metric space $(\mathcal{X}, d)$ consider the set

$$\mathrm{Lip}(\mathcal{X}) := \{f : \mathcal{X} \to \mathbb{R};\ f \text{ is a bounded Lipschitz function}\}.$$

It forms a vector space, and the Lipschitz constant $L(f)$ is a seminorm on this space. To define a convenient norm on this space we restrict ourselves to *bounded* metric spaces. These are spaces which have a finite diameter $\mathrm{diam}(\mathcal{X}) := \sup_{x,y\in\mathcal{X}} d(x,y)$. For the learning framework this is not a big drawback as the training and test data can always be assumed to come from a bounded region of the underlying space. For a bounded metric space $\mathcal{X}$ we choose the norm

$$\|f\|_L := \max\left\{L(f), \frac{\|f\|_\infty}{\mathrm{diam}(\mathcal{X})}\right\}$$

as our default norm on the space $\mathrm{Lip}(\mathcal{X})$. It is easy to see that this indeed is a norm. Note that in the mathematical literature, $\mathrm{Lip}(\mathcal{X})$ is usually endowed with the slightly different norm $\|f\| := \max\{L(f), \|f\|_\infty\}$. But we will see that the norm $\|\cdot\|_L$ fits very naturally in our classification setting, as already can be seen by the following intuitive argument. Functions that are used as classifiers are supposed to take positive and negative values on the respective classes and satisfy

$$\|f\|_\infty = \sup_x |f(x)| \leq \sup_{x,y} |f(x) - f(y)| \leq \mathrm{diam}(\mathcal{X})L(f), \tag{2}$$

that is $\|f\|_L = L(f)$. Hence, the L-norm of a classification decision function is determined by the quantity $L(f)$ we use as regularizer later on. Some more technical reasons for the choice of $\|\cdot\|_L$ will become clear later.

Another important space of Lipschitz functions is constructed as follows. Let $(\mathcal{X}_0, d)$ be a metric space with a distinguished "base point" e which is fixed in advance. $(\mathcal{X}_0, d, e)$ is called a *pointed metric space*. We define

$$\mathrm{Lip}_0(\mathcal{X}_0) := \{f \in \mathrm{Lip}(\mathcal{X}_0);\ f(e) = 0\}.$$

On this space, the Lipschitz constant $L(\cdot)$ is a norm. However, its disadvantage in the learning framework is the condition $f(e) = 0$, which is an inconvenient a priori restriction on our classifier as e has to be chosen in advance. To overcome this restriction, for a given bounded metric space $(\mathcal{X}, d)$ we define a corresponding

extended pointed metric space $\mathcal{X}_0 := \mathcal{X} \cup \{e\}$ for a new base element e with the metric

$$d_{\mathcal{X}_0}(x,y) = \begin{cases} d(x,y) & \text{for } x,y \in \mathcal{X} \\ \operatorname{diam}(\mathcal{X}) & \text{for } x \in \mathcal{X}, y = e. \end{cases} \tag{3}$$

Note that $\operatorname{diam}(\mathcal{X}_0) = \operatorname{diam}(\mathcal{X})$. Then we define the map

$$\psi : \operatorname{Lip}(\mathcal{X}) \to \operatorname{Lip}_0(\mathcal{X}_0), \quad \psi(f)(x) = \begin{cases} f(x) & \text{if } x \in \mathcal{X} \\ 0 & \text{if } x = e. \end{cases} \tag{4}$$

Lemma 3 (Isometry between Lipschitz function spaces) *ψ is an isometric isomorphism between* $\operatorname{Lip}(\mathcal{X})$ *and* $\operatorname{Lip}_0(\mathcal{X}_0)$.

Proof. Obviously, ψ is bijective and linear. Moreover, for $f_0 := \psi(f)$ we have

$$\begin{aligned} L(f_0) &= \sup_{x,y \in \mathcal{X}_0} \frac{|f_0(x) - f_0(y)|}{d_{\mathcal{X}_0}(x,y)} = \max\{\sup_{x,y \in \mathcal{X}} \frac{|f(x) - f(y)|}{d(x,y)}, \sup_{x \in \mathcal{X}} \frac{|f(x) - f(e)|}{d_{\mathcal{X}_0}(x,e)}\} \\ &= \max\{L(f), \frac{\|f\|_\infty}{\operatorname{diam}(\mathcal{X})}\} = \|f\|_L \end{aligned}$$

Hence, ψ is an isometry. ☺

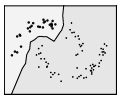
In some respects, the space $(\operatorname{Lip}_0(\mathcal{X}_0), L(\cdot))$ is more convenient to work with than $(\operatorname{Lip}(\mathcal{X}), \|\cdot\|_L)$. In particular it has some very useful duality properties. Let $(\mathcal{X}_0, d, e)$ be a pointed metric space with some distinguished base element e. A *molecule* of $\mathcal{X}_0$ is a function $m : \mathcal{X}_0 \to \mathbb{R}$ such that its support (i.e., the set where m has non-zero values) is a finite set and $\sum_{x \in \mathcal{X}_0} m(x) = 0$. For $x, y \in \mathcal{X}_0$ we define the *basic molecules* $m_{xy} := \mathbb{1}_x - \mathbb{1}_y$. It is easy to see that every molecule m can be written as a (non unique) finite linear combination of basic molecules. Thus we can define

$$\|m\|_{AE} := \inf \left\{ \sum_i |a_i| d(x_i, y_i);\ m = \sum_i a_i m_{x_i y_i} \right\}$$

which is a norm on the space of molecules. The completion of the space of molecules with respect to $\|\cdot\|_{AE}$ is called the Arens-Eells space $AE(\mathcal{X}_0)$. Denoting its dual space (i.e., the space of all continuous linear forms on $AE(\mathcal{X}_0)$) by $AE(\mathcal{X}_0)'$ the following theorem holds true (cf. Arens and Eells, 1956; Weaver, 1999).

Theorem 4 (Isometry between $AE(\mathcal{X}_0)'$ and $\operatorname{Lip}_0(\mathcal{X}_0)$) *$AE(\mathcal{X}_0)'$ is isometrically isomorphic to* $\operatorname{Lip}_0(\mathcal{X}_0)$.

This means that we can regard a Lipschitz function f on $\mathcal{X}_0$ as a linear functional T_f on the space of molecules, and the Lipschitz constant $L(f)$ coincides with the



operator norm of the corresponding functional T_f. For a molecule m and a Lipschitz function f this duality can be expressed as

$$\langle f, m\rangle = \sum_{x\in\mathcal{X}_0} m(x)f(x). \tag{5}$$

It can be proved that $\|m_{xy}\|_{AE} = d(x,y)$ holds for all basic molecules m_{xy}. Hence, it is possible to embed $\mathcal{X}_0$ isometrically in $AE(\mathcal{X}_0)$ via

$$\Gamma : \mathcal{X}_0 \to AE(\mathcal{X}_0),\ x \mapsto m_{xe}. \tag{6}$$

The norm $\|\cdot\|_{AE}$ has a nice geometrical interpretation in terms of the *mass transportation problem* (cf. Weaver, 1999): some product is manufactured in varying amounts at several factories and has to be distributed to several shops. The (discrete) transportation problem is to find an optimal way to transport the product from the factories to the shops. The costs of such a transport are defined as $\sum_{ij} a_{ij}d_{ij}$ where a_{ij} denotes the amount of the product transported from factory i to shop j and d_{ij} the distance between them. If f_i denotes the amount produced in factory i and s_i denotes the amount needed in shop i, the formal definition of the transportation problem is

$$\min_{i,j=1,\dots,n} \sum a_{ij}d_{ij} \quad \text{subject to} \quad a_{ij} \geq 0,\ \sum_j a_{ij} = s_j,\ \sum_i a_{ij} = f_i. \tag{7}$$

To connect the Arens-Eells space to this problem we identify the locations of the factories and shops with a molecule m. The points x with $m(x) > 0$ represent the factories, the ones with $m(x) < 0$ the shops. It can be proved that $\|m\|_{AE}$ equals the minimal transportation costs for molecule m. A special case is when the given molecule has the form $m_0 = \sum m_{x_iy_j}$. In this case, the transportation problem reduces to the *bipartite minimal matching problem*: given $2m$ points $(x_1,\dots,x_n,y_1,\dots,y_n)$ in a metric space, we want to match each of the x-points to one of the y-points such that the sum of the distances between the matched pairs is minimal. The formal statement of this problem is

$$\min_{\pi} \sum_{i,j} d(x_i, y_{\pi(i)}) \tag{8}$$

where the minimum is taken over all permutations π of the set $\{1,...,n\}$ (cf. Steele, 1997).

In Section 7 we will also need the notion of a vector lattice. A vector lattice is a vector space V with an ordering $\preceq$ which respects the vector space structure (i.e., for $x,y,z \in V, a > 0$: $x \preceq y \implies x+z \preceq y+z$ and $ax \preceq ay$) and such that for any two elements $f,g \in V$ there exists a greatest lower bound $\inf(f,g)$. In particular, the space of Lipschitz functions with the ordering $f \preceq g \iff \forall x\ f(x) \leq g(x)$ forms a vector lattice.

6 The Lipschitz classifier

Let $(\mathcal{X}, d)$ be a metric space and $(x_i, y_i)_{i=1,\dots,n} \subset \mathcal{X} \times \{\pm 1\}$ some training data. In order to be able to define hyperplanes, we want to embed $(\mathcal{X}, d)$ into a vector space, but without loosing or changing the underlying metric structure.

6.1 Embedding and large margin in Banach spaces

Our first step is to embed $\mathcal{X}$ by the identity mapping into the extended space $\mathcal{X}_0$ as described in (3), which in turn is embedded into $AE(\mathcal{X}_0)$ via (6). We denote the resulting composite embedding by

$$\Phi : \mathcal{X} \to AE(\mathcal{X}_0),\ x \mapsto m_x := m_{xe}.$$

Secondly, we identify $\mathrm{Lip}(\mathcal{X})$ with $\mathrm{Lip}_0(\mathcal{X}_0)$ according to (4) and then $\mathrm{Lip}_0(\mathcal{X}_0)$ with $AE(\mathcal{X}_0)'$ according to Theorem 4. Together this defines the map

$$\Psi : \mathrm{Lip}(\mathcal{X}) \to AE(\mathcal{X}_0)',\ f \mapsto T_f.$$

Lemma 5 (Properties of the embeddings) *The mappings Φ and Ψ have the following properties:*

1. *Φ is an isometric embedding of $\mathcal{X}$ into $AE(\mathcal{X}_0)$: to every point $x \in \mathcal{X}$ corresponds a molecule $m_x \in AE(\mathcal{X}_0)$ such that $d(x, y) = \|m_x - m_y\|_{AE}$ for all $x, y \in \mathcal{X}$.*
2. *$\mathrm{Lip}(\mathcal{X})$ is isometrically isomorphic to $AE(\mathcal{X}_0)'$: to every Lipschitz function f on $\mathcal{X}$ corresponds an operator T_f on $AE(\mathcal{X}_0)$ such that $\|f\|_L = \|T_f\|$ and vice versa.*
3. *It makes no difference whether we evaluate operators on the image of $\mathcal{X}$ in $AE(\mathcal{X}_0)$ or apply Lipschitz functions on $\mathcal{X}$ directly: $T_f m_x = f(x)$.*
4. *Scaling a linear operator is the same as scaling the corresponding Lipschitz function: for $a \in \mathbb{R}$ we have $aT_f = T_{af}$.*

Proof. All these properties are direct consequences of the construction and Equation (5).

☺

The message of this lemma is that it makes no difference whether we classify our training data on the space $\mathcal{X}$ with the decision function $\operatorname{sgn} f(x)$ or on $AE(\mathcal{X}_0)$ with the hyperplane $\operatorname{sgn}(T_f m_x)$. The advantage of the latter is that constructing a large margin classifier in a Banach space is a well studied problem. In Bennett and Bredensteiner (2000) and Zhou et al. (2002) it has been established that constructing a maximal margin hyperplane between the set X^+ of positive and X^- of negative training points in a Banach space V is equivalent to finding the distance between the convex hulls of X^+ and X^-. More precisely, let C^+ and C^- the convex hulls of

the sets X^+ and X^-. In the separable case, we define the margin of a separating hyperplane H between C^+ and C^- as the minimal distance between the training points and the hyperplane:

$$\rho(H) := \inf_{i=1,...,n} d(x_i, H).$$

The margin of the maximal margin hyperplane coincides with half the distance

$$d(C^+, C^-) = \inf_{p^+ \in C^+, p^- \in C^-} \|p^+ - p^-\|$$

between the convex hulls of the positive and negative training points. Hence, determining the maximum margin hyperplane can be understood as solving the optimization problem $\inf_{p^+ \in C^+, p^- \in C^-} \|p^+ - p^-\|$. By duality arguments (cf. Bennett and Bredensteiner, 2000) it can be seen that its solution coincides with the solution of

$$\sup_{T \in V'} \inf_{p^+ \in C^+, p^- \in C^-} \langle T, p^+ - p^- \rangle / \|T\|.$$

This can be equivalently rewritten as the optimization problem

$$\inf_{T \in V', b \in \mathbb{R}} \|T\| \text{ subject to } y_i(\langle T, x_i \rangle + b) \geq 1 \ \ \forall i = 1, ..., n. \tag{9}$$

A solution of this problem is called a large margin classifier. The decision function has the form $f(x) = \langle T, x \rangle + b$, and its margin is given by $1/\|T\|$. For details we refer to Bennett and Bredensteiner (2000) and Zhou et al. (2002).

6.2 Derivation of the algorithm

Now we can apply this construction to our situation. We embed $\mathcal{X}$ isometrically into the Banach space $AE(\mathcal{X}_0)$ and use the above reasoning to construct a large margin classifier. As the dual space of $AE(\mathcal{X}_0)$ is $\mathrm{Lip}_0(\mathcal{X}_0)$ and $\langle f, m_x \rangle = f(x)$, the optimization problem (9) in our case is

$$\inf_{f_0 \in \mathrm{Lip}_0(\mathcal{X}_0), b \in \mathbb{R}} L(f_0) \text{ subject to } y_i(f_0(x_i) + b) \geq 1 \ \ \forall i = 1, ..., n.$$

By the isometry stated in Theorem 4, this is equivalent to the problem

$$\inf_{f \in \mathrm{Lip}(\mathcal{X}), b \in \mathbb{R}} \|f\|_L \text{ subject to } y_i(f(x_i) + b) \geq 1 \ \ \forall i = 1, ..., n.$$

Next we want to show that the solution of this optimization problem does not depend on the variable b. To this end, we first set $g := f + b \in \mathrm{Lip}(\mathcal{X})$ to obtain

$$\inf_{g \in \mathrm{Lip}(\mathcal{X}), b \in \mathbb{R}} \|g - b\|_L \text{ subject to } y_i g(x_i) \geq 1 \ \ \forall i = 1, ..., n.$$

Then we observe that if the training data contains points from both classes, then

$$\|g-b\|_L = \max\{L(g-b), \frac{\|g-b\|_\infty}{\operatorname{diam}(\mathcal{X})}\} = \max\{L(g), \frac{\|g-b\|_\infty}{\operatorname{diam}(\mathcal{X})}\}$$
$$\geq L(g) = \max\{L(g), \frac{\|g\|_\infty}{\operatorname{diam}(\mathcal{X})}\} = \|g\|_L.$$

Here the last step is true because of the fact that g takes positive and negative values and thus $\|g\|_\infty/\operatorname{diam}(\mathcal{X}) \leq L(g)$ as we explained in Equation (2) of Section 5. Hence, under the constraints $y_i g(x_i) \geq 1$ we have $\inf_b \|g-b\|_L = L(g)$, and we can rewrite our optimization problem in the final form

$$\inf_{f\in \operatorname{Lip}(\mathcal{X})} L(f) \text{ subject to } y_i f(x_i) \geq 1,\ i=1,\dots,n. \qquad (*)$$

We call a solution of this problem a (hard margin) *Lipschitz classifier*. So we have proved:

Theorem 6 (Lipschitz classifier) *Let $(\mathcal{X}, d)$ be a bounded metric space, $(x_i, y_i)_{i=1,\dots,n} \subset \mathcal{X} \times \{\pm 1\}$ some training data containing points of both classes. Then a solution f of $(*)$ is a large margin classifier, and its margin is given by $1/L(f)$.*

One nice aspect about the above construction is that the margin constructed in the space $AE(\mathcal{X}_0)$ also has a geometrical meaning in the original input space $\mathcal{X}$ itself: it is a lower bound on the minimal distance between the "separation surface" $S := \{s \in \mathcal{X};\ f(s) = 0\}$ and the training points. To see this, normalize the function f such that $\min_{i=1,\dots,n} |f(x_i)| = 1$. This does not change the set S. Because of

$$1 \leq |f(x_i)| = |f(x_i) - f(s)| \leq L(f) d(x_i, s)$$

we thus get $d(x_i, s) \geq 1/L(f)$.
Analogously to SVMs we also define the soft margin version of the Lipschitz classifier by introducing slack variables ξ_i to allow some training points to lie inside the margin or even be misclassified:

$$\inf_{f\in \operatorname{Lip}(\mathcal{X})} L(f) + C\sum_{i=1}^{n} \xi_i \text{ subject to } y_i f(x_i) \geq 1 - \xi_i,\ \xi_i \geq 0. \qquad (**)$$

In regularization language, the soft margin Lipschitz classifier can be stated as

$$\inf_{f\in \operatorname{Lip}(\mathcal{X})} \ell(y_i f(x_i)) + \lambda L(f)$$

where the loss function ℓ is given by $\ell(y_i f(x_i)) = \max\{0, 1 - y_i f(x_i)\}$.

In Section 7, we will give an analytic expression for a solution of $(*)$ and show how $(**)$ can be written as a linear programming problem. However, it may be sensible to restrict the set over which the infimum is taken in order to avoid overfitting.

We thus suggest to consider the above optimization problems over subspaces of $\mathrm{Lip}(\mathcal{X})$ rather than the whole space $\mathrm{Lip}(\mathcal{X})$. In Section 9 we derive a geometrical interpretation of the choice of different subspaces. Now we want to point out some special cases.
Assume that we are given training points in some reproducing kernel Hilbert space H. As it is always the case for linear functions, the Lipschitz constant of a linear function in H' coincides with its Hilbert space norm. This means that the support vector machine in H chooses the same linear function as the Lipschitz algorithm, if the latter takes the subspace of linear functions as hypothesis space.
In the case where we optimize over the subset of all linear combinations of distance functions of the form $f(x) = \sum_{i=1}^{n} a_i d(x_i, x) + b$, the Lipschitz algorithm can be approximated by the linear programming machine (cf. Graepel et al., 1999b):

$$\inf_{a,b} \sum_{i=1}^{n} |a_i| \text{ subject to } y_i(\sum_{i=1}^{n} a_i d(x_i, x) + b) \geq 1.$$

The reason for this is that the Lipschitz constant of a function

$$f(x) = \sum_{i=1}^{n} a_i d(x_i, x) + b$$

is upper bounded by $\sum_i |a_i|$.
Furthermore, if we do not restrict the function space at all, then we will see in the next section that the 1-nearest neighbor classifier is a solution of the Lipschitz algorithm.

These examples show that the Lipschitz algorithm is a very general approach. By choosing different subsets of Lipschitz functions we recover several well known algorithms. As the Lipschitz algorithm is a large margin algorithm according to Theorem 6, the same holds for the recovered algorithms. For instance the linear programming machine, originally designed with little theoretical justification, can now be understood as a large margin algorithm.

7 Representer theorems

A crucial theorem in the context of SVMs and other kernel algorithms is the representer theorem (cf. Schölkopf and Smola, 2002). It states that even though the space of possible solutions of these algorithms forms an infinite dimensional space, there always exists a solution in the finite dimensional subspace spanned by the training points. It is because of this theorem that SVMs overcome the curse of dimensionality and yield computationally tractable solutions. In this section we prove a similar theorem for the Lipschitz classifiers $(*)$ and $(**)$. To simplify the discussion, we denote $\mathcal{D} := \{d(x, \cdot);\ x \in \mathcal{X}\} \cup \{\mathbb{1}\}$ and $\mathcal{D}_{\text{train}} := \{d(x_i, \cdot);\ x_i \text{ training point }\} \cup \{\mathbb{1}\}$, where $\mathbb{1}$ is the constant-1 function.

7.1 Soft margin case

We first start by recalling a general result which implies the classical representer theorem in the case of SVMs.

Lemma 7 (Minimum norm interpolation) *Let V be a function of $n+1$ variables which is non-decreasing in its $n+1$-st argument. Given n points $x_1, \ldots, x_n$ and a functional Ω, any function which is a solution of the problem*

$$\inf_f V(f(x_1), \ldots, f(x_n), \Omega(f)) \tag{10}$$

is a solution of the minimum norm interpolation problem

$$\inf_{f : \forall i,\, f(x_i) = a_i} \Omega(f) \tag{11}$$

for some $a_1, \ldots, a_n \in \mathbb{R}$.

Here, f being a solution of a problem of the form $\inf W(f)$ means $f = \operatorname{argmin} W(f)$. We learned this theorem from M. Pontil, but it seems to be due to C. Micchelli.

Proof. Let f_0 be a solution of the first problem. Take $a_i = f_0(x_i)$. Then for any function f such that $f(x_i) = a_i$ for all i, we have

$$\begin{aligned} &V(f(x_1), \ldots, f(x_n), \Omega(f)) \geq V(f_0(x_1), \ldots, f_0(x_n), \Omega(f_0)) \\ &= V(f(x_1), \ldots, f(x_n), \Omega(f_0)). \end{aligned}$$

Hence, by monotonicity of V we get $\Omega(f) \geq \Omega(f_0)$, which concludes the proof. ☺

The meaning of the above result is that if the solutions of problem (11) have specific properties, then the solutions of problem (10) will also have these properties. So instead of studying the properties of solutions of $(**)$ directly, we will investigate the properties of (11) when the functional Ω is the Lipschitz norm. We first need to introduce the concept of Lipschitz extensions.

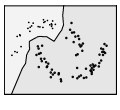
Lemma 8 (Lipschitz extension) *Given a function f defined on a finite subset $x_1, \ldots, x_n$ of $\mathcal{X}$, there exists a function f' which coincides with f on $x_1, \ldots, x_n$, is defined on the whole space $\mathcal{X}$, and has the same Lipschitz constant as f. Additionally, it is possible to explicitly construct f' in the form*

$$f'(x) = \alpha \min_{i=1,\ldots,n} (f(x_i) + L(f)d(x, x_i)) + (1-\alpha) \max_{i=1,\ldots,n} (f(x_i) - L(f)d(x, x_i)),$$

for any $\alpha \in [0,1]$, with $L(f) = \max_{i,j=1,\ldots,n}(f(x_i) - f(x_j))/d(x_i, x_j)$.

Proof. Consider the function $g(x) = \min_{i=1,\ldots,n}(f(x_i) + L(f)d(x, x_i))$. We have

$$|g(x) - g(y)| \leq \max_{i=1,\ldots,n} |f(x_i) + L(f)d(x, x_i) - f(x_i) - L(f)d(y, x_i)| \leq L(f)d(x, y),$$



so that $L(g) \leq L(f)$. Also, by definition $g(x_i) \leq f(x_i) + L(f)d(x_i, x_i) = f(x_i)$. Moreover, if i_0 denotes the index where the minimum is achieved in the definition of $g(x_i)$, i.e. $g(x_i) = f(x_{i_0}) + L(f)d(x_i, x_{i_0})$, then by definition of $L(f)$ we have $g(x_i) \geq f(x_{i_0}) + (f(x_i) - f(x_{i_0})) = f(x_i)$. As a result, for all $i = 1, \ldots, n$ we have $g(x_i) = f(x_i)$, which also implies that $L(g) = L(f)$.
Now the same reasoning can be applied to $h(x) = \max_{i=1,\ldots,n}(f(x_i) - L(f)d(x, x_i))$. Since $\alpha \in [0, 1]$ we have $f'(x_i) = f(x_i)$ for all i. Moreover, $L(\alpha g + (1-\alpha)h) \leq \alpha L(g) + (1-\alpha)L(h) = L(f)$ and thus $L(f') = L(f)$, which concludes the proof. ☺

From the above lemma, we obtain an easy way to construct solutions of minimum norm interpolation problems like (11) with Lipschitz norms, as is expressed in the next lemma.

Lemma 9 (Solution of the Lipschitz minimal norm interpolation problem) *Let $a_1, \ldots, a_n \in \mathbb{R}^n$, $\alpha \in [0, 1]$, $L_0 = \max_{i,j=1,\ldots,n}(a_i - a_j)/d(x_i, x_j)$, and*

$$f_\alpha(x) := \alpha \min_{i=1,\ldots,n} (a_i + L_0 d(x, x_i)) + (1-\alpha) \max_{i=1,\ldots,n} (a_i - L_0 d(x, x_i)) .$$

Then f_α is a solution of the minimal norm interpolation problem (11) *with $\Omega(f) = L(f)$. Moreover, when $\alpha = 1/2$ then f_α is a solution of the minimal norm interpolation problem* (11) *with $\Omega(f) = \|f\|_L$.*

Proof. Given that a solution f of (11) has to satisfy $f(x_i) = a_i$, it cannot have $L(f) < L_0$. Moreover, by Lemma 8 f_α satisfies the constraints and has $L(f) = L_0$, hence it is a solution of (11) with $\Omega(f) = L(f)$.
When one takes $\Omega(f) = \|f\|_L$, any solution f of (11) has to have $L(f) \geq L_0$ and $\|f\|_\infty \geq \max_i |a_i|$. The proposed solution f_α with $\alpha = 1/2$ not only satisfies the constraints $f_\alpha(x_i) = a_i$ but also has $L(f) = L_0$ and $\|f\|_\infty = \max_i |a_i|$, which shows that it is a solution of the considered problem.
To prove that $\|f\|_\infty = \max_i |a_i|$, consider $x \in \mathcal{X}$ and denote by i_1 and i_2 the indices where the minimum and the maximum, respectively, are achieved in the definition of $f_\alpha(x)$. Then one has

$$f_{1/2}(x) \leq \frac{1}{2}\left(a_{i_2} + L_0 d(x, x_{i_2})\right) + \frac{1}{2}\left(a_{i_2} - L_0 d(x, x_{i_2})\right) = a_{i_2} ,$$

and similarly $f_{1/2}(x) \geq a_{i_1}$. ☺

Now we can formulate a general representer theorem for the soft margin Lipschitz classifier.

Theorem 10 (Soft margin representer theorem) *There exists a solution of the soft margin Lipschitz classifier* (∗∗) *in the vector lattice spanned by $\mathcal{D}_{\text{train}}$ which is of the form*

$$f(x) = \frac{1}{2}\min(a_i + L_0 d(x, x_i)) + \frac{1}{2}\max(a_i - L_0 d(x, x_i))$$

for some real numbers $a_1, \ldots, a_n$ with $L_0 := \max_{i,j}(a_i - a_j)/d(x_i, x_j)$. Moreover one has $\|f\|_L = L(f) = L_0$.

Proof. The first claim follows from Lemmas 7 and 9. The second claim follows from the fact that a solution of $(**)$ satisfies $\|f\|_L = L(f)$. ☺

Theorem 10 is remarkable as the space $\mathrm{Lip}(\mathcal{X})$ of possible solutions of $(**)$ contains the whole vector lattice spanned by $\mathcal{D}$. The theorem thus states that even though the Lipschitz algorithm searches for solutions in the whole lattice spanned by $\mathcal{D}$ it always manages to come up with a solution in the sublattice spanned by $\mathcal{D}_{\text{train}}$.

7.2 Algorithmic consequences

Theorem 10 states that there always exists a solution f of $(**)$ in a certain parametric form with n parameters $a_1, ..., a_n$. As a consequence, we can rewrite the original soft margin optimization problem

$$\inf_{f \in \mathrm{Lip}(\mathcal{X})} L(f) + C \sum_{i=1}^{n} \xi_i \quad \text{subject to} \quad y_i f(x_i) \geq 1 - \xi_i,\ \xi_i \geq 0$$

in the form

$$\min_{a_1,\ldots,a_n \in \mathbb{R}} \max_{\substack{i,j=1,\ldots,n \\ i \neq j}} \frac{a_i - a_j}{d(x_i, x_j)} + C \sum_{i=1}^{n} \xi_i \quad \text{subject to} \quad y_i a_i \geq 1 - \xi_i,\ \xi_i \geq 0$$

which is equivalent to the linear program

$$\min_{\substack{a_1,\ldots,a_n \in \mathbb{R} \\ \rho \in \mathbb{R}}} \rho + C \sum_{i=1}^{n} \xi_i \quad \text{subject to} \quad y_i a_i \geq 1 - \xi_i,\ \xi_i \geq 0,\ \frac{a_i - a_j}{d(x_i, x_j)} \geq \rho$$

To construct a soft margin Lipschitz classifier, we thus have to find the parameters $a_1, ..., a_n$ which solve this linear program and use the function f defined in Theorem 10 as classifier. Note, however, that in practical applications, the solution found by this procedure will usually overfit as it optimizes $(**)$ over the whole class $\mathrm{Lip}(\mathcal{X})$. Later we will see how this can be avoided by considering subclasses of Lipschitz functions.

7.3 Hard margin case

The representer theorem for the soft margin case clearly also holds in the hard margin case, so that there will always be a solution of $(*)$ in the vector lattice spanned by $\mathcal{D}_{\text{train}}$. But in the hard margin case, also a different representer theorem is valid. We denote the set of all training points with positive label by X^+, the set of the training points with negative label by X^-, and for two subsets $A, B \subset \mathcal{X}$ we define $d(A, B) := \inf_{a \in A, b \in B} d(a, b)$.

Theorem 11 (Hard margin representer theorem) *Problem* $(*)$ *always has a solution which is a linear combination of distances to* sets *of training points.*

To prove this theorem we first need a simple lemma.

Lemma 12 (Optimal Lipschitz constant) *The Lipschitz constant* L^* *of a solution of* $(*)$ *satisfies* $L^* \geq 2/d(X^+, X^-)$.

Proof. For a solution f of $(*)$ we have

$$\begin{aligned} L(f) &= \sup_{x,y \in \mathcal{X}} \frac{|f(x) - f(y)|}{d(x,y)} \geq \max_{i,j=1,\dots,n} \frac{|f(x_i) - f(x_j)|}{d(x_i, x_j)} \\ &\geq \max_{i,j=1,\dots,n} \frac{|y_i - y_j|}{d(x_i, x_j)} = \frac{2}{\min_{x_i \in X^+, x_j \in X^-} d(x_i, x_j)} = \frac{2}{d(X^+, X^-)}. \end{aligned}$$

☺

Lemma 13 (Solutions of $(*)$) *Let* $L^* = 2/d(X^+, X^-)$. *For all* $\alpha \in [0,1]$, *the following functions solve* $(*)$*:*

$$f_\alpha(x) := \alpha \min_i (y_i + L^* d(x, x_i)) + (1 - \alpha) \max_i (y_i - L^* d(x, x_i))$$

$$g(x) := \frac{d(x, X^-) - d(x, X^+)}{d(X^+, X^-)}$$

Proof. By Lemma 8, f_α has Lipschitz constant L^* and satisfies $f_\alpha(x_i) = y_i$. Moreover, it is easy to see that $y_i g(x_i) \geq 1$. Using the properties of Lipschitz constants stated in Section 5 and the fact that the function $d(x, \cdot)$ has Lipschitz constant 1 we see that $L(g) \leq L^*$. Thus f_α and g are solutions of $(*)$ by Lemma 12. ☺

The functions f_α and g lie in the vector lattice spanned by $\mathcal{D}_{\text{train}}$. As g is a linear combination of distances to sets of training points we have proved Theorem 11.

It is interesting to have a closer look at the functions of Lemma 13. The functions f_0 and f_1 are the smallest and the largest functions, respectively, that solve problem $(*)$ with equality in the constraints: any function f that satisfies $f(x_i) = y_i$ and has Lipschitz constant L^* satisfies $f_0(x) \leq f(x) \leq f_1(x)$. The functions g and $f_{1/2}$ are especially remarkable:

Lemma 14 (1-nearest neighbor classifier) *The functions* g *and* $f_{1/2}$ *defined above have the sign of the* 1*-nearest neighbor classifier.*

Proof. It is obvious that $g(x) > 0 \iff d(x, X^+) < d(x, X^-)$ and $g(x) < 0 \iff d(x, X^+) > d(x, X^-)$. For the second function, we rewrite $f_{1/2}$ as follows:

$$f_{1/2}(x) = \frac{1}{2}(\min(L^* d(x, X^+) + 1, L^* d(x, X^-) - 1) - \min(L^* d(x, X^+) - 1, L^* d(x, X^-) + 1)) .$$

Consider x such that $d(x, X^+) \geq d(x, X^-)$. Then $d(x, X^+) + 1 \geq d(x, X^-) - 1$ and thus

$$f_{1/2}(x) = \frac{1}{2}\left(L^* d(x, X^-) - 1 - \min(L^* d(x, X^+) - 1, L^* d(x, X^-) + 1)\right) \leq 0 .$$

The same reasoning applies to the situation $d(x, X^+) \leq d(x, X^-)$ to yield $f_{1/2}(x) \geq 0$ in this case. ☺

Note that g needs not reach equality in the constraints on all the data points, whereas the function $f_{1/2}$ always satisfies equality in the constraints. Lemma 14 has the surprising consequence that according to Section 6, the 1-nearest neighbor classifier actually is a large margin classifier.

7.4 Negative results

So far we have proved that $(*)$ always has a solution which can be expressed as a linear combination of distances to sets of training points. But maybe we even get a theorem stating that we always find a solution which is a linear combination of distance functions to single training points? Unfortunately, in the metric space setting such a theorem is not true in general. This can be seen by the following counterexample:

Example 1 *Assume four training points x_1, x_2, x_3, x_4 with distance matrix*

$$D = \begin{pmatrix} 0 & 2 & 1 & 1 \\ 2 & 0 & 1 & 1 \\ 1 & 1 & 0 & 2 \\ 1 & 1 & 2 & 0 \end{pmatrix}$$

and label vector $y = (1, 1, -1, -1)$. Then the set

$$\{f : \mathcal{X} \to \mathbb{R} |\ y_i f(x_i) \geq 1,\ f(x) = \sum_{i=1}^{4} a_i d(x_i, x) + b\}$$

is empty. The reason for this is that the distance matrix is singular and we have $d(x_1, \cdot) + d(x_2, \cdot) = d(x_3, \cdot) = d(x_4, \cdot)$. Hence, in this example, $(*)$ has no solution which is a linear combination of distances to single training points. But it still has a solution as linear combination of distances to sets of training points according to Theorem 11.

Another negative result is the following. Assume that instead of looking for solutions of $(*)$ in the space of all Lipschitz functions we only consider functions in the vector space spanned by $\mathcal{D}$. Is it in this case always possible to find solution in the linear span of $\mathcal{D}_{train}$? The answer is no again. An example for this is the following:

Example 2 *Let $\mathcal{X} = \{x_1, ..., x_5\}$ consist of five points with distance matrix*

$$D = \begin{pmatrix} 0 & 2 & 1 & 1 & 1 \\ 2 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 2 & 1 \\ 1 & 1 & 2 & 0 & 2 \\ 1 & 1 & 1 & 2 & 0 \end{pmatrix}.$$

Let the first four points be training points with the label vector $y = (-1, -1, -1, 1)$. As above there exists no feasible function in the vector space spanned by $\mathcal{D}_{train}$. But as the distance matrix of all five points is invertible, there exist feasible functions in the vector space spanned by $\mathcal{D}$.

In the above examples the problem was that the distance matrix on the training points was singular. But there are also other sources of problems that can occur. In particular it can be the case that the Lipschitz constant of a function restricted to the training set takes the minimal value L^*, but the Lipschitz constant on the whole space $\mathcal{X}$ is larger. Then it can happen that although we can find a linear combination of distance functions that satisfies $f(x_i) = y_i$, the function f has a Lipschitz constant larger than L^* and thus is no solution of $(*)$. An example for this situation is the following:

Example 3 *Let $\mathcal{X} = \{x_1, ..., x_5\}$ consist of five points with distance matrix*

$$D = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 2 \\ 1 & 1 & 0 & 2 & 1 \\ 1 & 1 & 2 & 0 & 1 \\ 1 & 2 & 1 & 1 & 0 \end{pmatrix}.$$

Let the first four points be training points with the label vector $y = (1, 1, -1, -1)$. The optimal Lipschitz constant in this problem is $L^* = 2/d(X^+, X^-) = 2$. The function $f(x) = -2d(x_1, x) - 2d(x_2, x) + 3$ has this Lipschitz constant if we evaluate it on the training points only. But if we also consider x_5, the function has Lipschitz constant 4.

These examples show that, in general, Theorem 11 cannot be improved to work in the vector space instead of the vector lattice spanned by $\mathcal{D}_{\text{train}}$. This also holds if we consider some subspaces of the set of Lipschitz functions. Thus we are in the interesting situation that it is not enough to consider distance functions to single training points – we have to deal with distances to sets of training points.

8 Error Bounds

In this section we compute error bounds for the Lipschitz classifier using Rademacher averages (recall the definition in Section 1). We will describe two different ways to compute Rademacher averages for sets of Lipschitz functions. One way is a classical approach using entropy numbers and leads to an upper bound on R_n. For this approach we always assume that the metric space $(\mathcal{X}, d)$ is precompact (i.e., it can be covered by finitely many balls of radius ε for every $\varepsilon > 0$).
The other way is more elegant: because of the definition of $\|\cdot\|_L$ and the resulting isometries, the maximum discrepancy of a $\|\cdot\|_L$-unit ball of $\mathrm{Lip}(\mathcal{X})$ is the same as of the corresponding unit ball in $AE(\mathcal{X}_0)'$. Hence it will be possible to express $\tilde{R}_n$ as the norm of an element of the Arens-Eells space. This norm can then be computed via bipartite minimal matching. In the following, B always denotes the unit ball of the considered function space.

8.1 The duality approach

The main insight to compute the maximum discrepancy by the duality approach is the following observation:

$$\sup_{\|f\|_L \le 1} |\sum_{i=1}^{n} f(x_i) - f(x_i')| = \sup_{\|T_f\| \le 1} |\sum_{i=1}^{n} T_f m_{x_i} - T_f m_{x_i'}| =$$

$$= \sup_{\|T_f\| \le 1} |\langle T_f, \sum_{i=1}^{n} m_{x_i} - m_{x_i'} \rangle| = \|\sum_{i=1}^{n} m_{x_i x_i'}\|_{AE}$$

Applying this to the definition of the maximum discrepancy immediately yields

$$\tilde{R}_n(B) = \frac{1}{n} E \|\sum_{i=1}^{n} m_{X_i X_i'}\|_{AE}. \tag{12}$$

As we already explained in Section 5, the norm $\|\sum_{i=1}^{n} m_{X_i X_i'}\|_{AE}$ can be interpreted as the costs of a minimal bipartite matching between $\{X_1, \ldots, X_n\}$ and $\{X_1', \ldots, X_n'\}$. To compute the right hand side of (12) we need to know the expected value of random instances of the bipartite minimal matching problem, where we assume that the points X_i and X_i' are drawn iid from the sample distribution. In particular we want to know how this value scales with the number n of points as this indicates how fast we can learn. This question has been solved for some special cases of random bipartite matching. Let the random variable C_n describe the minimal bipartite matching costs for a matching between the points $X_1, \ldots, X_n$ and $X_1', \ldots, X_n'$ drawn iid according to some distribution P. In Dobric and Yukich (1995) it has been proved that for an arbitrary distribution on the unit square of $\mathbb{R}^d$ with $d \ge 3$ we have $\lim C_n/(n^{d-1/d}) = c > 0$ a.s. for some constant c. The upper bound $EC_n \le c\sqrt{n \log n}$ for arbitrary distributions on the unit square in $\mathbb{R}^2$ was presented in Talagrand (1992). These results, together with Equation (12), lead to the following maximum discrepancies:

Theorem 15 (Maximum discrepancy of unit ball of Lip($[0,1]^d$)) *Let $\mathcal{X} = [0,1]^d \subset \mathbb{R}^d$ with the Euclidean metric. Then the maximum discrepancy of the $\|\cdot\|_L$-unit ball B of $\mathrm{Lip}(\mathcal{X})$ satisfies*

$$\begin{aligned} &\tilde{R}_n(B) \le c_2\sqrt{\log n}/\sqrt{n} \quad \text{for all } n \in \mathbb{N} && \text{if } d = 2 \\ &\lim_{n\to\infty} \tilde{R}_n(B)\, \sqrt[d]{n} = c_d > 0 && \text{if } d \ge 3 \end{aligned}$$

where c_d ($d \ge 2$) are constants which are independent of n but depend on d.

Note that this procedure gives (asymptotically) exact results rather than upper bounds in cases where we have (asymptotically) exact results on the bipartite matching costs. This is for example the case for cubes in $\mathbb{R}^d, d \ge 3$ as Dobric and Yukich (1995) gives an exact limit result, or for $\mathbb{R}^2$ with the uniform distribution.

8.2 Generalized entropy bound

Recall that the covering number $N(\mathcal{X}, \varepsilon, d)$ of a totally bounded metric space $(\mathcal{X}, d)$ is the smallest number of balls of radius ε with centers in $\mathcal{X}$ which can cover $\mathcal{X}$ completely. A classical theorem of Dudley (1987) bounds the Rademacher complexity in terms of covering numbers:

Theorem 16 (Classical entropy bound) *For every class $\mathcal{F}$ of functions there exists a constant C such that*

$$\hat{R}_n(\mathcal{F}) \le \frac{C}{\sqrt{n}} \int_0^\infty \sqrt{\log N(\mathcal{F}, u, L_2(\mu_n))}\, du$$

where μ_n is the empirical distribution of the sample.

When we want to apply this bound to Lipschitz function classes in the next section we come upon the problem that the integral on the right hand side diverges in some situation. To avoid this we will now prove an adapted version of the entropy bound where this problem does not occur:

Theorem 17 (Generalized entropy bound) *Let $\mathcal{F}$ be a class of functions and $X_1, \dots, X_n$ iid sample points with empirical distribution μ_n. Then, for every $\varepsilon > 0$,*

$$\hat{R}_n(\mathcal{F}) \le 2\varepsilon + \frac{4\sqrt{2}}{\sqrt{n}} \int_{\varepsilon/4}^\infty \sqrt{\log N(\mathcal{F}, u, L_2(\mu_n))}\, du.$$

The idea of the proof of Theorem 17 is the following. Instead of bounding the Rademacher complexity on the whole set of functions $\mathcal{F}$, we first consider a maximal ε-separating subset $\mathcal{F}_\varepsilon$ of $\mathcal{F}$. This is a maximal subset such that all its points have distance at least ε to each other. To this special set we will then apply the classical entropy bound. Then we will estimate how much we loose by this procedure.

The following lemma can be found as Lemma 3.10 in Bousquet (2002) (for the definition of a separable process see also van der Vaart and Wellner 1996).

Lemma 18 (ε-separations of an empirical process) *Let $\{Z_t; t \in T\}$ be a separable stochastic process satisfying for $\lambda > 0$ the increment condition*

$$\forall s, t \in T : E\big(e^{\lambda(Z_t - Z_s)}\big) \leq e^{\lambda^2 c^2 d^2(s,t)/2}.$$

Let $\varepsilon \geq 0$ and $\delta > 0$. If $\varepsilon > 0$, let T_ε denote a maximal ε-separated subset of T and let $T_\varepsilon = T$ otherwise. Then for all t_0,

$$E\left(\sup_{t \in T_\varepsilon, d(t,t_0) \leq \delta} Z_t - Z_{t_0}\right) \leq 4\sqrt{2}c \int_{\varepsilon/4}^{\delta/2} \sqrt{\log N(T,u,d)}du.$$

To apply this lemma to the Rademacher complexity of a function class $\mathcal{F}$, we choose the index set $T = \mathcal{F}$, the fixed index $t_0 = f_0$ for some $f_0 \in \mathcal{F}$, the empirical process $Z_f = \frac{1}{n}\sum \sigma_i f(X_i)$, and $\delta \to \infty$. Note that the Rademacher complexity satisfies the increment condition of Lemma 18 with respect to the $L_2(\mu_n)$–distance with constant $c = \sqrt{n}$. Moreover, observe that $E(\sup_t Z_t - Z_{t_0}) = E(\sup_t Z_t) - E(Z_{t_0})$ and $E(Z_{t_0}) = E(\frac{1}{n}\sum \sigma_i f_0(X_i)) = 0$. Together with the symmetry of the distribution of Z_f we thus get the next lemma:

Lemma 19 (Entropy bound for ε-separations) *Let $(X_i)_{i=1,\dots,n}$ be iid training points with empirical distribution μ_n, $\mathcal{F}$ an arbitrary class of functions, and $\mathcal{F}_\varepsilon$ a maximal ε-separating subset of $\mathcal{F}$ with respect to $L_2(\mu_n)$- norm. Then*

$$E\left(\sup_{f \in \mathcal{F}_\varepsilon} \frac{1}{n}\Big|\sum_i \sigma_i f(X_i)\Big| \Big| X_1, \dots, X_n\right) \leq \frac{4\sqrt{2}}{\sqrt{n}} \int_{\varepsilon/4}^{\infty} \sqrt{\log N(\mathcal{F}, u, L_2(\mu_n))}\ du.$$

With this lemma we achieved that the integral over the covering numbers starts at $\varepsilon/4$ instead of 0 as it is the case in Theorem 16. The price we pay is that the supremum on the left hand side is taken over the smaller set $\mathcal{F}_\varepsilon$ instead of the whole class $\mathcal{F}$. Our next step is to bound the mistake we make by this procedure.

Lemma 20 *Let $\mathcal{F}$ be a class of functions and $\mathcal{F}_\varepsilon$ a maximal ε-separating subset of $\mathcal{F}$ with respect to $\|\cdot\|_{L_2(\mu_n)}$. Then $|R_n(\mathcal{F}) - R_n(\mathcal{F}_\varepsilon)| \leq 2\varepsilon$.*

Proof. We want to bound the expression

$$|R_n(\mathcal{F}) - R_n(\mathcal{F}_\varepsilon)| = E\frac{1}{n}\left|\sup_{f \in \mathcal{F}} |\sum \sigma_i f(X_i)| - \sup_{f \in \mathcal{F}_\varepsilon} |\sum \sigma_i f(X_i)|\right|.$$

First look at the expression inside the expectation, assume that the σ_i and X_i are fixed and that $\sup_{f \in \mathcal{F}} |\sum \sigma_i f(x_i)| = |\sum \sigma_i f^*(x_i)|$ for some function f^* (if f^* does not exist we additionally have to use a limit argument). Let $f_\varepsilon \in \mathcal{F}_\varepsilon$ such that $\|f^* - f_\varepsilon\|_{L_2(\mu_n)} \leq 2\varepsilon$. Then,

$$\begin{aligned}\frac{1}{n}\left|\sup_{f \in \mathcal{F}} |\sum \sigma_i f(x_i)| - \sup_{f \in \mathcal{F}_\varepsilon} |\sum \sigma_i f(x_i)|\right| &\leq \frac{1}{n}\left||\sum \sigma_i f^*(x_i)| - |\sum \sigma_i f_\varepsilon(x_i)|\right| \\ &\leq \frac{1}{n}\left|\sum \sigma_i (f^*(x_i) - f_\varepsilon(x_i))\right| \leq \|f^* - f_\varepsilon\|_{L_1(\mu_n)} \leq \|f^* - f_\varepsilon\|_{L_2(\mu_n)} \leq 2\varepsilon\end{aligned}$$

As this holds conditioned on all fixed values of σ_i and X_i we get the same for the expectation. This proves the lemma. ☺

To prove Theorem 17 we now combine lemmas 19 and 20.

8.3 Covering number approach

To apply this theorem we need to know covering numbers of spaces of Lipschitz functions. This can be found for example in Kolmogorov and Tihomirov (1961), pp.353–357.

Theorem 21 (Covering numbers for Lipschitz function balls) *For a totally bounded metric space $(\mathcal{X}, d)$ and the unit ball B of $(\mathrm{Lip}(\mathcal{X}), \|\cdot\|_L)$,*

$$2^{N(\mathcal{X},4\varepsilon,d)} \leq N(B,\varepsilon,\|\cdot\|_\infty) \leq \left(2\left\lceil\frac{2\operatorname{diam}(\mathcal{X})}{\varepsilon}\right\rceil+1\right)^{N(\mathcal{X},\frac{\varepsilon}{4},d)}.$$

If, in addition, $\mathcal{X}$ is connected and centered (i.e., for all subsets $A \subset \mathcal{X}$ with $\operatorname{diam}(A) \leq 2r$ there exists a point $x \in \mathcal{X}$ such that $d(x,a) \leq r$ for all $a \in A$),

$$2^{N(\mathcal{X},2\varepsilon,d)} \leq N(B,\varepsilon,\|\cdot\|_\infty) \leq \left(2\left\lceil\frac{2\operatorname{diam}(\mathcal{X})}{\varepsilon}\right\rceil+1\right)\cdot 2^{N(\mathcal{X},\frac{\varepsilon}{2},d)}.$$

Combining Theorems 17 and 21 and using $N(\mathcal{F},u,L_2(\mu_n)) \leq N(\mathcal{F},u,\|\cdot\|_\infty)$ now gives a bound on the Rademacher complexity of balls of $\mathrm{Lip}(\mathcal{X})$:

Theorem 22 (Rademacher complexity of unit ball of $\mathrm{Lip}(\mathcal{X})$) *Let $(\mathcal{X}, d)$ be a totally bounded metric space with diameter $\operatorname{diam}(\mathcal{X})$ and B the ball of Lipschitz functions with $\|f\|_L \leq 1$. Then, for every $\varepsilon > 0$,*

$$R_n(B) \leq 2\varepsilon + \frac{4\sqrt{2}}{\sqrt{n}}\int_{\varepsilon/4}^{4\operatorname{diam}(\mathcal{X})}\sqrt{N(\mathcal{X},\frac{u}{4},d)\log\left(2\left\lceil\frac{2\operatorname{diam}(\mathcal{X})}{u}\right\rceil+1\right)}\,du.$$

If, in addition, $\mathcal{X}$ is connected and centered, we have

$$R_n(B) \leq 2\varepsilon + \frac{4\sqrt{2}}{\sqrt{n}}\int_{\varepsilon/4}^{2\operatorname{diam}(\mathcal{X})}\sqrt{N(\mathcal{X},\frac{u}{2},d)\log 2+\log(2\left\lceil\frac{2\operatorname{diam}(\mathcal{X})}{u}\right\rceil+1)}\,du.$$

In our framework this is a nice result as the bound on the complexity of balls of $\mathrm{Lip}(\mathcal{X})$ only uses the metric properties of the underlying space $\mathcal{X}$.

Now we want to compare the results of Theorems 15 and 22 for two simple examples.

Example 4 (d-dimensional unit square, $d \geq 3$) *Let $\mathcal{X} = [0,1]^d \subset \mathbb{R}^d, d \geq 3$, with the Euclidean metric $\|\cdot\|_2$. This is a connected and centered space. In Theorem 15 we showed that $\tilde{R}_n(B)$ asymptotically scales as $1/\sqrt[d]{n}$, and this result cannot be improved. Now we want to check whether Theorem 22 achieves a similar scaling rate. To this end we choose $\varepsilon = 1/\sqrt[d]{n}$ (as we know that we cannot obtain a rate smaller than this) and use that the covering numbers of $\mathcal{X}$ have the form $N(\mathcal{X}, \varepsilon, \|\cdot\|_2) = c/\varepsilon^d$ (e.g., page 1 of Mendelson and Vershynin, 2003). After evaluating the second integral of Theorem 22 we find that $R_n(B)$ indeed scales as $1/\sqrt[d]{n}$.*

Example 5 (2-dimensional unit square) *Let $\mathcal{X} = [0,1]^2 \subset \mathbb{R}^2$ with the Euclidean metric. Applying Theorem 22 similar to Example 4 yields a bound on $R_n(B)$ that scales as $\log n/\sqrt{n}$.*

In case of Example 4 the scaling behavior of the upper bound on $R_n(B)$ obtained by the covering number approach coincides with the exact result for $\tilde{R}_n(B)$ derived in Theorem 15. In case of Example 5 the covering number result $\log n/\sqrt{n}$ is slightly worse than the result $\sqrt{\log(n)}/\sqrt{n}$ obtained in Theorem 15.

8.4 Complexity of Lipschitz RBF classifiers

In this section we want to derive a bound for the Rademacher complexity of radial basis function classifiers of the form

$$\mathcal{F}_{rbf} := \{f : \mathcal{X} \to \mathbb{R} |\ f(x) = \sum_{k=1}^{l} a_k g_k(d(p_k, x)),\ g_k \in \mathcal{G},\ l < \infty\}, \tag{13}$$

where $p_k \in \mathcal{X}$, $a_k \in \mathbb{R}$, and $\mathcal{G} \subset \mathrm{Lip}(\mathcal{X})$ is a (small) set of $\|\cdot\|_\infty$-bounded Lipschitz functions on $\mathbb{R}$ whose Lipschitz constants are bounded from below by a constant $c > 0$. As an example, consider $\mathcal{G} = \{g : \mathbb{R} \to \mathbb{R} |\ g(x) = \exp(-x^2/\sigma^2), \sigma \geq 1\}$. The special case $\mathcal{G} = \{id\}$ corresponds to the function class which is used by the linear programming machine. It can easily be seen that the Lipschitz constant of an RBF function satisfies $L(\sum_k a_k g_k(d(p_k, \cdot))) \leq \sum_k |a_k| L(g_k)$. We define a norm on $\mathcal{F}_{rbf}$ by

$$\|f\|_{rbf} := \inf \left\{ \sum_k |a_k| L(g_k);\ f = \sum_k a_k g_k(d(p_k, \cdot)) \right\}$$

and derive the Rademacher complexity of a unit ball B of $(\mathcal{F}_{rbf}, \|\cdot\|_{rbf})$. Substituting a_k by $c_k/L(g_k)$ in the expansion of f we get

$$\begin{aligned} \sup_{f \in B} |\sum_{i=1}^{n} \sigma_i f(x_i)| &= \sup_{\sum |a_k| L(g_k) \leq 1, p_k \in \mathcal{X}, g_k \in \mathcal{G}} |\sum_{i=1}^{n} \sigma_i \sum_{k=1}^{l} a_k g_k(d(p_k, x_i))| \\ &= \sup_{\sum |c_k| \leq 1, p_k \in \mathcal{X}, g_k \in \mathcal{G}} |\sum_{i=1}^{n} \sigma_i \sum_{k=1}^{l} \frac{c_k}{L(g_k)} g_k(d(p_k, x_i))| \end{aligned}$$

$$= \sup_{\sum |c_k| \leq 1, p_k \in \mathcal{X}, g_k \in \mathcal{G}} |\sum_{k=1}^{l} c_k \sum_{i=1}^{n} \sigma_i \frac{1}{L(g_k)} g_k(d(p_k, x_i))|$$

$$= \sup_{p \in \mathcal{X}, g \in \mathcal{G}} |\sum_{i=1}^{n} \sigma_i \frac{1}{L(g)} g(d(p, x_i))| \tag{14}$$

For the last step observe that the supremum in the linear expansion in the second last line is obtained when one of the c_k is 1 and all the others are 0. To proceed we introduce the notations $h_{p,g}(x) := g(d(p,x))/L(g)$, $\mathcal{H} := \{h_{p,g};\ p \in \mathcal{X}, g \in \mathcal{G}\}$, and $\mathcal{G}_1 := \{g/L(g);\ g \in \mathcal{G}\}$. We rewrite the right hand side of Equation (14) as

$$\sup_{p \in \mathcal{X}, g \in \mathcal{G}} |\sum_{i=1}^{n} \sigma_i \frac{1}{L(g)} g(d(p, x_i))| \quad = \quad \sup_{h_{p,g} \in \mathcal{H}} |\sum_{i=1}^{n} \sigma_i h_{p,g}(x_i)|$$

and thus obtain $R_n(B) = R_n(\mathcal{H})$. To calculate the latter we need the following:

Lemma 23 $N(\mathcal{H}, 2\varepsilon, \|\cdot\|_\infty) \leq N(\mathcal{X}, \varepsilon, d) N(\mathcal{G}_1, \varepsilon, \|\cdot\|_\infty)$.

Proof. First we observe that for $h_{p_1,g_1}, h_{p_2,g_2} \in \mathcal{H}$

$$\begin{aligned}
\|h_{p_1,g_1} - h_{p_2,g_2}\|_\infty &= \sup_{x \in \mathcal{X}} |\frac{g_1(d(p_1,x))}{L(g_1)} - \frac{g_2(d(p_2,x))}{L(g_2)}| \\
&\leq \sup_{x \in \mathcal{X}} \left(|\frac{g_1(d(p_1,x))}{L(g_1)} - \frac{g_1(d(p_2,x))}{L(g_1)}| + |\frac{|g_1(d(p_2,x))}{L(g_1)} - \frac{g_2(d(p_2,x))}{L(g_2)}| \right) \\
&\leq \sup_{x \in \mathcal{X}} |d(p_1,x) - d(p_2,x)| + \|\frac{g_1}{L(g_1)} - \frac{g_2}{L(g_2)}\|_\infty \\
&\leq d(p_1,p_2) + \|\frac{g_1}{L(g_1)} - \frac{g_2}{L(g_2)}\|_\infty =: d_{\mathcal{H}}(h_{p_1,g_1}, h_{p_2,g_2})
\end{aligned} \tag{15}$$

For the step from the second to the third line we used the Lipschitz property of g_1. Finally, it is easy to see that $N(\mathcal{H}, 2\varepsilon, d_{\mathcal{H}}) \leq N(\mathcal{X}, \varepsilon, d) N(\mathcal{G}_1, \varepsilon, \|\cdot\|_\infty)$. ☺

Plugging Lemma 23 in Theorem 17 yields the following Rademacher complexity:

Theorem 24 (Rademacher complexity of unit ball of $\mathcal{F}_{rbf}$) *Let B be the unit ball of $(\mathcal{F}_{rbf}, \|\cdot\|_{rbf})$, $\mathcal{G}_1$ the rescaled functions of $\mathcal{G}$ as defined above, and $w := \max\{\mathrm{diam}(\mathcal{X}, d), \mathrm{diam}(\mathcal{G}_1, \|\cdot\|_\infty)\}$. Then, for every $\varepsilon > 0$,*

$$R_n(B) \leq 2\varepsilon + \frac{4\sqrt{2}}{\sqrt{n}} \int_{\varepsilon/4}^{w} \sqrt{\log N(\mathcal{X}, \frac{u}{2}, d) + \log N(\mathcal{G}_1, \frac{u}{2}, \|\cdot\|_\infty)}\ du.$$

This theorem is a huge improvement compared to Theorem 22 as instead of the covering numbers we now have log-covering numbers in the integral. As an example consider the linear programming machine on $\mathcal{X} = [0,1]^d$. Because of $\mathcal{G} = \{id\}$, the second term in the square root vanishes, and the integral over the log-covering numbers of $\mathcal{X}$ can be bounded by a constant independent of ε. As result we obtain that in this case $R_n(B)$ scales as $1/\sqrt{n}$.

9 Choosing subspaces of $\mathrm{Lip}(\mathcal{X})$

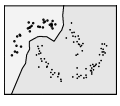
So far we always considered the isometric embedding of the given metric space into the Arens-Eells space and discovered many interesting properties of this embedding. But there exist many different isometric embeddings which could be used instead. Hence, the construction of embedding the metric space isometrically into some Banach space and then using a large margin classifier in this Banach space is also possible with different Banach spaces than the Arens-Eells space. For example, Hein and Bousquet (2003) used the Kuratowski embedding, which maps a metric space $\mathcal{X}$ isometrically in the space of continuous functions $(C(\mathcal{X}), \|\cdot\|_\infty)$ (see Example 6 below). Now it is a natural question whether there are interesting relationships between large margin classifiers constructed by the different isometric embeddings, especially with respect to the Lipschitz classifier.

A second question concerns the choice of subspaces of $\mathrm{Lip}(\mathcal{X})$. At the end of Section 6 we already explained that we have to work on some "reasonable" subspace of Lipschitz functions to apply the Lipschitz classifier in practice. This is justified by complexity arguments, but does the large margin interpretation still hold if we do this? Is there some geometric intuition which could help choosing a subspace?

It will turn out that both questions are inherently related to each other. We will show that there is a correspondence between embedding $\mathcal{X}$ into a Banach space V and constructing the large margin classifier on V on the one hand, and choosing a subspace F of $\mathrm{Lip}(\mathcal{X})$ and constructing the Lipschitz classifier from F on the other hand. Ideally, we would like to have a one-to-one correspondence between V and F. In one direction this would mean that we could realize any large margin classifier on any Banach space V with the Lipschitz classifier on an appropriate subspace F of Lipschitz functions. In the other direction this would mean that choosing a subspace F of Lipschitz functions corresponds to a large margin classifier on some Banach space V. We could then study the geometrical implications of a certain subspace F via the geometric properties of V.

Unfortunately, such a nice one-to-one correspondence between V and F is not always true, but in many cases it is. We will show that given an embedding into some vector space V, the hypothesis class of the large margin classifier on V always corresponds to a subspace F of Lipschitz functions (Lemma 28). In general, this correspondence will be an isomorphism, but not an isometry. The other way round, given a subspace F of Lipschitz functions, under some conditions we can construct a vector space V such that $\mathcal{X}$ can be isometrically embedded into V and the large margin classifiers on V and F coincide (Lemma 29).

The key ingredient in this section is the fact that $AE(\mathcal{X}_0)$ is a free Banach space. The following definition can be found for example in Pestov (1986).



Definition 25 (Free Banach space) *Let $(\mathcal{X}_0, d, e)$ be a pointed metric space. A Banach space $(E, \|\cdot\|_E)$ is a free Banach space over $(\mathcal{X}_0, d, e)$ if the following properties hold:*

1. *There exists an isometric embedding $\Phi : \mathcal{X}_0 \to E$ with $\Phi(e) = 0$, and E is the closed linear span of $\Phi(\mathcal{X}_0)$.*
2. *For every Banach space $(V, \|\cdot\|_V)$ and every Lipschitz map $\Psi : \mathcal{X}_0 \to V$ with $L(\Psi) = 1$ and $\Psi(e) = 0$ there exists a linear operator $T : E \to V$ with $\|T\| = 1$ such that $T \circ \Phi = \Psi$.*

It can be shown that the free Banach space over $(\mathcal{X}, d, e)$ always exists and is unique up to isomorphism (cf. Pestov, 1986).

Lemma 26 (AE is a free Banach space) *For any pointed metric space $(\mathcal{X}_0, d, e)$, $AE(\mathcal{X}_0)$ is a free Banach space.*

Proof. Property (1) of Definition 25 is clear by construction. For a proof of property (2), see for example Theorem 2.2.4 of Weaver (1999). ☺

We are particularly interested in the case where the mapping $\Psi : \mathcal{X}_0 \to V$ of Definition 25 is an isometric embedding of $\mathcal{X}_0$ into some vector space V. Firstly we want to find out under which conditions its dual V' is isometric isomorphic to some subspace F of $\mathrm{Lip}(\mathcal{X})$. Secondly, given a subspace F of $\mathrm{Lip}(\mathcal{X})$ the question is whether there exists a Banach space V such that $\mathcal{X}_0$ can be embedded isometrically into V and simultaneously V' is isometric isomorphic to F. Both questions will be answered by considering the mapping T of Definition 25 and its adjoint T'. The following treatment will be rather technical, and it might be helpful to have Figure 1 in mind, which shows which relations we want to prove.

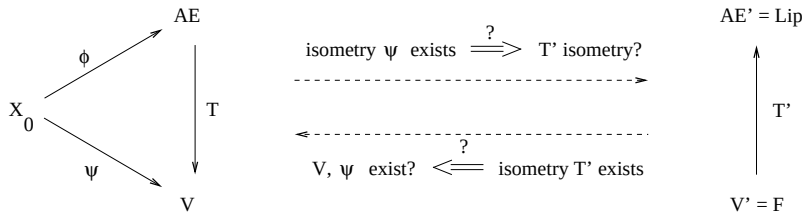


Figure 1: *Relations between Banach spaces and subspaces of Lipschitz functions. The left part shows the commutative diagram corresponding to the free Banach space property of $AE(\mathcal{X}_0)$. The right part shows the adjoint mapping T' of T. The dotted arrows in the middle show the relationships we want to investigate.*

Now we want to go into detail and start with the first question. For simplicity, we make the following definition.

Definition 27 (Dense isometric embedding) *Let $(\mathcal{X}_0, d)$ a metric space and V a normed space. A mapping $\Psi : \mathcal{X}_0 \to V$ is called a* dense isometric embedding *if Ψ is an isometry and if V is the norm-closure of* $\operatorname{span}\{\Psi(x); x \in \mathcal{X}_0\}$.

Lemma 28 (Construction of F for given V) *Let $(\mathcal{X}_0, d)$ be a pointed metric space, $(V, \|\cdot\|_V)$ a normed space and $\Psi : \mathcal{X}_0 \to V$ a dense isometric embedding. Then V' is isomorphic to a closed subspace $F \subset \operatorname{Lip}_0(\mathcal{X}_0)$, and the canonical injection $i : F \to \operatorname{Lip}_0(\mathcal{X}_0)$ satisfies $\|i\| \leq 1$.*

Proof. Recall the notation $m_x := \Phi(x)$ from Section 6 and analogously denote $v_x := \Psi(x)$. Let $T : AE(\mathcal{X}_0) \to V$ the linear mapping with $T \circ \Phi = \Psi$ as in Definition 25. As Ψ is an isometry, T satisfies $\|T\| = 1$, and maps $AE(\mathcal{X}_0)$ on some dense subspace of V. Consider the adjoint $T' : V' \to AE(\mathcal{X}_0)'$. It is well known (e.g., Chapter 4 of Rudin, 1991) that $\|T\| = \|T'\|$ and that T' is injective iff the range of T is dense. Thus, in our case T' is injective. As by construction also $\langle T m_x, v' \rangle = \langle T' v', m_x \rangle$, we have a unique correspondence between the linear functions in V' and some subspace $F := T'V' \subset AE(\mathcal{X}_0)'$: for $g \in V'$ and $f = T'g \in \operatorname{Lip}_0(\mathcal{X}_0)$ we have $g(v_x) = f(m_x)$ for every $x \in \mathcal{X}_0$. The canonical inclusion i corresponds to the adjoint T'. ☺

Lemma 28 shows that the hypothesis space V' constructed by embedding $\mathcal{X}$ into V is isomorphic to a subset $F \subset \operatorname{Lip}_0(\mathcal{X}_0)$. But it is important to note that this isomorphism is not isometric in general. Let $g \in V'$ and $f \in \operatorname{Lip}_0(\mathcal{X}_0)$ be corresponding functions, that is $f = T'g$. Because of $\|T'\| = 1$ we know that $\|f\|_{AE'} \leq \|g\|_V$, but in general we do not have equality. This means that the margins $\|g\|_{V'}$ and $\|f\|_{AE'}$ of corresponding functions are measured with respect to different norms and might have different sizes. As a consequence, the solutions of the two large margin problems

$$\min_{g \in V'} \|g\|_{V'} \text{ subject to } y_i g(v_{x_i}) \geq 1$$

and

$$\min_{f \in F} \|f\|_L \text{ subject to } y_i f(x_i) \geq 1$$

might be different, even though the sets of feasible functions are the same in both cases.

To illustrate this we will consider two examples. The first one shows how the large margin classifier in V can give different results than the one constructed by using the corresponding subspace for the Lipschitz classifier. In the second example we show a situation where both classifiers coincide.

Example 6 (Kuratowski embedding) *Let $(\mathcal{X}, d)$ be an arbitrary compact metric space and $(\mathcal{C}(\mathcal{X}), \|\cdot\|_\infty)$ the space of continuous functions on $\mathcal{X}$. Define $\Psi : \mathcal{X} \to \mathcal{C}(\mathcal{X})$, $x \mapsto d(x, \cdot)$. This mapping is an isometric embedding called Kuratowski*

embedding, and it has been used in Hein and Bousquet (2003) to construct a large margin classifier. We want to compare the large margin classifiers resulting from the Kuratowski embedding and the embedding in the Arens-Eells space. As an example consider the finite metric space $\mathcal{X} = \{x_1, ..., x_4\}$ *with distance matrix*

$$D = \begin{pmatrix} 0 & 5 & 3 & 6 \\ 5 & 0 & 4 & 1 \\ 3 & 4 & 0 & 5 \\ 6 & 1 & 5 & 0 \end{pmatrix}.$$

Let $V = \text{span}\{d(x, \cdot);\ x \in \mathcal{X}\} \subset C(\mathcal{X})$*, endowed with the norm* $\|\cdot\|_\infty$*. V is a 4-dimensional vector space. Let* V' *its dual space. Via the mapping* T'*, each linear operator* $g \in V'$ *corresponds to the linear operator* $f \in \text{Lip}_0(\mathcal{X}_0)$ *with* $f(x_i) = \langle g, d(x_i, \cdot)\rangle =: c_i$*. Now we want to compare the norms of g in* V' *and f in* $\text{Lip}(\mathcal{X})$*. The norm of g in* V' *can be computed as follows:*

$$\begin{aligned} \|g\|_{V'} &= \sup\{\langle g, v\rangle :\ v \in V,\ \|v\|_V \le 1\} \\ &= \sup\{\langle g, \sum_{i=1}^4 a_i d(x_i, \cdot)\rangle :\ a_i \in \mathbb{R},\ \|\sum_{i=1}^4 a_i d(x_i, \cdot)\|_\infty \le 1\} \\ &= \sup\{\sum_{i=1}^4 a_i c_i :\ a_i \in \mathbb{R},\ -1 \le \sum_{i=1}^4 a_i d(x_i, x_j) \le 1 \text{ for all } j = 1, ..., 4\ \}. \end{aligned}$$

For given function $g \in V'$ *(that is, for given values* c_i*) this norm can be computed by a linear program. Consider the two functions* $g_1, g_2 \in V'$ *with values on* x_1, x_2, x_3, x_4 *given as* $(-1, -1, -1, -1)$ *and* $(1, 0, 1, 0)$*, respectively, and let* $f_1, f_2 \in \text{Lip}_0(\mathcal{X}_0)$ *be the corresponding Lipschitz functions. Then we have* $\|f_1\|_L = 0.166 < 0.25 = \|f_2\|_L$ *and* $\|g_1\|_{V'} = 0.366 > 0.28 = \|g_2\|_{V'}$*. So the norms* $\|\cdot\|_{V'}$ *and* $\|\cdot\|_L$ *do not coincide, and moreover there is no monotonic relationship between them. If the maximal margin algorithm had to choose between functions* f_1 *and* f_2*, it would come to different solutions, depending whether the underlying norm is* $\|\cdot\|_{V'}$ *as for the large margin classifier in* V' *or* $\|\cdot\|_L$ *as for the Lipschitz classifier in* $T'V'$*.*

Example 7 (Normed space) *Let* $(\mathcal{X}, \|\cdot\|_\mathcal{X})$ *be a normed vector space with dual* $(\mathcal{X}', \|\cdot\|_{\mathcal{X}'})$*. As the norm of linear functions coincides with their Lipschitz constant,* $\mathcal{X}'$ *is* isometrically *isomorphic to a subspace of* $\text{Lip}_0(\mathcal{X}_0)$*. This means that it makes no difference whether we construct a large margin classifier on the normed space* $\mathcal{X}$ *directly or ignore the fact that* $\mathcal{X}$ *is a normed space, embed* $\mathcal{X}$ *into* $AE(\mathcal{X}_0)$ *and then construct the Lipschitz classifier on* $AE(\mathcal{X}_0)$ *with the subspace* $T'\mathcal{X}'$*. We already mentioned this fact in Section 6 when we stated that the SVM solution is the same one as the Lipschitz classifier on* $\mathcal{X}'$*.*

Now we want to investigate our second question: given some subspace $F \subset \text{Lip}_0(\mathcal{X}_0)$, is F the dual space of some Banach space V such that $\mathcal{X}_0$ can be embedded isometrically into V and $V' \simeq F$? To answer this question we have to deal with some

technical problems. First of all, F has to possess a *pre-dual*, that is a vector space V whose dual V' coincides with F. In general, not every Banach space possesses a pre-dual, and if it exists, it needs not be unique. Secondly, it turns out that the canonical injection $T' : F \to \mathrm{Lip}_0(\mathcal{X}_0)$ has to have a *pre-adjoint*, that is a mapping $T : AE(\mathcal{X}_0) \to V$ whose adjoint coincides with T'. Pre-adjoints also not always exist. In general, neither the existence of a pre-dual nor the existence of pre-adjoints are easy to prove. One situation where both can be handled is the case where F is closed under pointwise convergence:

Lemma 29 (Construction of V for given F) *Let $\mathcal{X}_0$ be a bounded metric space, and F a subspace of $(\mathrm{Lip}_0(\mathcal{X}_0), L(\cdot))$ which is closed under pointwise convergence and satisfies the condition*

$$\sup_{f \in F, L(f) \leq 1} |f(x) - f(y)| = d(x, y) \tag{16}$$

for all $x, y \in \mathcal{X}_0$. Then there exists a normed space V such that $\mathcal{X}_0$ can be isometrically embedded into V and its dual V' is isometrically isomorphic to F.

Before we can start with the proof we need two more definitions: Let M be a subspace of some Banach space V and N a subspace of the dual space V'. Then the annihilator $M^\perp$ and the pre-annihilator ${}^\perp N$ are defined as $M^\perp = \{T \in V';\ Tm = 0 \text{ for all } m \in M\}$ and ${}^\perp N = \{e \in V;\ Te = 0 \text{ for all } T \in N\}$. As the proof is a bit technical, we refer to Megginson (1998) for background reading.

Proof. For a bounded metric space $\mathcal{X}_0$, the topology of pointwise convergence on $\mathrm{Lip}_0(\mathcal{X}_0)$ coincides with its weak* topology. Thus by assumption, F is weak*-closed, which implies that ${}^\perp F$ is a closed subspace of $AE(\mathcal{X}_0)$. Hence, the quotient space $V := AE(\mathcal{X}_0)/{}^\perp F$ exists, and there exists an isometric isomorphism between V' and $({}^\perp F)^\perp$. As F is weak*-closed, $({}^\perp F)^\perp = F$. So V is a pre-dual of F. Let $T' : F \to \mathrm{Lip}_0(\mathcal{X}_0)$ be the canonical inclusion. It has a pre-adjoint, namely the quotient mapping $\pi : AE(\mathcal{X}_0) \to V$. Define the mapping $\Psi : \mathcal{X}_0 \to V$, $x \mapsto \pi m_x =: v_x$. We have

$$\langle f, v_x \rangle = \langle f, \pi m_x \rangle = \langle T' f, m_x \rangle = \langle f, m_x \rangle = f(x).$$

Hence, by assumption (16), Ψ is an isometry:

$$\|\Psi(x) - \Psi(y)\|_V = \sup_{f \in F, L(f) \leq 1} \{|\langle f, v_x - v_y \rangle|\} = \sup_{f \in F, L(f) \leq 1} \{|f(x) - f(y)|\} = d(x, y).$$

☺

Lemma 29 gives a nice interpretation of what it means geometrically to choose a subspace F of Lipschitz functions: the Lipschitz classifier with hypothesis space F corresponds to embedding $\mathcal{X}$ isometrically into the pre-dual V of F and constructing the large margin classifier on V directly. Condition (16), which F has to satisfy to allow this interpretation, intuitively means that F has to be a "reasonably large" subspace.

Example 8 (Linear combination of distance functions) *Let F be the subspace of* $\mathrm{Lip}(\mathcal{X})$ *consisting of functions of the form* $f(x) = \sum_i a_i d(x_i, x) + b$*, and* $\bar{F} \subset \mathrm{Lip}(\mathcal{X})$ *its closure under pointwise convergence. As norm on* $\bar{F}$ *we take the Lipschitz constant. On* $\bar{F}$*, condition* (16) *is satisfied: trivially, we always have* $\leq$ *in* (16)*, and for given* $x, y \in \mathcal{X}$*, equality is reached for the function* $f = d(x, \cdot)$*. So we can conclude by Lemma 29 that the Lipschitz classifier on* $\bar{F}$ *has the geometrical interpretation explained above.*

10 Discussion

We derived a general approach to large margin classification on metric spaces which uses Lipschitz functions as decision functions. Although the Lipschitz algorithm, which implements this approach, has been derived in a rather abstract mathematical framework, it boils down to an intuitively plausible mechanism: it looks for a decision function which has a small Lipschitz constant. This agrees with the regularization principle that tries to avoid choosing functions with a high variation. The solution of the Lipschitz algorithm is well behaved as, by the representer theorems of Section 7, it can always be expressed by distance functions to training points. For some special cases, the solution corresponds to solutions of other well known algorithms, such as the support vector machine, the linear programming machine, or the 1-nearest neighbor classifier. We provide Rademacher complexity bounds for some of the involved function classes which can be used to bound the generalization error of the classifier.

In spite of all those nice properties there are several important questions which remain unanswered. To apply the Lipschitz algorithm in practice it is important to choose a suitable subspace of Lipschitz functions as hypothesis space. In Section 9 we found a geometrical explanation of what the choice of certain subspaces F means: it is equivalent to using a different isometric embedding of the metric space into some Banach space. But this explanation does not solve the question of which subspace we should choose in the end. Moreover, there exist isometric embeddings in certain Banach spaces which have no such interpretation in terms of subspaces of Lipschitz functions. For example, Hein and Bousquet (2003) studied the Kuratowski embedding of a metric space into its space of continuous functions to construct a large margin algorithm. As we explained in Example 6, the large margin classifier resulting from this embedding can be different from the Lipschitz classifier. It is an interesting question how different embeddings into different Banach spaces should be compared. One way to do this could be comparing the capacities of the induced function spaces. An interesting question in this context is to find the "smallest space" (for instance, in terms of the Rademacher complexities) in which a given data space can be embedded isometrically.

There is also a more practical problem connected to the choice of the subspace

of Lipschitz functions. To implement the Lipschitz algorithm for a given subspace of Lipschitz functions, we need to know some way to efficiently compute the Lipschitz constants of the functions in the chosen subspace. For example, in case of the linear programming machine it was possible to bound the Lipschitz constants of the functions in the parameterized subspace of functions $\sum_i a_i d(x_i, \cdot) + b$ in terms of their parameters by $\sum_i |a_i|$. But in many cases, there is no obvious parametric representation of the Lipschitz constant of a class of functions. Then it is not clear how the task of minimizing the Lipschitz constant can be efficiently implemented.

An even more heretic question is whether isometric embeddings should be used at all. In our approach we adopted the point of view that a meaningful distance function between the training points is given by some external knowledge, and that we are not allowed to question it. But in practical applications it is often the case that distances are estimated by some heuristic procedure which might not give a sensible result for all the training points. In those cases the paradigm of isometric embedding might be too strong. Instead we could look for bi-Lipschitz embeddings or low distortion embeddings of the metric space into some Banach space, or even into some Hilbert space. We would then loose some (hopefully unimportant) information on the distances in the metric space, but the gain might consist in a simpler structure of the classification problem in the target space.

Finally, many people argue that for classification only "local properties" should be considered. One example is the assumption that the data lies on some low dimensional manifold in a higher dimensional space. In this case, the meaningful information consists of the intrinsic distances between points along the manifold. In small neighborhoods, those distances are close to the distances measured in the enclosing space, but for points which are far away from each other this is not true any more. In this setting it is not very useful to perform an isometric embedding of the metric space into a Banach space as the additional linear structure the Banach space imposes on the training data might be more misleading than helpful. Here a different approach has to be taken, but it is not clear how a large margin algorithm in this setting can be constructed, or even whether in this case the large margin paradigm should be applied at all.

Chapter IV

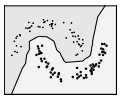
A Compression Approach to Support Vector Model Selection

In the last chapters we studied the behavior of machine learning algorithms for a fixed similarity or dissimilarity function. The topic of this chapter is how the parameters of a similarity function can be determined from the given training data. To make an optimal choice of parameters we first have to decide which learning algorithm we want to use. In this chapter this will be simply a support vector machine with Gaussian kernels. For this algorithm we will then interpret the constructed classifiers with methods from information theory and data compression. This will lead to an implementation of Ockham's razor: we choose the set of parameters such that the resulting classifier is as "simple to describe" as possible. This will be done by constructing "compression coefficients" which measure how well the information about the training labels can be encoded with the help of a given hyperplane. The main idea is to relate the coding precision to geometrical concepts such as the width of the margin or the shape of the data in the feature space. These compression coefficients will then be tested in model selection experiments. As a result we find that compression coefficients can fairly accurately predict the parameters for which the test error is minimized.

1 Model selection via compression bounds

1.1 Model selection

In statistical learning theory, a model $\mathcal{M}$ is simply a class of functions. For a given model, the learning step consists in choosing the function from the model which best explains our training data. In many applications, it is not clear from the beginning which kind of function class is appropriate to model the data. If the model is too simple, it might not contain a good classifier at all. If the model is too complex it is very likely that we will overfit. The term model selection now refers to the problem of choosing a model from a given class of models $(\mathcal{M}_i)_{i \in I}$ which is well suited to our



training data.

In the case of hard margin support vector machines, the model class only depends on the kernel function we choose. Each kernel function k gives rise to a model $\mathcal{M}_k := \{\sum_i a_i k(x_i, \cdot) + b;\ a_i, b \in \mathbb{R}\}$. In this context, model selection is the problem of choosing a good kernel function. Often we already know what the type of the kernel function we want to use, for example the Gaussian kernel $k_\sigma(x, y) = \exp(-\|x - y\|^2/\sigma^2)$. Here the class of models is a parametric class, each particular value of σ corresponding to some model $\mathcal{M}_\sigma$. Model selection then reduces to choosing the best parameter σ. For the soft margin support vector machine, additionally to the kernel function we also have to determine the soft margin parameter C (recall the soft margin optimization problem in Section III.3). Choosing both σ and C is the model selection problem we will study in this chapter.

One standard way to perform model selection is to use model complexity estimates and generalization bounds. The model complexity assesses "how large" a function class is. Examples for such complexity measures are covering numbers of the function space, the VC dimension, or the Rademacher complexity. The generalization bounds derived in statistical learning theory usually have the form

$$\mathcal{R}(f) \leq \mathcal{R}_{\text{emp}}(f) + \mathcal{C}(\mathcal{F})$$

where $\mathcal{F}$ is the given function class, $\mathcal{C}$ some complexity measure of it, and f a function in $\mathcal{F}$ (which might be chosen by a certain algorithm). Given the training data, such a bound can be used for model selection by choosing the model $\mathcal{F}$ which minimizes the right hand side $\min_{f \in \mathcal{F}} \mathcal{R}_{\text{emp}}(f) + \mathcal{C}(\mathcal{F})$ over the given set of models. To make this feasible in practice, the complexity measure $\mathcal{C}$ has to be easy to compute. Often this is not the case, for example if the complexity is expressed in terms of covering numbers of the function space.

1.2 Data compression and learning

One particular measure of complexity of a function class can be obtained by data compression arguments. The general idea is that the complexity of a model (with respect to the given training data) is low if the information contained in the training data can be compressed effectively with the help of the model. Following the conventions in the compression literature, models will also be called hypothesis spaces in the following, and the terms "classifier" and "hypothesis" will be used synonymously.

In the framework of Vapnik (1998, Section 6.2), classifiers are used to construct codes that can transmit the labels $y_1, ..., y_n$ of a set of training patterns $x_1, ..., x_n$. What those codes essentially do is to tell the receiver which classifier h from a given hypotheses class he should use to reconstruct the training labels from the training

patterns (this is described in detail in Section 2). For such a code, the *compression coefficient* $C(h)$ is defined as

$$C(h) := \frac{\text{number of bits to code } y_1, ..., y_n \text{ using } h}{n}. \quad (1)$$

The denominator corresponds to the number of bits we need to transmit the uncompressed binary vector $(y_1, ..., y_n)$. The numerator tells us how many bits we need to transmit the same information using the code constructed with help of classifier h. Hence, the compression coefficient is a number between 0 and 1 which describes how efficient the code works. Intuitively, if the compression coefficient $C(h)$ is small, h is a "simple" hypothesis which we expect not to overfit too much and hence to have a small generalization error. This belief is supported by the following theorem, which bounds the risk $R(h)$ of a classifier h with respect to the 0-1-loss in terms of the compression coefficient:

Theorem 1 (Section 6.2 in Vapnik 1998) *With probability at least $1-\eta$ over n random training points drawn iid according to the unknown distribution P, the risk $R(h)$ of classifier h is bounded by*

$$R(h) \leq 2\ln(2)C(h) - \frac{\ln(\eta)}{n}$$

simultaneously for all classifiers h in a finite hypotheses set.

This bound has the disadvantage that it is only valid in the restricted setting where the hypotheses space is finite and independent of the training data.

A different bound that directly works in the coding setting has been stated by Blum and Langford (2003). Their setting is slightly different from the one of Vapnik. For a given set of training and test points, the sender constructs a code h that transmits the labels of both training and test points. R_{test} is then defined as the error which the code h makes on the given test set.

Theorem 2 (Corollary 3 in Blum and Langford 2003) *With probability at least $1-\eta$ over n random training points and n random test points drawn iid according to the unknown distribution P, and for all codes h which encode the training labels without error, the error $R_{test}(h)$ on the test set satisfies*

$$R_{test}(h) \leq C(h) - \frac{\ln \eta}{n}.$$

The advantage of this bound is that the problem of data dependency does not occur. It is proved within the coding framework, without assuming a fixed hypotheses space. It is valid for all codes, as long as receiver and sender agree on how the code works before the sender knows the training data. In particular, the codes may depend on the training data in some predefined way, as it will be the case for the codes we are going to construct.

In general it is well-known that there is a tight connection between learning and coding. In some way, the process of learning corresponds to finding efficient codes to describe the input space. In the minimum description length (MDL) framework (cf. Barron et al., 1998, and references therein), this connection is made explicit. The MDL principle states that, among a given set of hypotheses, one should choose the hypothesis that achieves the shortest description of the training data. Intuitively, this seems to be a reasonable choice: by Shannon's source coding theorem (cf. Cover and Thomas, 1991) we know that an efficient code is closely related to the data generating distribution. Moreover, an easy-to-describe hypothesis is less likely to overfit than a more complicated one.

There are several connections between the MDL principle and other learning methods. It can be shown that selecting the hypothesis with the highest posterior probability in a Bayesian setting is equivalent to choosing the hypothesis with the shortest code (cf. Hansen and Yu, 2001). For SVMs, the compression scheme approach of Floyd and Warmuth (1995), which describes the generalization of a learning algorithm by its ability to reduce the training set to a few important points, leads to a bound in terms of the number of support vectors. Combining this result with large margin bounds yields the sparse margin bound of Herbrich et al. (2000). In McAllester (1999), a PAC-Bayesian bound for learning algorithms was derived. For a given prior distribution P on the hypotheses space, it bounds the generalization error essentially by the quantity $-\ln P(U)/m$, where U is the subset of hypotheses consistent with the training examples. Intuitively we can argue that, according to Shannon's theorem, $-\ln P(U)$ corresponds to the length of the shortest code for this subset.

In the following we want to explore whether the connection between statistical learning theory and data compression is only of theoretical interest or whether it can also be exploited for practical purposes. Our idea is to use the compression coefficient bounds above for model selection. The main work will consist in finding a way to actually compute compression coefficients for a given model and given training data. The special case we will consider is model selection for soft margin SVMs. The first part of the work (Section 2) will consist in using hyperplanes learned by SVMs to construct codes for the training labels. Those codes work by encoding the direction of the separating hyperplane. To make them efficient, we will use geometric concepts such as the size of the margin or the shape of the data in the feature space, and sparsity arguments. The main insight of this section is on how to transform those geometrical concepts into an actual code for labels. In the end we obtain compression coefficients that contain quantities already known to be meaningful in the statistical learning theory of SVMs, such as the radius-margin term R^2/ρ^2, the eigenvalues of the kernel matrix, and the number of support vectors. In the second part (Section 3) we then test the model selection performance of our compression coefficients on benchmark data sets. We find that the compression coefficients perform comparable or better than several standard bounds.

Now we want to establish some notation for this chapter. For a given kernel function k we denote the kernel matrix by $K := (k(x_i, x_j))_{i,j=1,...,n}$. We will denote its eigenvalues by $\lambda_1, ..., \lambda_n$, where the λ_i are sorted in non-increasing order. Later on we will also consider the kernel matrix K_{SV} restricted to the span of the support vectors. To define it, let $\{x_{i_1}, ..., x_{i_s}\} \subset \{x_1, ..., x_n\}$ the set of support vectors, $SV := \{k \mid x_k \in \text{span}\{x_{i_1}, ..., x_{i_s}\}\}$ the indices of those training points which are in the subspace spanned by the support vectors. Then the kernel matrix restricted to the span of the support vectors is defined as $K_{SV} := (k(x_i, x_j))_{i,j \in SV}$.

The sphere S_R^{d-1} is the surface of a ball with radius R in the space $\mathbb{R}^d$. The function log will always denote the logarithm to the base 2. Code lengths will often be given by some logarithmic term, for instance $\lceil \log n \rceil$. To keep the notations simple, we will omit the ceil brackets and simply write $\log n$.

2 Compression Coefficients for SVMs

The basic setup for the compression coefficient framework is the following. We are given n pairs $(x_i, y_i)_{i=1,...,n}$ of training patterns with labels and assume that an imaginary sender and receiver both know the training *patterns*. It will be the task of the sender to transmit the *labels* of the training patterns to the receiver. This reflects the basic structure of a classification problem: we want to predict the labels y for given patterns x. That is, we want to learn something about $P(y|x)$. Sender and receiver are allowed to agree on the details of the code before transmission starts. Then the sender gets the training data and chooses a classifier that separates the training data. He transmits to the receiver which classifier he chose. The receiver can then apply this classifier to the training patterns and reconstruct all the labels.

To understand how this works let us consider a simple example. Before knowing the data, sender and receiver agree on a finite hypotheses space containing k hypotheses $h_1, ..., h_k$. The sender gets the training patterns and the labels, and the receiver is allowed to look at the training patterns only. Now the sender inspects the training data. For simplicity, let us first assume that one of the hypotheses, say h_7, classifies all training points correctly. In this case the sender transmits "7" to the receiver. The receiver can now reconstruct the labels of the training patterns by classifying them according to hypothesis h_7. Now consider the case where there is no hypothesis that classifies all training patterns correctly. In this case, the receiver cannot reconstruct all labels without error if the sender only transmits the hypothesis. Additionally he has to know which of the training points are misclassified by this hypothesis. In our example, assume that hypothesis 7 misclassifies the training points 3, 14, and 20. The information the sender now transmits is "hypothesis: 7; misclassified points: 3, 14, 20". The receiver can then construct the labels of all patterns according to h_7 and flip the labels he obtained for patterns 3, 14, and 20. After this, he has labeled all training patterns correctly.

This example shows how a classification hypothesis can be used to transmit the labels of training points. In general, a finite, fixed hypothesis space as it was used

in the example may not contain a good hypothesis for previously unseen training points and will result in long codes. To avoid this, the sender will have to adapt the hypothesis space to the training points and communicate this to the receiver.

One principle that can already be observed in the example above is the way training errors are handled. As above, the code will always consist of two parts: the first part which serves to describe the hypothesis, and the second part, which tells the receiver which of the training points are misclassified by this hypothesis.

2.1 Relation between margin and coding precision

In the following we want to investigate how SVM hyperplanes can be used for coding the labels. The main part of those codes will consist in transmitting the direction of the separating hyperplane constructed by the SVM. We will always consider the simplified problem where the hyperplane goes through the origin. The hypotheses space consists of all possible directions the normal vector can take. It can be identified with the unit sphere in the feature space. In case of an infinite dimensional feature space, recall that by the representer theorem the solution of an SVM always lies in the subspace spanned by the training points. Thus the normal vector we want to code is a vector in a Hilbert space of dimension at most m (where m is the number of training points). The trick to determine the precision by which we have to code this vector is to interpret the margin in terms of coding precision: Suppose the data are (correctly) classified by a hyperplane with normal vector ω and margin ρ. A fact that often has been observed (e.g., Schölkopf and Smola, 2002, p. 194) is that in case of a large margin, small perturbations of the direction of the hyperplane will not change the classification result on the training points. In compression language this means that we do not have to code the direction of the hyperplane with high accuracy – the larger the margin, the less accurate we have to code. So we will adapt the precision by which we code this direction to the width of the margin. Suppose that all training patterns lie in a ball of radius R around the origin and that they are separated by a hyperplane H through the origin with normal vector ω and margin ρ. Then every "slightly rotated" hyperplane that still lies within the margin achieves the same classification result on all training points as the original hyperplane (cf. Figure 1a). Thus, instead of using the hyperplane with normal vector ω to separate the training points we could use any convenient vector v as normal vector – as long as the corresponding hyperplane still remains inside the margin. In this context, note that rotating a hyperplane by some angle α corresponds to rotating its normal vector by the same angle. We denote the set of normal vectors such that the corresponding hyperplanes still lie inside the margin the *rotation region* of ω. The region in which the corresponding hyperplanes lie will be called the *rotation region of* H (cf. Figures 1a and b).

To code the direction of ω, we will construct a discrete set of "codebook vectors" on the sphere. An arbitrary vector will be coded by choosing the closest codebook vector. From the preceding discussion we can see that the set of codebook vectors has to be constructed in such a way that the closest codebook vector for every possible

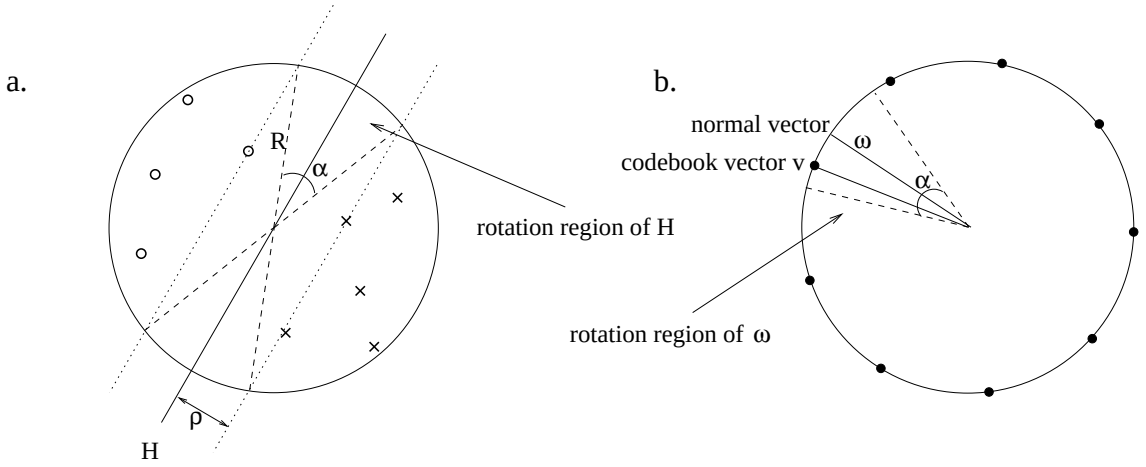


Figure 1: (a) The training points are separated by hyperplane H with margin ρ. If we change the direction of H such that the new hyperplane still lies inside the rotation region determined by the margin (dashed lines), the new hyperplane obtains the same classification result on the training points as H. (b) The hyperplanes in the rotation region of H (indicated by the dashed lines) correspond to normal vectors inside the rotation region of ω. The black points indicate the positions of equidistant codebook vectors on the sphere. The distance between those vectors has to be chosen so small that in each cone of angle α there is at least one codebook vector. In this example, vector v is the codebook vector closest to normal vector ω, and by construction it lies inside the rotation region of ω.

normal vector ω is inside the rotation region of ω (cf. Figure 1b). An equivalent formulation is to construct a set of points on the surface of the sphere such that the balls of radius ρ centered at those points cover the sphere. The minimal number of balls we need to achieve this is called the covering number of the sphere.

Proposition 3 (Covering numbers of spheres) *The number u_d of balls of radius ρ which are required to cover the sphere S_R^{d-1} of radius R in d-dimensional Euclidean space ($d \geq 2$) satisfies*

$$\left(\frac{R}{\rho}\right)^{d-1} \leq u_d \leq 2\left\lceil\frac{R\pi}{\rho}\right\rceil^{d-1}.$$

The constant in the upper bound can be improved, but as we will be interested in log-covering numbers later, this does not make much difference in our application.

Proof. We prove the upper bound by induction. For $d = 2$, the sphere is a circle which can be covered with $\lceil 2R\pi/\rho\rceil \leq 2\lceil R\pi/\rho\rceil$ balls. Now assume the proposition is true for the sphere S_R^{d-1}. To construct a ρ-covering on S_R^d we first cover the cylinder $S_R^{d-1} \times [-R\pi/2, R\pi/2]$ with a grid of $\tilde{u}_{d+1} := u_d \cdot \lceil R\pi/\rho\rceil$ points. This grid is a ρ-cover of the cylinder. The grid is then mapped on the sphere such that the

one edge of the cylinder is mapped on the north pole, the other edge on the south pole, and the 'equator' of the cylinder is mapped to the equator of the sphere. As the distances between the grid points do not increase by this mapping, the projected points form a ρ-cover of the sphere S_R^d. By the induction assumption, the number of points in this ρ-cover satisfies

$$u_{d+1} \leq \tilde{u}_{d+1} = u_d \cdot \lceil R\pi/\rho \rceil \leq 2 \lceil R\pi/\rho \rceil^{d-1} \cdot \lceil R\pi/\rho \rceil = 2 \lceil R\pi/\rho \rceil^d .$$

We construct a lower bound on the covering number by dividing the surface area of the whole sphere by the area of the part of the surface covered by one single covering ball. The area of this part is smaller than the whole surface of the small ball. So we get a lower bound by dividing the surface area of S_R^{d-1} by the surface area of S_ρ^{d-1}. As the surface area of a sphere with radius R is R^{d-1} times the surface area of the unit sphere we get $(R/\rho)^{d-1}$ as lower bound for the covering number. ☺

Now we can explain how the sender will encode the direction of the separating hyperplane. Before getting the data, sender and receiver agree on a procedure on how to determine the centers of a covering of a unit sphere, given the number of balls to use for this covering, and on a way of enumerating these centers in some order. Furthermore, they agree on which kernel the sender will use for his SVM. Both sender and receiver get to see the training patterns, the sender also gets the training labels. Now the sender trains a (soft margin) SVM on the training data to obtain a hyperplane that separates the training patterns with some margin ρ (maybe with some errors). Then he computes the number u of balls of radius ρ one needs to cover the unit sphere in the feature space according to Proposition 3. He constructs such a covering according to the procedure he and the receiver agreed on. The centers of the covering balls form his set of codebook vectors. The sender enumerates the set of codebook vectors in the predefined way from 1 to u. Then he chooses a codebook vector which lies inside the rotation region of the normal vector (this is always possible by construction). We denote its index $i_u \in \{1, ..., u\}$. Now he transmits the total number u of codebook vectors and the index i_u of the one he chose. The receiver now constructs the same set of codebook vectors according to the common procedure, enumerates them in the same predefined way as the sender and picks vector i_u. This is the normal vector of the hyperplane he was looking for, and he can now use the corresponding hyperplane to classify the training patterns. In the codes below, we refer to the pair (u, i_u) as *position of the codebook vector*.

When we count how many bits the sender needs to transmit the two numbers u and i_u we have to keep in mind that to decode, the receiver has to know which parts of the binary string belong to u and i_u, respectively. The number u of codebook vectors is given as in Proposition 3, but as it depends on the margin it cannot be bounded independent from the training data. So the sender cannot use a fixed number of bits to encode u. Instead we use a trick described in Cover and Thomas (1991, p. 149): To build a code for the number u, we take the binary representation of u and duplicate every bit. To mark the end of the code we use the string 01.

As an example, $u = 27$ with the binary representation 11011 will be coded as 111100111101. So the receiver knows that the code of u is finished when he comes upon a pair of nonequal bits. We now apply this trick recursively: to code u, we first have to code the length $\log u$ of its binary code and then send the actual bits of the code of u. But instead of coding $\log u$ with the duplication code explained above, we can also first transmit the length $\log \log u$ of the code of $\log u$ and then transmit the code of $\log u$, and so on. At some point we stop this recursive procedure and code the last remaining number with the duplication code described above. This procedure of coding u needs $\log^*(u) := \log u + \log \log u + ...$ bits, where the sum continues until the last positive term (cf. p. 150 in Cover and Thomas, 1991). Having transmitted u, we can send i_u with $\log u$ bits, because i_u is a number between 1 and u and the sender now already knows u. All in all, the sender needs $\log^* u + \log u$ bits to transmit the position of the codebook vector.

The second part of the code deals with transmitting which of the training patterns are misclassified by the hyperplane corresponding to the chosen codebook vector. We have to be careful how we define the misclassified training points in case of a soft margin SVM. For a soft margin SVM it is allowed that some training points lie inside the margin. For those training points which end up inside the rotation region of the hyperplane, our rotation argument breaks down. It cannot be guaranteed that when the receiver uses the hyperplane corresponding to the codebook vector, the points inside the rotation region are classified in the same way as the sender classified them with the original hyperplane. Thus the sender has to transmit which points are inside the rotation region, and he also has to send their labels. All other points can be treated more easily. The sender has to transmit which of the points outside the rotation region were misclassified. The receiver knows that those points will also be misclassified by his hyperplane, and he can flip their labels to get them right. Below we will refer to the part consisting of the information on the points inside the rotation region and on the misclassified points outside the rotation region as *misclassification information*.

The number r of points inside the rotation region is a number between 0 and n, thus we can transmit its binary representation using $\log n$ bits. After transmitting r, the receiver knows how many training points lie inside the region, but not which of them. There are $\binom{n}{r}$ possibilities which of the training points are the points inside the rotation region. Before transmission, sender and receiver agree on an ordering on those possibilities. Now the sender can transmit the index i_r of the one that is the true one. As this is a number between 1 and $\binom{n}{r}$, and as the receiver at this point already knows r, this can be encoded with $\log \binom{n}{r}$ bits. Next the sender can use r bits to send the labels of the r points inside the rotation region. Finally the sender has to transmit the number l of misclassified training points outside the rotation region. It is a number between 1 and $n - r$, thus we can use $\log(n - r)$ bits for this. To transmit which of the vectors are the misclassified ones, we send the index i_l with $\log \binom{n-r}{l}$ bits. All together, we need $\log n + \log \binom{n}{r} + r + \log(n - r) + \log \binom{n-r}{l}$ bits to transmit the misclassification information. For simplicity, we bound this quantity from above by $(r + l + 2) \log n + r$.

Now we can formulate our first code:

Code 1 *Sender and receiver agree on the training patterns, a fixed kernel, and on a procedure for choosing the positions of t balls to cover the unit sphere in a d-dimensional Euclidean space. Now the sender trains an SVM with the fixed kernel on the training patterns, determines the size of the margin ρ and the number u of balls he needs to cover the sphere up to the necessary accuracy according to Proposition 3. Furthermore, he determines which of the training patterns lie inside the rotation region and which patterns outside this region are misclassified by the SVM solution. Now he transmits*

- *the position of the codebook vector (*$\log^* u + \log u$ *bits)*
- *the misclassification information (* $(r + l + 2) \log n + r$ *bits)*

To decode this information, the receiver constructs a covering of the sphere in the feature space with $t := u$ balls according to the common procedure, determines the used codebook vector i, and constructs a hyperplane using this vector as normal vector. He classifies all training patterns according to this hyperplane, labels the points inside the rotation region as transmitted by the sender, and flips the labels of the misclassified training points outside the rotation region.

As defined in Equation (1), the compression coefficient of a code is given by the number of bits it needs to transmit all its information, divided by the number n of labels that were transmitted. Hence, according to our computations above, the compression coefficient of Code 1 is given by

$$C_1 = \frac{1}{n} \left(\log^* u + \log u + (r + l + 2) \log n + r \right)$$

with $u = 2 \lceil R\pi/\rho \rceil^{d-1}$ according to Proposition 3.

2.2 Using the shape of the data in the feature space

Now we want to refine this code in several aspects. In the construction above we worked with the smallest sphere which contains all the training points. But in practice, the shape of the data in the feature space is typically ellipsoid rather than spherical. This means that large parts of the sphere we used above are actually empty, and thus the code we constructed is not very efficient. Now we want to take into account the shape of the data in the feature space to construct a shorter code. In this setting it will turn out to be convenient to choose the hypotheses space to have the same shape as the data space. The reason for this is the following: When using the rotation argument from above, we observe that in the ellipsoid situation the maximal rotation angle of the hyperplane induced by some fixed margin ρ depends on the actual direction of the hyperplane (cf. Figure 2). This means that the sets of points on a *spherical hypotheses space* which classify the training data in the same way as the hyperplane also have different sizes, depending on the

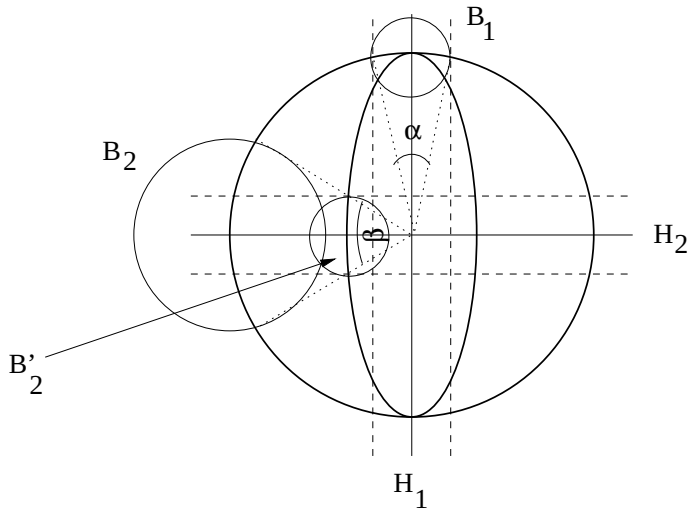


Figure 2: In this figure, the ellipse represents the data domain and the large circle represents a spherical hypotheses space. Consider two hyperplanes H_1 and H_2 with equal margin. The balls B_1 resp. B_2 indicate the sets of hypotheses that yield the same classification result as H_1 resp. H_2. The sizes of those balls depend on the direction of the hyperplane with respect to the ellipse. In the example shown, B_1 is smaller than B_2, hence H_1 must be coded with higher accuracy than H_2. Note that if we use the ellipse itself as hypotheses space, we have to consider the balls B_1 and B_2' which are centered on the surface of the ellipse and have equal size.

direction of the hyperplane. To construct an optimal set of codebook vectors on the spherical hypotheses space we thus had to cover the sphere with balls of different sizes. Instead of this spherical hypotheses space now consider a hypotheses space which has the same ellipsoid form as the data space. In this case, the sets of vectors which correspond to directions inside the rotation regions can be represented by balls of equal sizes, centered on the surface of the ellipsoid (cf. Figure 2).

Now we want to determine the shape of the ellipsoid containing the data points in the feature space. The lengths of the principal axes of this ellipse can be described in terms of the eigenvalues of the kernel matrix:

Proposition 4 (Shape of the data ellipse) *For given training patterns $(x_i)_{i=1,...,m}$ and kernel k, let $\lambda_1, ..., \lambda_d$ be the eigenvalues of the kernel matrix $K = (k(x_i, x_j))_{i,j=1,...,m}$. Then all training patterns are contained in an ellipse with principal axes of lengths $\sqrt{\lambda_1}, ..., \sqrt{\lambda_d}$ in the feature space.*

Proof. The trick of the proof is to interpret the eigenvectors of the kernel matrix, who originally live in $\mathbb{R}^d$, as vectors in the feature space. Let $H_m := \text{span}\{\delta_{x_i} | i = 1, ..., m\}$ the subspace of the feature space spanned by the training examples. It is endowed with the scalar product $\langle \delta_{x_i}, \delta_{x_j} \rangle_K = k(x_i, x_j)$. Let $(e_i)_{i=1,...,m}$ the canonical basis of $\mathbb{R}^m$ and $\langle \cdot, \cdot \rangle_{\mathbb{R}^m}$ the Euclidean scalar product. Define the mapping $T : \mathbb{R}^m \rightarrow$

$H_m,\ e_i \mapsto \delta_{x_i}$. For $u = \sum_{i=1}^m u_i e_i,\ v = \sum_{j=1}^m v_j e_j \in \mathbb{R}^m$ we have

$$\langle Tu, Tv\rangle_K = \langle \sum_{i=1}^m u_i \delta_{x_i}, \sum_{j=1}^m v_j \delta_{x_j}\rangle_K = \sum_{i,j=1}^m u_i v_j \langle \delta_{x_i}, \delta_{x_j}\rangle_K = \tag{2}$$

$$= \sum_{i,j=1}^m u_i v_j k(x_i, x_j) = u' K v\ . \tag{3}$$

Let $v_1, ..., v_d \in \mathbb{R}^m$ be the normalized eigenvectors of the matrix K corresponding to the eigenvalues $\lambda_1, ..., \lambda_d$, that is $Kv_i = \lambda_i v_i$ and $\langle v_i, v_j\rangle_{\mathbb{R}^m} = \delta_{ij}$. From Equation (3) we can deduce

$$\langle Tv_i, Tv_j\rangle_K = v_i' K v_j = v_i' \lambda_j v_j = \lambda_j \langle v_i, v_j\rangle_{\mathbb{R}^m} = \lambda_i \delta_{ij},$$

in particular $\|Tv_i\|_K = \sqrt{\lambda_i}$. Furthermore we have

$$\langle \delta_{x_i}, Tv_j\rangle_K = \langle \delta_{x_i}, \sum_{l=1}^m (v_j)_l \delta_{x_l}\rangle_K = \sum_{l=1}^m (v_j)_l k(x_i, x_l) = (Kv_j)_i = (\lambda_j v_j)_i = \lambda_j (v_j)_i.$$

Altogether we can now see that in the feature space, all data points δ_{x_i} lie in the ellipse whose principal axes have direction Tv_j and length $\sqrt{\lambda_j}$ because the ellipse equation is satisfied:

$$\sum_{j=1}^d \left(\frac{\langle \delta_{x_i}, \frac{Tv_j}{\|Tv_j\|_K}\rangle_K}{\sqrt{\lambda_j}} \right)^2 = \sum_j \left((v_j)_i\right)^2 \leq 1.$$

Here the last equality follows from the fact that $\sum_j \left((v_j)_i\right)^2$ is the Euclidean norm of a row vector of the orthonormal matrix containing the eigenvectors $(v_1, ..., v_d)$. ☺

Now that we know the shape of the ellipse, we have to find out how many balls of radius ρ we need to cover its surface. As surfaces of ellipses in high dimensions are complicated to deal with, we simplify our calculation. Instead of covering the surface of the ellipse, we will cover the ellipse completely. This means that we use one extra dimension (volume instead of area), but in high dimensional spaces this does not make much difference, especially if some of the axes are very small. Computing a rough bound on the covering numbers of an ellipse is easy:

Proposition 5 (Covering numbers of ellipses) *The number u of balls of radius ρ which are required to cover a d-dimensional ellipse with principal axes $c_1, ..., c_d$ satisfies*

$$\prod_{i=1}^d \frac{c_i}{\rho} \leq u \leq \prod_{i=1}^d \left\lceil \frac{2c_i}{\rho} \right\rceil .$$

Proof. The smallest parallelepiped containing the ellipse has side lengths $2c_1, ..., 2c_d$ and can be covered with a grid of $\prod_{i=1}^{d} \lceil 2c_i/\rho \rceil$ balls of radius ρ. This gives an upper bound on the covering number.

To obtain a lower bound we divide the volume of the ellipse by the volume of one single ball. Let v_d be the volume of a d-dimensional unit ball. Then the volume of a d-dimensional ball of radius ρ is $\rho^d v_d$ and the volume of an ellipse with axes $c_1, ..., c_d$ is given by $v_d \prod_{i=1}^{d} c_d$. So we need at least $\prod_{i=1}^{d}(c_i/\rho)$ balls. ☺

Now we can formulate the refined code:

Code 2 *This code works analogously to Code 1, the only difference is that sender and receiver work with a covering of the data ellipse instead of a covering of the enclosing sphere.*

The compression coefficient of Code 2 is

$$C_2 = \frac{1}{n} \left(\log^* u + \log u + (r + l + 2) \log n + r \right),$$

with $u = \prod_{i=1}^{d} \lceil 2\sqrt{\lambda_i}/\rho \rceil$ according to Propositions 4 and 5.

It is interesting to notice that the main complexity term in the compression coefficient is the logarithm of the number of balls needed to cover the region of interest of the hypotheses space. So the shape of the bounds we obtain is very similar to classical bounds based on covering numbers in statistical learning theory. This is not so surprising since we explicitly approximate our hypotheses space by covers, but there is a somewhat deeper connection. Indeed, when we construct our code, we consider all normal vectors in a certain region as equivalent with respect to the labeling they give of the data. This means that we define a metric on the set of possible normal vectors which is related to the induced Hamming distance on the data (that is the natural distance in the "coordinate projections" of our function class on the data). Hence, when we adapt the size of the balls to the direction in hypotheses space (in Figure 2), we actually say that Hamming distance 1 on the data translates into a certain radius. Hence, we are led to build covers in this induced distance which is exactly the distance which is used in classical covering number bounds for classification. So the compression approach gives another motivation, of information theoretic flavor, for considering that the right measure of the capacity of a function class is the metric entropy of its coordinate projections.

2.3 Support vectors help reducing the coding dimension

Both compression coefficients we derived so far implicitly depend on the dimension d of the feature space in which we code the hyperplane. Above we always used $d = n$ as the solution of an SVM always lives in the subspace spanned by the training examples. But as the solution even lies in the subspace spanned by the support vectors, an easy dimension reduction can be achieved by working in this subspace.

The procedure then works as follows: the sender trains the SVM and determines the support vectors and the margin ρ. The ellipse that we have to consider now is the ellipse determined by the kernel matrix K_{SV} restricted to the linear span of the support vectors (cf. notations at the end of Section 1). The reason for this is that we are only interested in what happens if we slightly change the direction of the normal vector *within* the subspace spanned by the support vectors.

To let the receiver know the subspace he is working in, the sender has to transmit which of the training patterns are support vectors. This part of the code will be called *support vector information*. As the number s of support vectors is between 0 and n, the sender first codes s with $\log n$ bits and then the index i_s of the actual support vectors among the $\binom{n}{s}$ possibilities with $\log \binom{n}{s}$ bits. So the support vector information can be coded with $\log n + \log \binom{n}{s} \leq (s+1) \log n$ bits. After submitting the information about the support vectors, the code proceeds analogously to Code 2.

Code 3 *Sender and receiver agree on the training patterns, the kernel, and the procedure of covering an ellipsoid. After training an SVM, the sender transmits*

- *the support vector information (*$(s+1)\log n$ *bits),*
- *the position of the codebook vector (*$\log^* u + \log u$ *bits),*
- *the misclassification information (*$(r+l+2)\log n + r$ *bits).*

To decode, the receiver constructs the hypotheses space consisting of the data ellipse projected on the subspace spanned by the support vectors. He covers this ellipse with u balls and chooses the vector representing the normal vector of the hyperplane. Then he labels the training patterns by first projecting them into the subspace and then classifying them according to the hyperplane. Finally, he deals with the misclassified training points as in the codes before.

The compression coefficient of Code 3 is given as

$$C_3 = \frac{1}{n}\left(\log^* u + \log u + (r+l+s+3)\log n + r\right),$$

with $u \leq \prod_{i=1}^{s} \lceil 2\sqrt{\gamma_i}/\rho \rceil$ according to Propositions 4 and 5. Here γ_i denote the eigenvalues of the restricted kernel matrix K_{SV}.

2.4 Reducing the coding dimension with kernel PCA

A further dimension reduction can be obtained with the following idea: It has been empirically observed that on most data sets the axes of the data ellipse decrease fast for large dimensions. Once the axis in one direction is very small, we want to discard this dimension by projecting in a lower dimensional subspace using kernel principal component decomposition (cf. Schölkopf and Smola, 2002). A projection P will be allowed if the image $P(\omega)$ of the normal vector ω is still within the rotation region

induced by the margin ρ. In this case we construct codebook vectors for $P(\omega)$ in the lower dimensional subspace. We have to make sure that the vector representing $P(\omega)$ is still contained in the original rotation region.

In more detail, this approach works as follows: First we train the SVM and get the normal vector ω and margin ρ. For convenience, we now normalize ω to length R by $\omega_0 := \frac{\omega}{||\omega||}R$. After normalizing, we know that a vector v still lies inside the rotation region if $\|\omega_0 - v\| \leq \rho$. Now we perform a kernel PCA of the training data in the feature space. For $d_P \in \{1, ..., n\}$ let P be the projection on the subspace spanned by the first d_P eigenvectors. To determine whether we are allowed to perform projection d_P we have to check whether $\|P(\omega_0) - \omega_0\| \leq \rho$. If not, the hyperplane corresponding to $P(\omega_0)$ is not within the rotation region any more, and we are not allowed to make this projection. Otherwise, we are still within the rotation region after projecting ω_0, so we can discard the last $n - d_P$ dimensions. In this case, we call P a *valid* projection. We then can encode the projected normal vector $P(\omega_0)$ in an d_P-dimensional subspace. As $P(\omega_0)$ is not in the center of the rotation region any more, we have to code its direction more precisely now. We have to ensure that the codebook vector v for $P(\omega_0)$ still is in the original rotation region, that is $\|v - \omega_0\| \leq \rho$. Define

$$c_P := \frac{1}{\rho}\|\omega_0 - P(\omega_0)\|$$

(note that for a valid projection, $c_P \in [0, 1]$), and choose the radius r of the covering balls as $r = \rho\sqrt{1 - c_P^2}$. Then we have

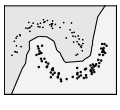
$$\|\omega_0 - v\|^2 \leq \|\omega_0 - P(\omega_0)\|^2 + \|P(\omega_0) - v\|^2 \leq c_P^2\rho^2 + (1 - c_P^2)\rho^2 = \rho^2 \ .$$

Thus the codebook vector v is still within the allowed distance of ω_0.

All in all our procedure now works as follows: The sender trains an SVM. From now on he works in the subspace spanned by the support vectors only. In this subspace, he performs the PCA and determines the smallest d_P such that the projection on the subspace of d_P dimensions is a valid projection.The principal axes of the ellipse in the subspace are now given by the first d_P eigenvalues of the restricted kernel matrix K_{SV}, and we have to construct a covering of this ellipse with covering radius $r = \rho\sqrt{1 - c_P^2}$. Then the code proceeds as before. The number d_P will be called the *projection information* and can be encoded with $\log m$ bits.

Code 4 *Sender and receiver agree on the training patterns, the kernel, the procedure of covering ellipsoids, and on how to perform kernel PCA. The sender trains the SVM and chooses a valid projection on some subspace. He transmits*

- *the support vector information (*$(s + 1) \log n$ *bits),*
- *the projection information (*$\log n$ *bits),*
- *the position of the codebook vector (*$\log^* u + \log u$ *bits),*
- *the misclassification information (*$(r + l + 2) \log n + r$ *bits).*



To decode, the receiver constructs the hypotheses space consisting of the ellipse in the subspace spanned by the support vectors. Then he performs a PCA in this subspace and projects the hypotheses space on the subspace spanned by the first d_P principal components. He covers the remaining ellipse with u balls and continues as in the codes before.

The compression coefficient of this code is given by

$$C_4 = \frac{1}{n}\left(\log^* u + \log u + (r + l + s + 4)\log n + r\right),$$

with $u \leq \prod_{i=1}^{d_P} \lceil 2\sqrt{\gamma_i}/(\rho\sqrt{1-c_P^2}\,)\rceil$, c_P as described above, and γ_i the eigenvalues of the restricted kernel matrix K_{SV}.

2.5 A pure support vector code

So far we always considered codes which use the direction of the hyperplane as hypothesis. A totally different approach is to reduce the data by transmitting support vectors and their labels. Then the receiver can train his own SVM on the support vectors and will get the same result as the sender. Note that in this case, we not only have to transmit which of the vectors are support vectors as in the other codes, but also the labels of the support vectors. On the other hand we have the advantage that we do not have to treat the points inside the rotation region separately as they are support vectors anyway. The misclassification information only consists in the misclassified points which are not support vectors. This simple code works as follows:

Code 5 *Sender and receiver agree on training patterns and a kernel. The sender sends*

- *the support vector information ($(s+1)\log n$ bits),*
- *the labels of the support vectors (s bits),*
- *the information on the misclassified points outside the rotation region ($(l+1)\log n$ bits).*

To decode this information, the receiver trains an SVM with the support vectors as training set. Then he computes the classification result of this SVM for the remaining training patterns and flips the labels of the misclassified non-support vector training points.

This code has a compression coefficient of

$$C_5 = \frac{1}{n}\left((s + l + 2)\log n + s\right).$$

3 Experiments

To test the utility of the derived compression coefficients for applications, we ran model selection experiments on different artificial and real world data sets.

3.1 The setup

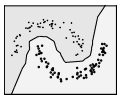
We used all data sets in the benchmark data set repository compiled and explained in detail in Rätsch et al. (2001). The data sets are available at http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm. Most of the data sets in this repository are preprocessed versions of data sets originating from the UCI, Delve, or STATLOG repositories. In particular, all data sets are normalized. The data sets are called banana (5300 points), breast cancer (277 points), diabetis (768 points), flare-solar (1066 points), german (1000 points), heart (2700 points), image (2310 points), ringnorm (7400 points), splice (3175 points), thyroid (215 points), titanic (2201 points), twonorm (7400 points), waveform (5100 points). To be consistent with earlier versions of this manuscript we also used the data sets abalone (4177 points), Wisconsin breast cancer (683 points) from the UCI repository, and the US postal handwritten digits data set (9298 points). In all experiments, we first permuted the whole data set and divided it into as many disjoint training subsets of sample size $m = 100$ or 500 as possible. We centered each training subset in the feature space as described in Section 14.2. of Schölkopf and Smola (2002) and then used it to train soft margin SVMs with Gaussian kernels. The test error was computed on the training subset's complement.

For different choices of the soft margin parameter $C \in [10^0, ..., 10^5]$ and the kernel width $\sigma \in [10^{-2}, ..., 10^3]$ we computed the compression coefficients and chose the parameters where the compression coefficients were minimal. Note that as we centered the data in the feature space, the radius R can be approximately computed as the maximum distance of the centered training points to the origin. We compared the test errors corresponding to the chosen parameters to the ones obtained by model selection criteria from different generalization bounds. Recall that we denote the empirical and true risk of a classifier by $\mathcal{R}_{\text{emp}}$ and $\mathcal{R}$, respectively. Note that the definition of the risks varies slightly for the different bounds, for example with respect to the used loss function. We refer to the cited papers for details. The bounds we consider are the following:

- The radius-margin bound of Vapnik (1998). We here cite the version stated in Bartlett and Shawe-Taylor (1999): with probability at least $1 - \delta$,

$$\mathcal{R} \leq \mathcal{R}_{\text{emp}} + \sqrt{\frac{c}{n}(\frac{R^2}{\rho^2}\log^2 n - \log\delta)}.$$

  The quantity we compute in the experiments is $\mathcal{R}_{\text{emp}} + \sqrt{(R^2 \log n)/(\rho^2 n)}$.



- The rescaled radius-margin bound of Chapelle and Vapnik (2000). It uses the shape of the training data in the feature space to refine the classical radius margin bound. In this case, the quantity R^2/ρ^2 in the above bound is replaced by

$$\sum_{k=1}^{n} \lambda_k^2 \max_{i=1,\ldots,n} A_{ik}^2 (\sum_{j=1,\ldots,n} A_{jk} y_j \alpha_j)^2,$$

  where A is the matrix of the normalized eigenvectors of the kernel matrix K, λ_k are the eigenvalues of the kernel matrix, and α the coefficients of the SVM solution.

- The trace bound of Bartlett and Mendelson (2001) which contains the eigenvalues of the kernel matrix: with probability at least $1 - \delta$,

$$\mathcal{R} \leq \mathcal{R}_{\text{emp}} + R_n + \sqrt{\frac{8 \ln(2/\delta)}{n}},$$

  where the Rademacher complexity R_n is given as follows. Let $\lambda_1, \lambda_2, \ldots$ the eigenvalues of the integral operator $T_k f(x) = \int k(x,y) f(y) dP(y)$. Then the Rademacher complexity is bounded by $R_n \leq \sqrt{\sum_{j=1}^{\infty} \lambda_j / n}$. As we cannot compute $\sum_j \lambda_j$ in practice, in the experiments we replace this quantity by $\sum_{i=1}^{n} \sigma_i$ where σ_i are the eigenvalues of the kernel matrix. This is a reasonable procedure as the trace of the kernel matrix is concentrated and converges to the trace of the integral operator (Shawe-Taylor et al., 2002). The quantity we thus compute in the experiments is $\mathcal{R}_{\text{emp}} + \sqrt{\sum_{i=1}^{n} \sigma_i / n}$.

- The compression scheme bound of Floyd and Warmuth (1995) using the sparsity of the SVM solution: with probability at least $1 - \delta$,

$$\mathcal{R} \leq \frac{1}{n-s} \left(\ln \binom{n}{s} + \ln \frac{n^2}{\delta} \right),$$

  where s is the number of support vectors. In the experiments we computed the quantity $\ln \binom{n}{s} / (n-s)$. Note that if $s = n$ this bound is infinite. We omitted those cases from the plots.

- The sparse margin bound of Herbrich et al. (2000) which uses the size of the margin and the sparsity of the solution: with probability at least $1 - \delta$,

$$\mathcal{R} \leq \frac{1}{n-s} \left(\kappa \ln \frac{en}{\kappa} + \ln \frac{n^2}{\delta} \right),$$

  where $\kappa = \min(\lceil \frac{R^2}{\rho^2} + 1 \rceil, d+1)$. For our experiments we compute the quantity $\frac{1}{n-s} \left(\kappa \ln \frac{en}{\kappa} \right)$. In case $s = n$ this bound is infinite and we omit those cases from the plots.

- The span estimate of Vapnik and Chapelle (2000), cf. also Opper and Winther (2000). This bound is different from all the other bounds as it estimates the leave-one-out error of the classifier. To achieve this, it bounds for each support vector how much the solution would change if this particular support vector were removed from the training set.

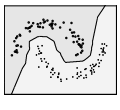
All experimental results shown below were obtained with training set size $n = 100$; the results for $n = 500$ are comparable. The interested reader can study those results, as well as many more plots which we cannot show here because they would use too much space, on the webpage http://www.kyb.tuebingen.mpg.de/bs/people/ule.

3.2 Results

The goal of our first experiment is to use the compression coefficients to select the kernel width σ. In Figure 3 we study which of the five compression coefficients achieves the best results on this task. The plots in this figure were obtained as follows. For each training set, each parameter C and each compression coefficient we chose the kernel width σ for which the compression coefficient was minimal. Then we evaluated the test error for the chosen parameters on the test set and computed the mean over all runs on the different training sets. Plotted are the means of these test errors versus parameter C, as well as the means of the minimal true test errors.

We can observe that for nearly all data sets, the compression coefficients C_2, C_3, and C_4 yield better results than the simple coefficients C_1 and C_5. This can be explained by the fact that C_1 and C_5 only use one part of information (the size of the margin or the number of support vectors, respectively), while the other coefficients combine several parts of information (margin, shape of data, number of support vectors). When we compare coefficients C_3 and C_4 we observe that they have nearly identical values. This indicates that the gain we obtain by projecting into a smaller subspace using kernel PCA is outweighed by the additional information we have to transmit about this projection. As C_3 is simpler to compute, we thus prefer C_3 to C_4. From now on we want to evaluate the results of the most promising compression coefficients C_2 and C_3.

In Figure 4 we compare compression coefficients C_2 and C_3 to all the other model selection criteria. The plots were obtained in the same way as the ones above. We see that for most data sets, the performance of C_2 is rather good, and in most cases it is better than C_3. It performs nearly always comparable or better than the standard bounds, in particular it is nearly always better than the widely-used radius margin bound. Among all bounds, C_2 and the span bound achieve the best results. Comparing those two bounds shows that no one is superior to the other one: C_2 is better than the span bound on five data sets (abalone, banana, diabetis, german, usps), the span bound is better than C_2 on five data sets (image, ringnorm, splice, thyroid, waveform), and they achieve similar results on six data sets (breast-cancer, flare-solar, heart, titanic, twonorm, wisconsin).



The goal of the next experiment was not only to select the kernel width σ, but to find the best values for the kernel width σ and the soft margin parameter C simultaneously. Its results can be seen in Table 1, which was produced as follows. For each training set we chose the parameters σ and C for which the respective bounds were minimal, and evaluated the test error for the chosen parameters. Then we computed the means over the different training runs. The first column contains the mean value of the minimal test error. The other columns contain the offset by which the test errors selected by the different bounds are worse than the optimal test error. On most data sets, the radius margin bound, the rescaled radius margin bound, and the sparse margin bound perform rather poorly. Often their results are worse than those of the other bounds by one order of magnitude. Among the other bounds, C_2 and the span bound are the two superior bounds. Between those two bounds, there is a tendency towards the span bound in this experiment: C_2 beats the span bound on 6 data sets, the span bound beats C_2 on 10 data sets. Thus C_2 does not perform as good as the span bound, but it gets close.

To explain the good performance of the compression coefficients we now want to analyze their properties in more detail. As the compression coefficients are a sum of several terms it is a natural question which of the terms has the largest influence on the actual value of the sum. To answer this question we look at Figure 5, where we plotted the different parts of C_3: the term $(s+1)\log n$ corresponding to the support vector information, the term $\log^* u + \log u$ corresponding to the position of the codebook vector, the term $(r+1)\log n + r$ corresponding to the points inside the rotation region, and the term $(l+1)\log n$ corresponding to the information on the misclassified points outside the rotation region. For each fixed soft margin parameter C we chose the kernel width σ where the value of the compression coefficient C_3 is minimal. For this value of σ, we plotted the means of the different terms. Here we only show the plots for compression coefficient C_3 (as C_3 has more different terms than C_2). The plots on the other data sets, as well as the plots for C_2, are very similar to the ones we show. We find that all terms (except the term "misclass. inside" which is negligible) are of the same order of magnitude, with no term consistently dominating the other ones. This behavior is attractive as it shows that all different parts of information have substantial influence on the value of the compression coefficient.

Finally we want to study the shapes of the curves of the different bounds. As we use the values of the bounds to predict the qualitative behavior of the test error it is important that the shapes of the bounds' curves are similar to the shape of the test error curve. In Figure 6 we plot those shapes for the compression coefficients C_2, C_3, and the other bounds versus the kernel width σ. We show the plots for every second value of C (to cover the whole range of values of C we used). First of all we can observe that the value of the compression coefficient is often larger than 1. This is also the case for several of the other bounds, and is due to the fact that most bounds only yield nontrivial results for very large sample sizes. This need not be a problem for applications as we use the bounds to predict the qualitative behavior of the test error, not the quantitative one. Secondly, the compression scheme and the

data set	test error	C_2	C_3	span	rm	rrm	trace	sm	cs
abalone	0.224	0.017	0.036	0.029	0.137	0.134	0.012	0.084	0.072
banana	0.124	0.021	0.024	0.020	0.347	0.278	0.141	0.202	0.047
breast-cancer	0.251	0.062	0.065	0.209	0.034	0.034	0.028	0.034	0.042
diabetis	0.247	0.012	0.049	0.025	0.103	0.103	0.059	0.103	0.080
flare-solar	0.339	0.026	0.027	0.020	0.120	0.110	0.030	0.101	0.101
german	0.263	0.037	0.042	0.026	0.037	0.037	0.060	0.037	0.044
heart	0.156	0.015	0.015	0.021	0.303	0.168	0.071	0.159	0.053
image	0.105	0.074	0.028	0.023	0.275	0.235	0.031	0.183	0.028
ringnorm	0.021	0.054	0.052	0.007	0.405	0.017	0.075	0.178	0.068
splice	0.198	0.039	0.053	0.016	0.249	0.035	0.054	0.084	0.053
thyroid	0.026	0.030	0.017	0.026	0.178	0.178	0.022	0.178	0.017
titanic	0.220	0.010	0.010	0.012	0.103	0.103	0.008	0.065	0.041
twonorm	0.027	0.016	0.026	0.007	0.029	0.006	0.027	0.009	0.026
usps	0.103	0.075	0.071	0.020	0.278	0.141	0.072	0.190	0.072
waveform	0.118	0.047	0.040	0.019	0.179	0.120	0.045	0.179	0.042
wisconsin	0.028	0.007	0.011	0.015	0.006	0.013	0.027	0.005	0.008

Table IV.1: Model selection results for selecting the kernel width σ and the soft margin parameter C simultaneously. For each training set and each bound we chose the parameters σ and C for which the bound was minimal, and evaluated the test error for the chosen parameters. Shown are the mean values of the test errors over the different training sets. The first column contains the value of the test error. The other columns contain the offset by which the test errors achieved by the different bounds are worse than the optimal test error.

sparse margin bound suffer from the fact that they only attain finite values when the number of support vectors is smaller than the number of training vectors. Among all bounds, only C_2, C_3 and the span bound seem to be able to predict the shape of the test error curve.

The main conclusions we can draw from all experimental results is that in all three tasks (predicting the shape of the test error curve, choosing parameter σ, choosing parameters σ and C) the span bound and compression coefficient C_2 have the best performance among all bounds, where none of the two bounds is clearly superior to the other one. The latter fact is also remarkable for the following reason. All considered bounds apart from the span bound use the capacity of the model class to bound the expected risk of the classifier. The span bound on the other hand is a clever way of computing an upper bound on the leave-one-out error of the classifier, which is known to be an almost unbiased estimator of the true risk. Thus the methods by which those bounds are derived are intrinsically different. Our results now show that the bounds derived by studying the size of the model class can achieve results in practice that are comparable to using the state of the art span bound.

4 Conclusions

We derived five compression coefficients for SVMs which combine information on the geometry of the training data in the feature space with information about geometry and sparsity of the classifier. In our model selection experiments it turned out that the compression coefficients can be readily used to predict the parameters where the test error is small. Our favorite compression coefficient is C_2 because it is easy to compute and yields good results in the experiments. The results it achieves are comparable to those of the state of the art span bound. The theoretical justification for using compression coefficients are the generalization bounds we cited in Section 2. They were proved in an abstract coding theoretic setting. We now derived methods to apply these bounds in practical applications. This shows that the connection between information theory and learning can be exploited in every-day machine learning applications.

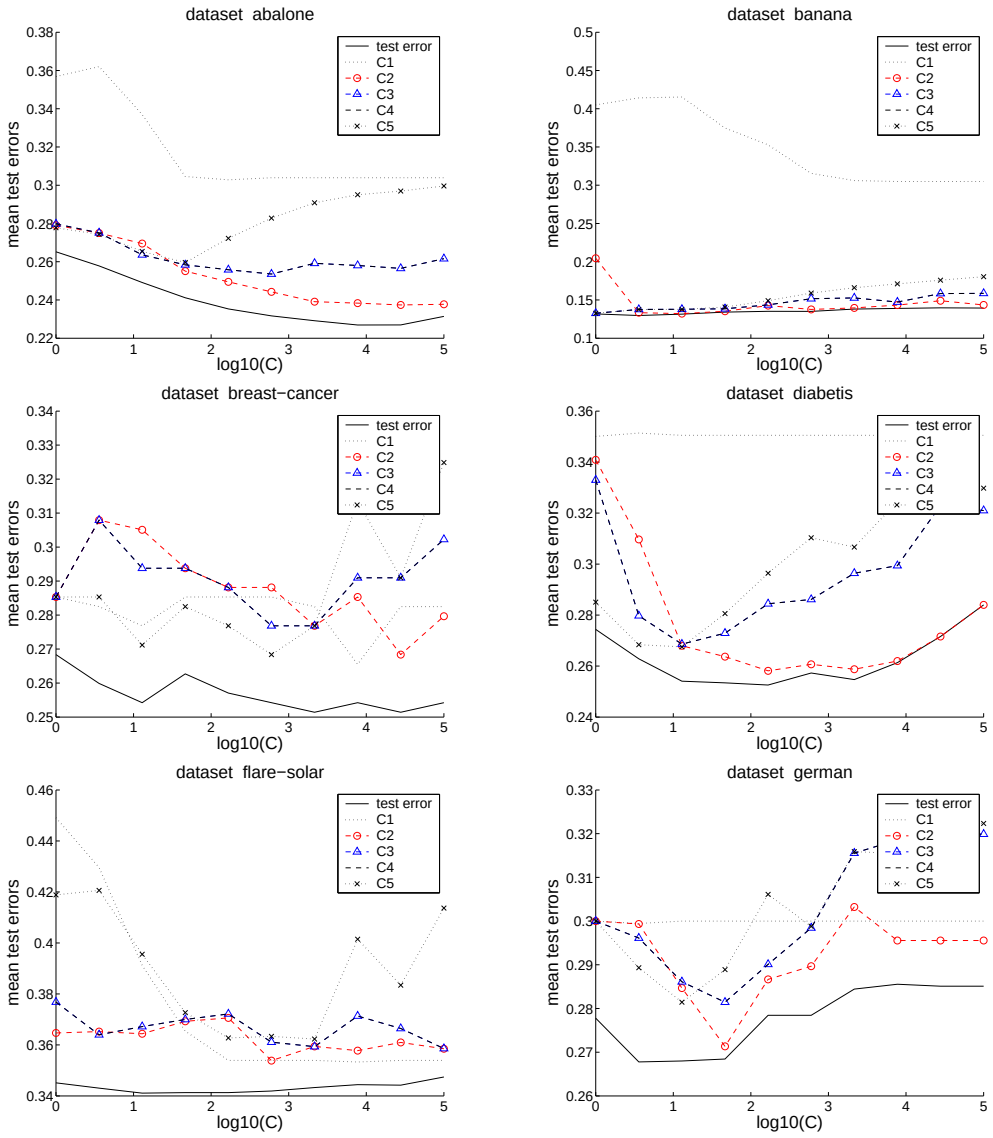


Figure 3: Comparison among the compression coefficients. For each training set, each soft margin parameter C and each compression coefficient we chose the kernel width σ for which the compression coefficient was minimal and evaluated the test error for the chosen parameters. Plotted are the mean values of the test errors over the different training sets, as well as the means of the true minimal test errors (this figure is continued on the next page).

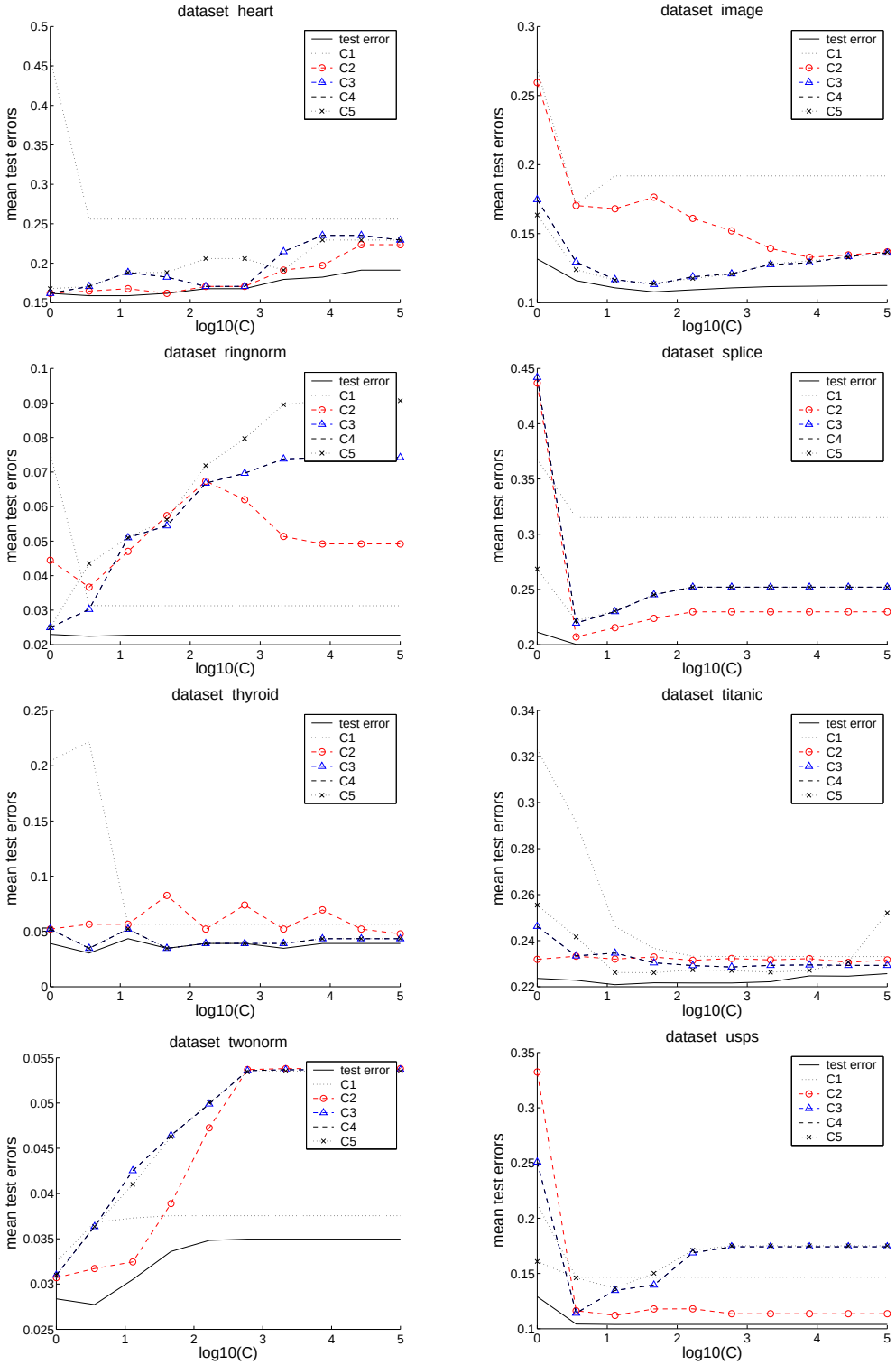


Figure 3, continued

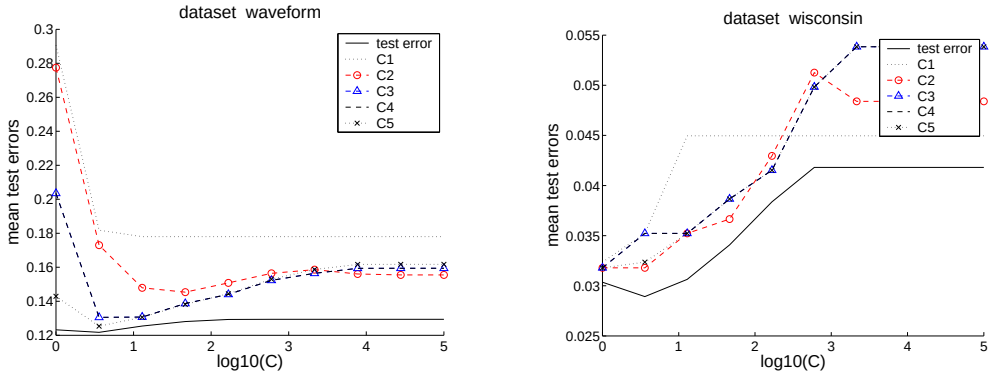


Figure 3, continued

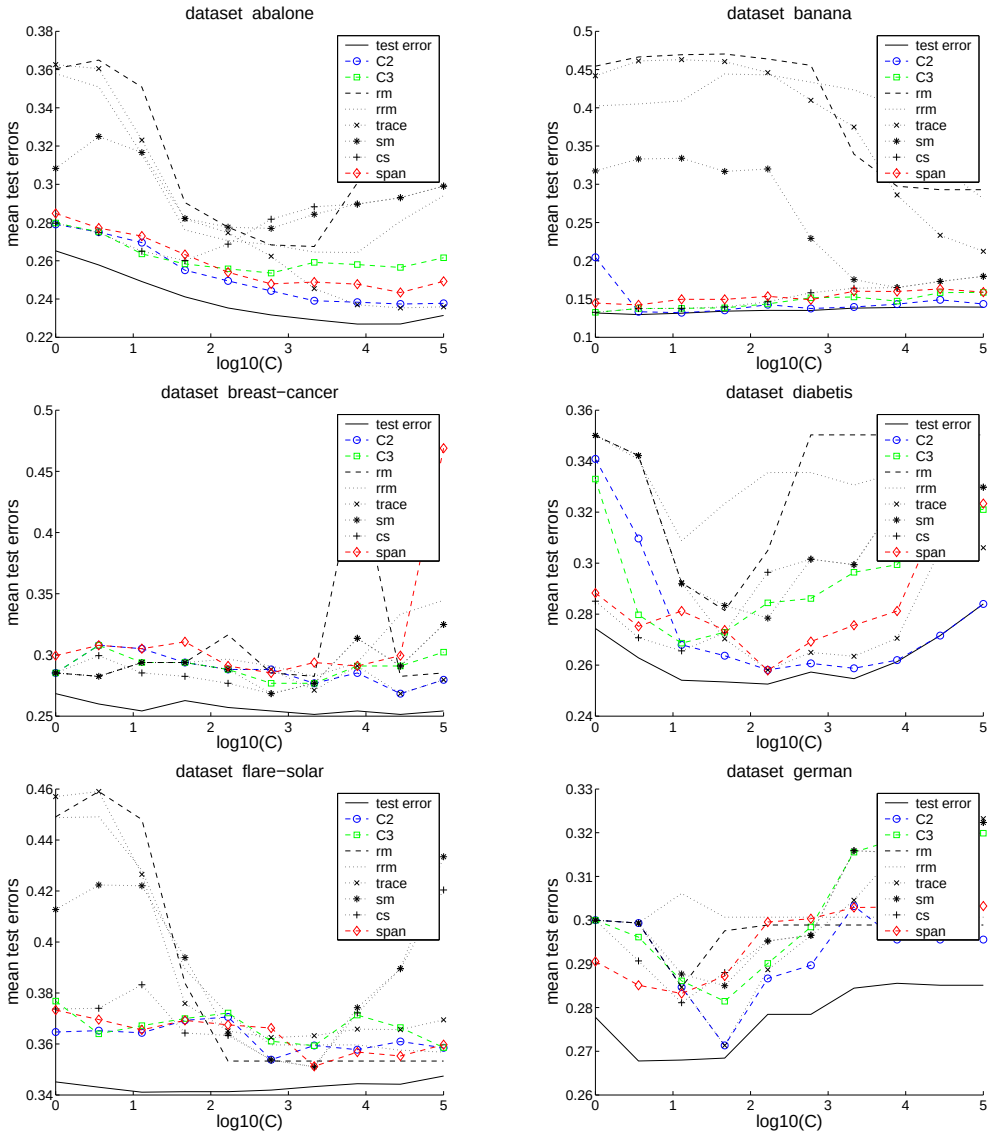


Figure 4: Comparison between C_2, C_3, and the other bounds. For each training set, each soft margin parameter C and each bound we chose the kernel width σ for which the bound was minimal and evalutated the test error for the chosen parameters. Plotted are the mean values of the test errors over the different training sets. In the legend we use the abbreviations rm = radius margin bound, rrm = rescaled radius margin bound, sm = sparse margin bound, and cs = compression scheme bound (continued on the next page).

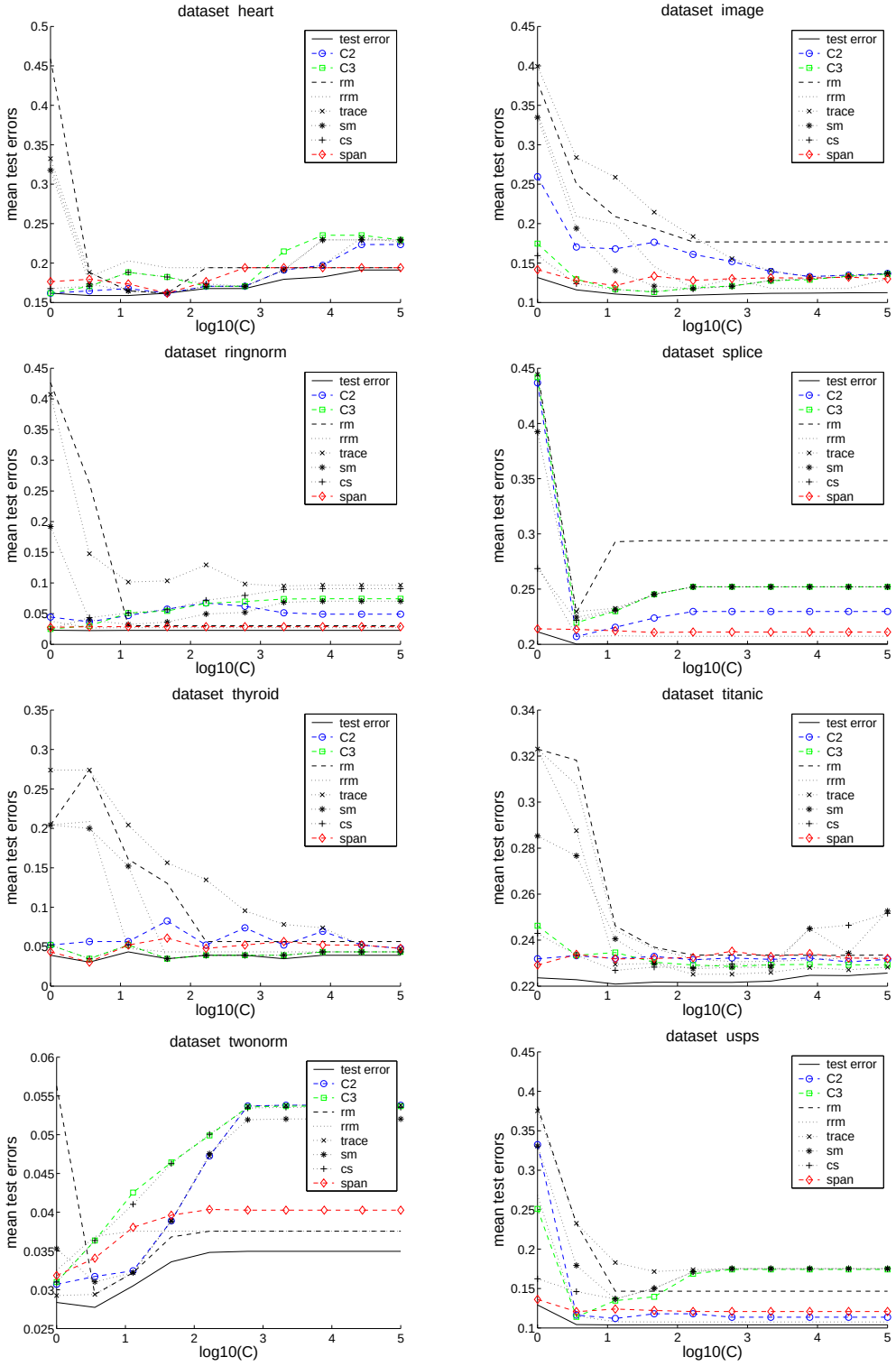


Figure 4, continued

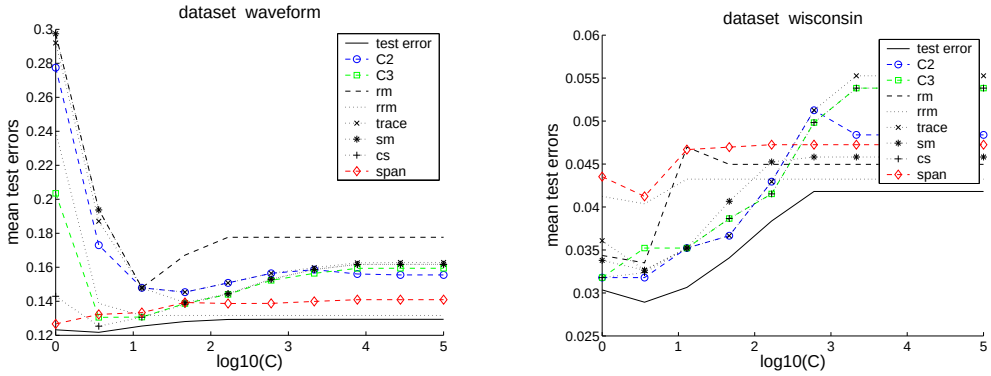


Figure 4, continued

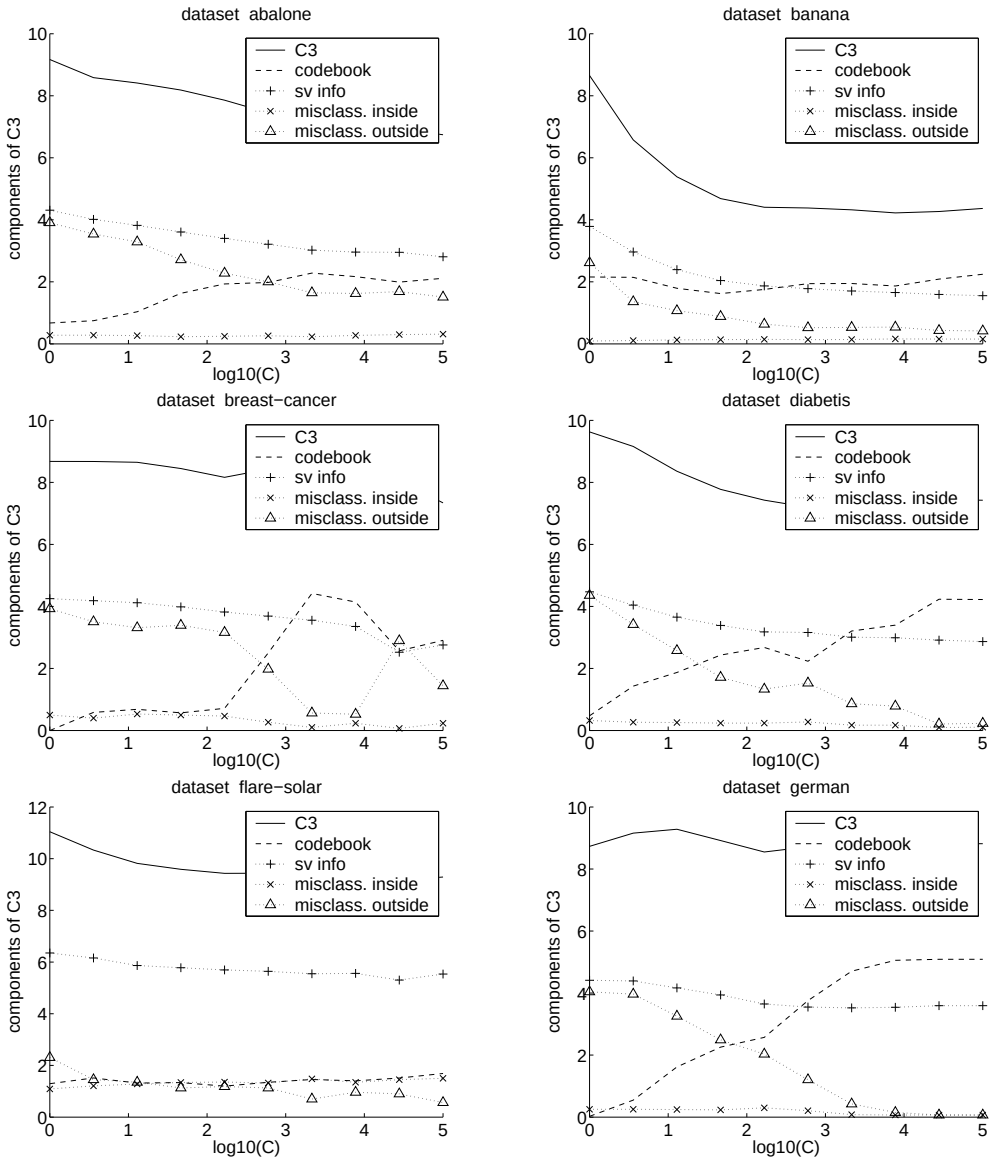


Figure 5. Here we study the relationship between the different components of C_3. We plot the lengths of the codes for the support vector information ("sv info"), the position of the codebook vector ("codebook"), the information on the points inside the rotation region ("misclass. inside"), and the information about the misclassified points outside the rotation region ("misclass. outside").

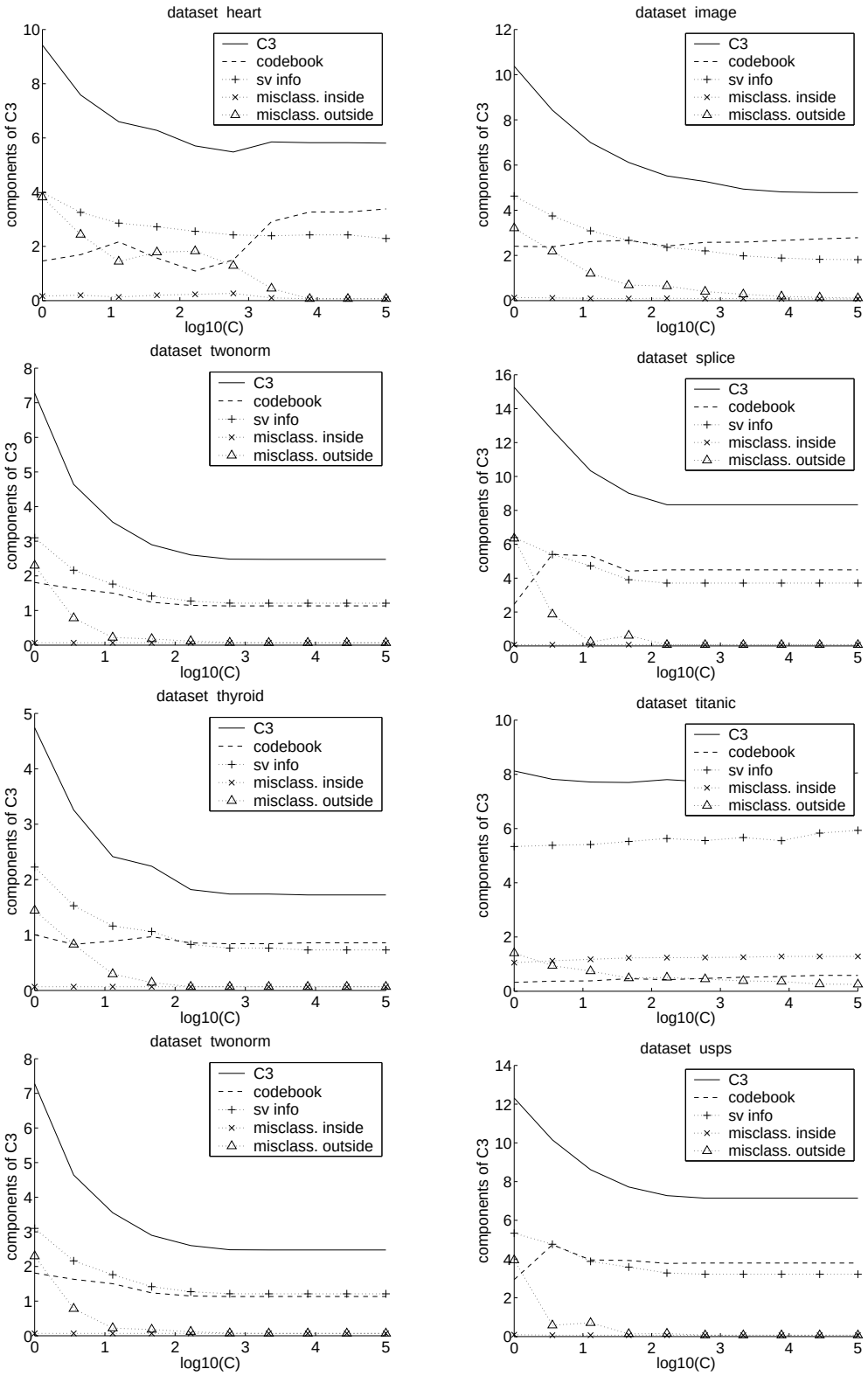


Figure 5, continued

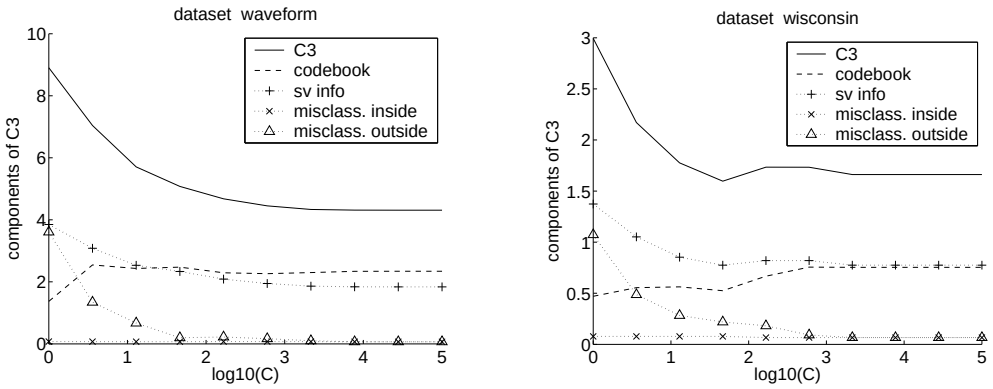


Figure 5, continued

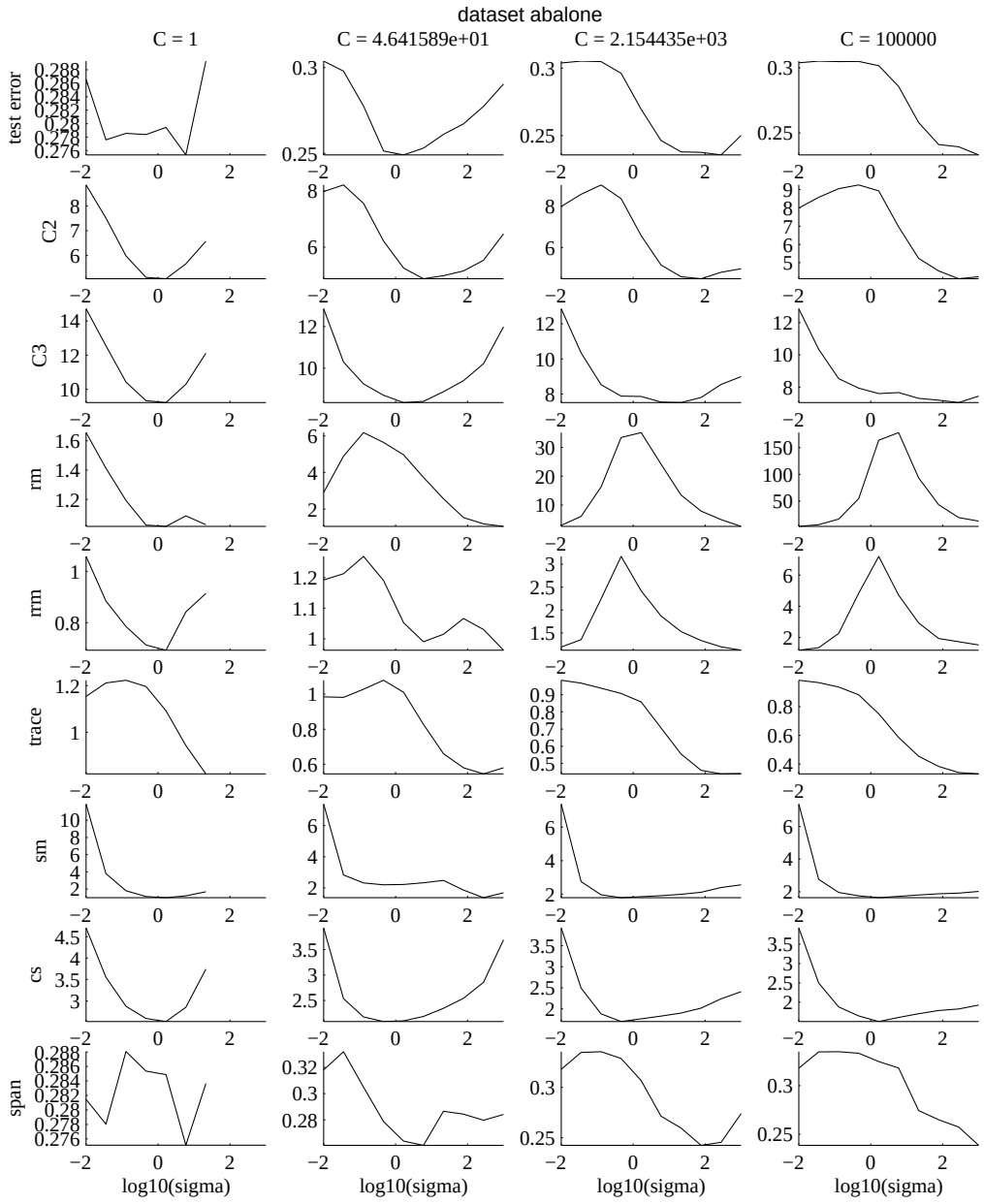


Figure 6: Shapes of curves. Plotted are the mean values of the bounds themselves over the different training runs versus the kernel width σ, for fixed values of the soft margin parameter C.

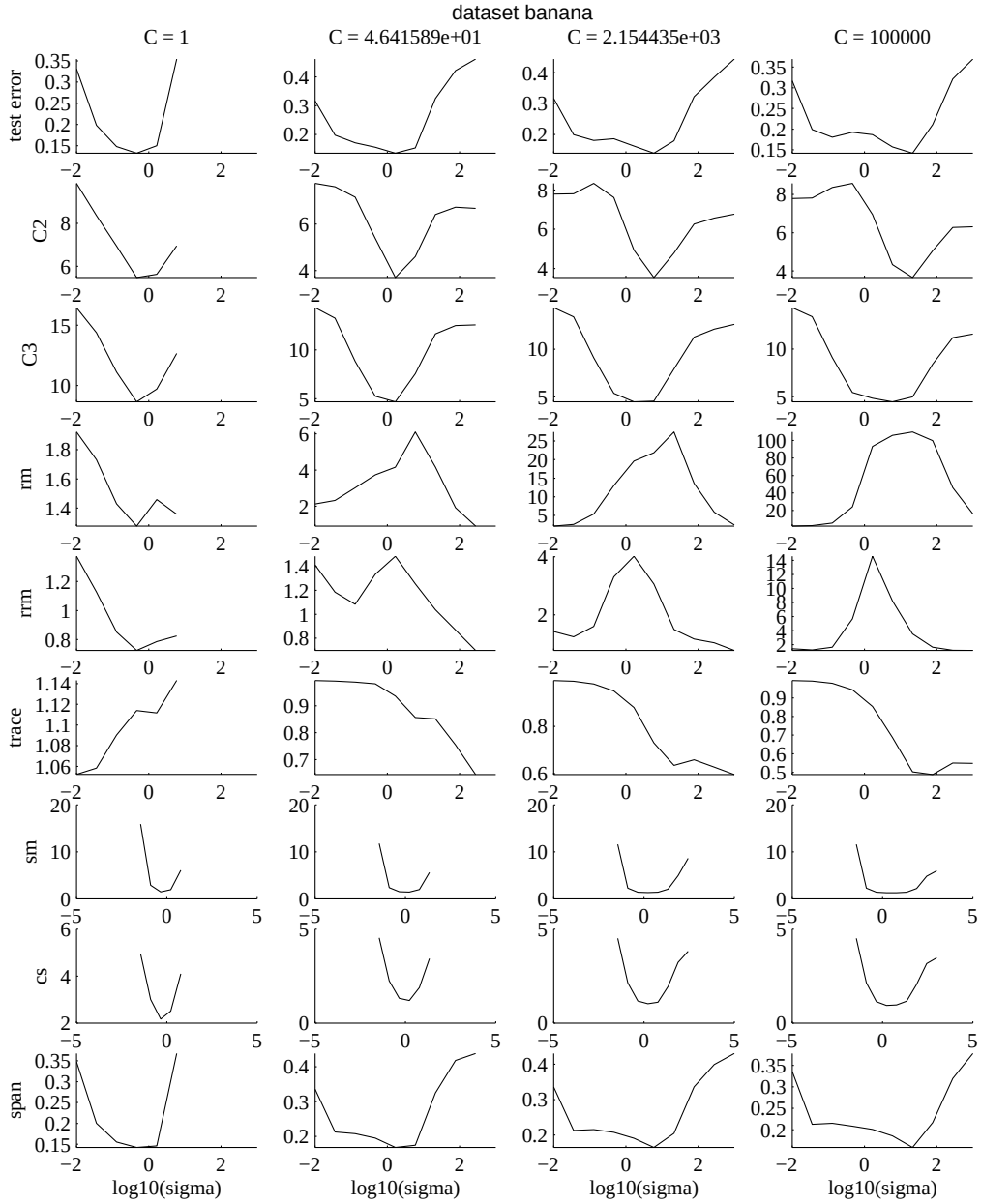


Figure 6, continued

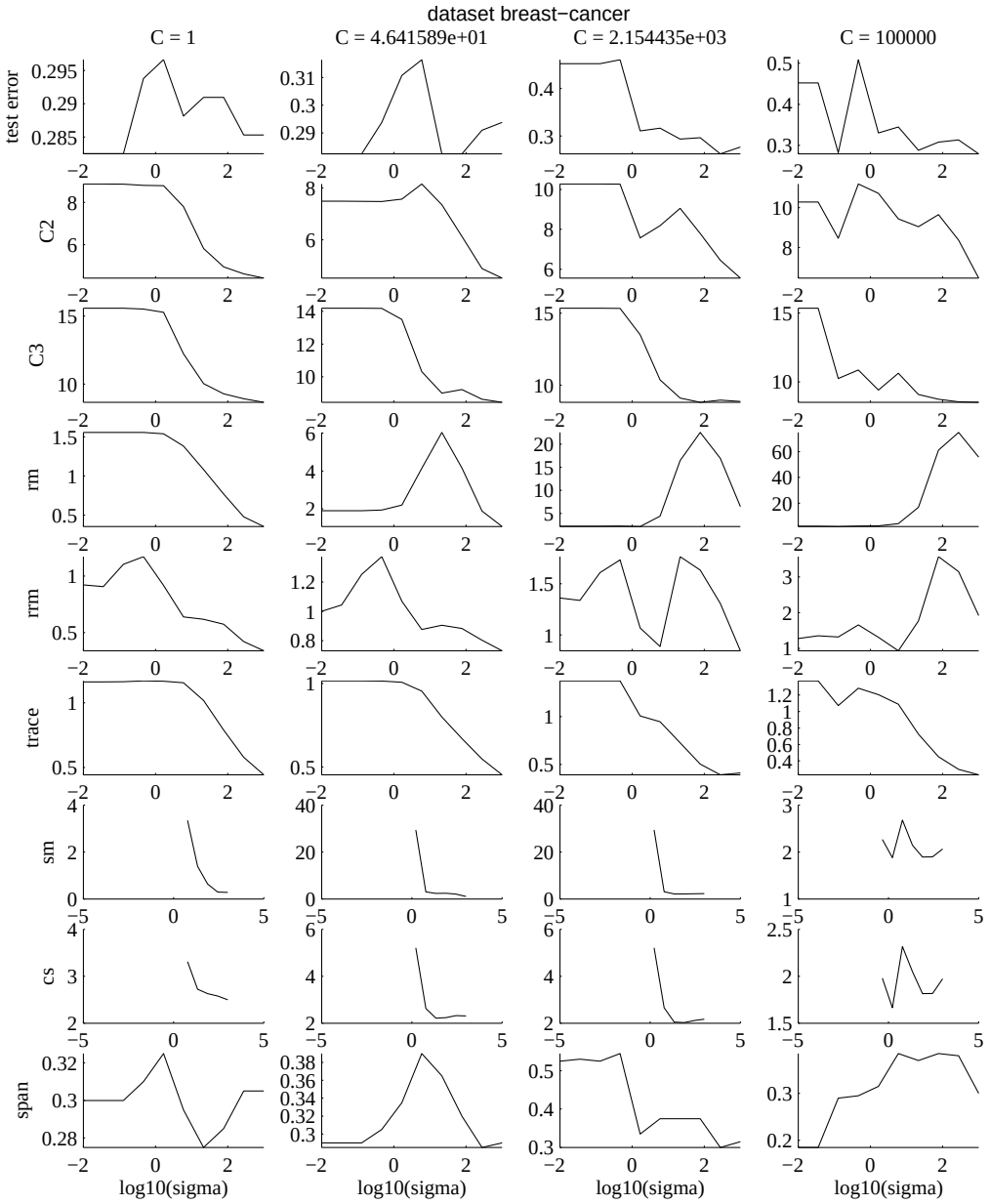


Figure 6, continued

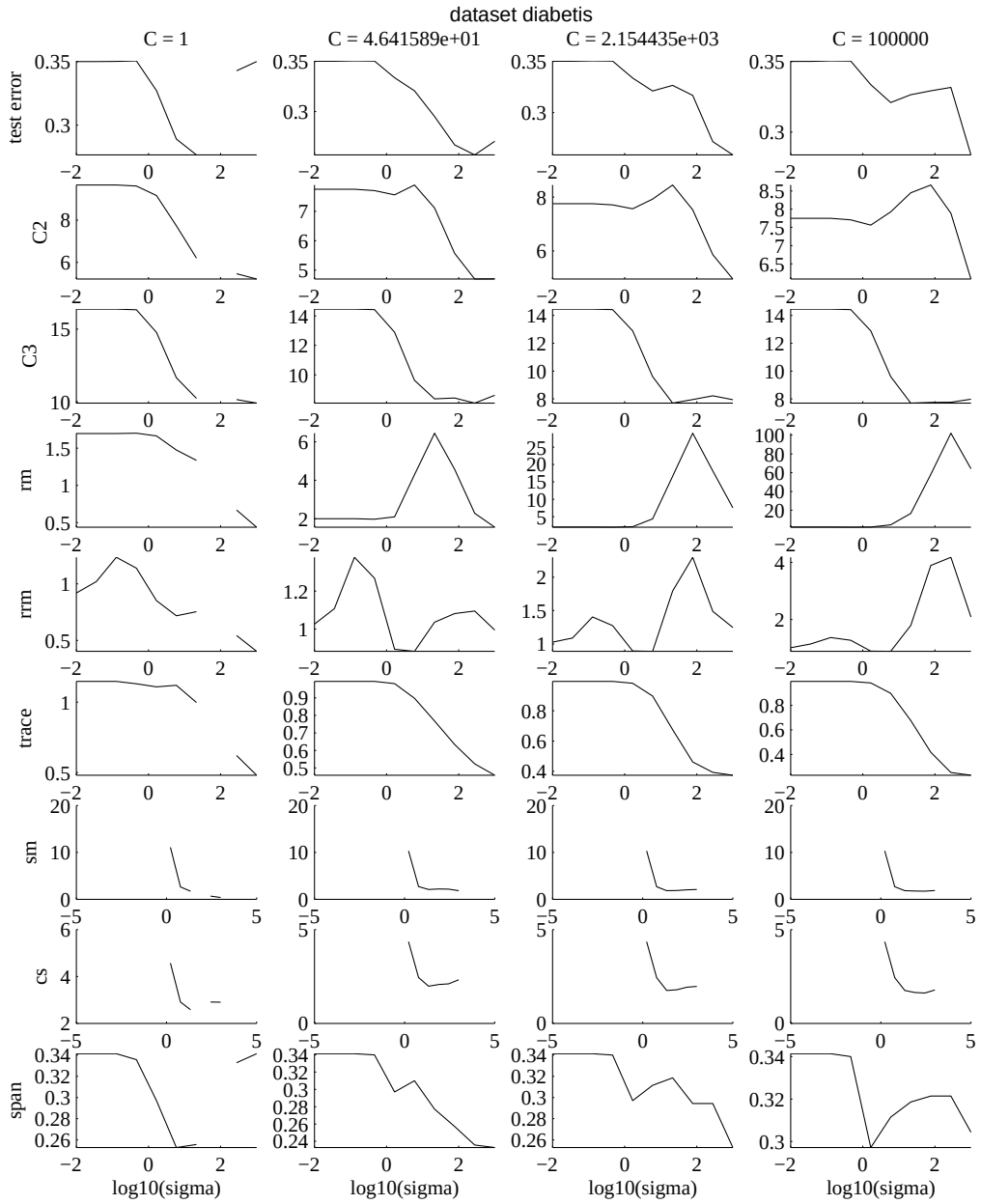


Figure 6, continued

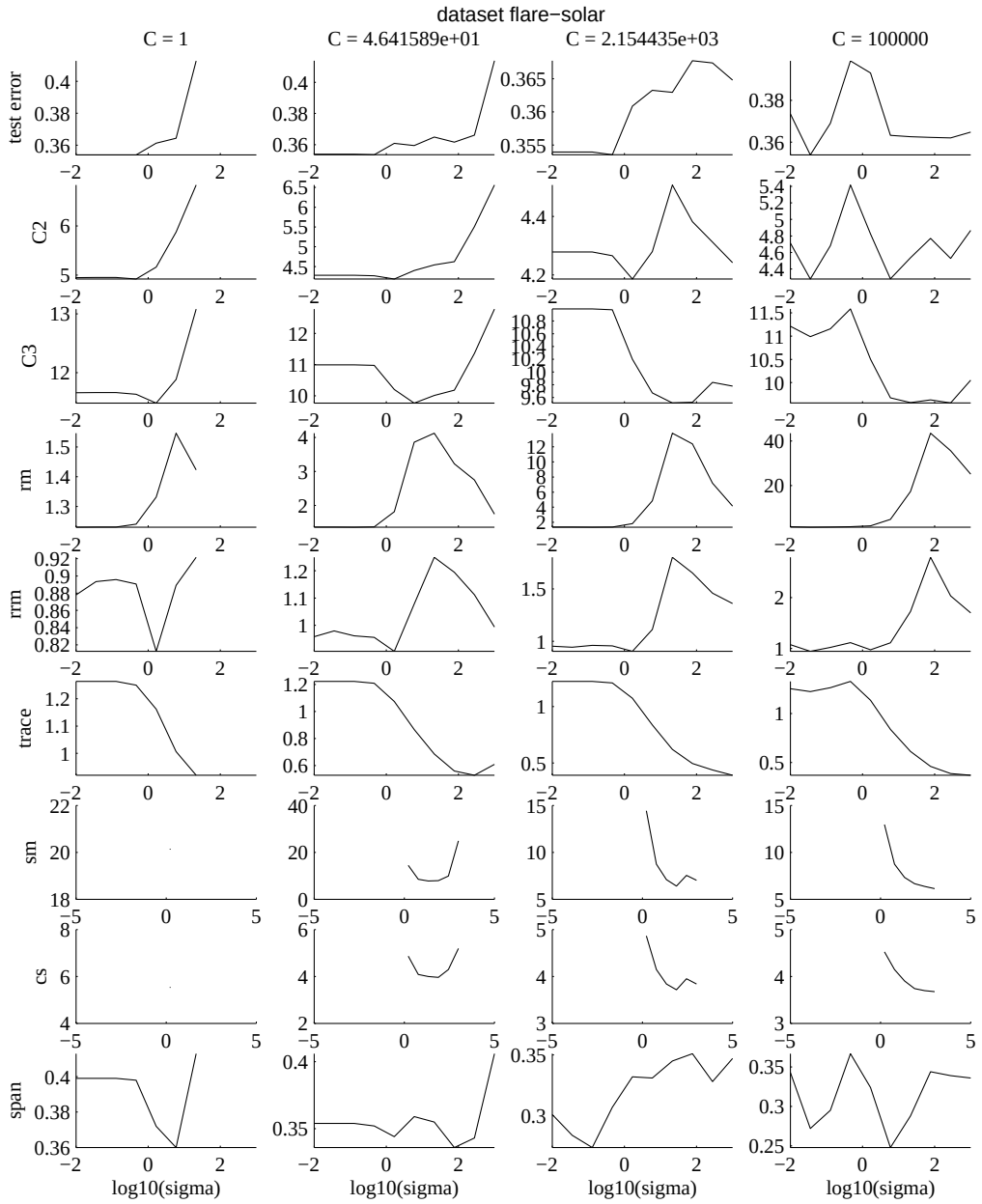


Figure 6, continued

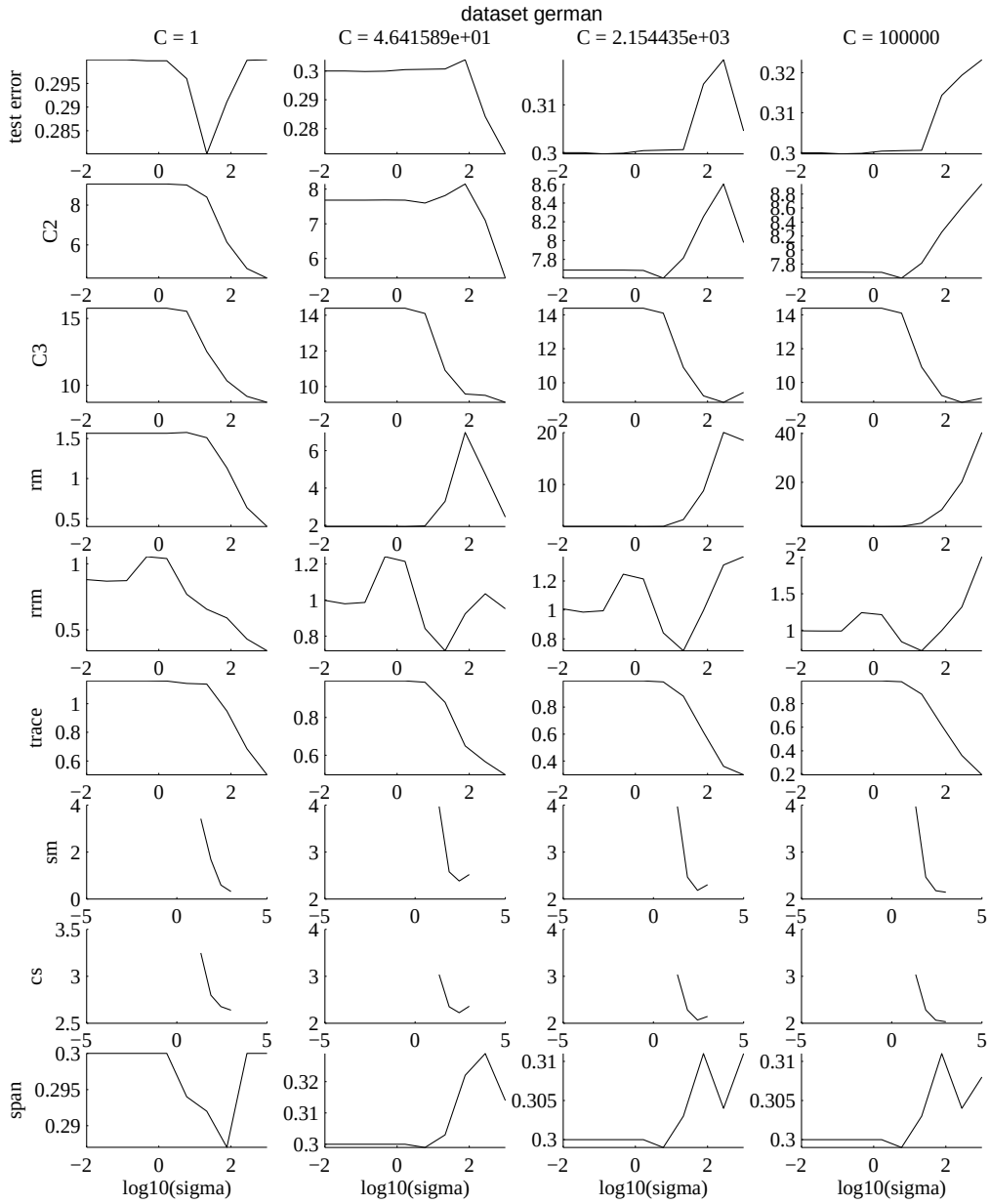


Figure 6, continued

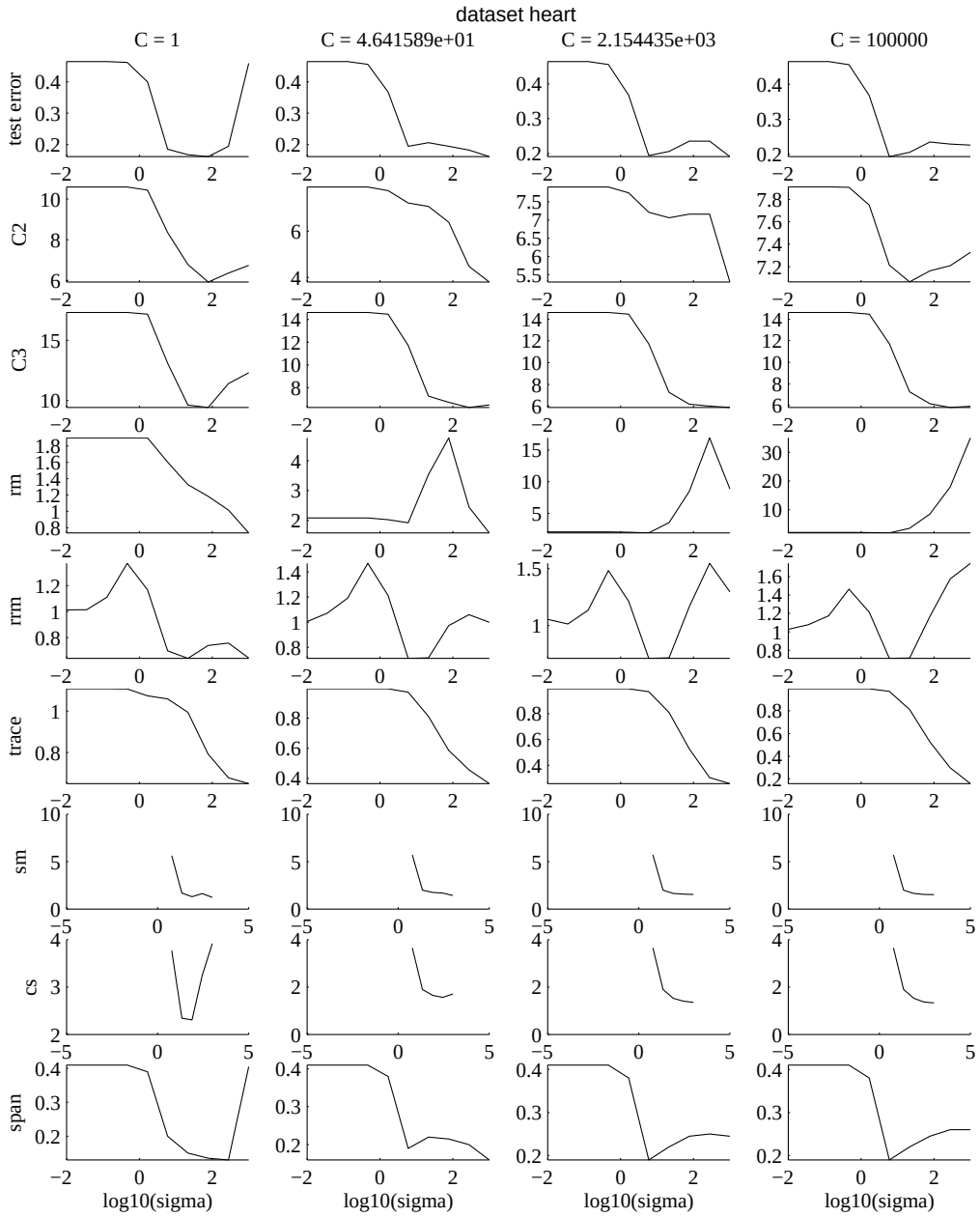


Figure 6, continued

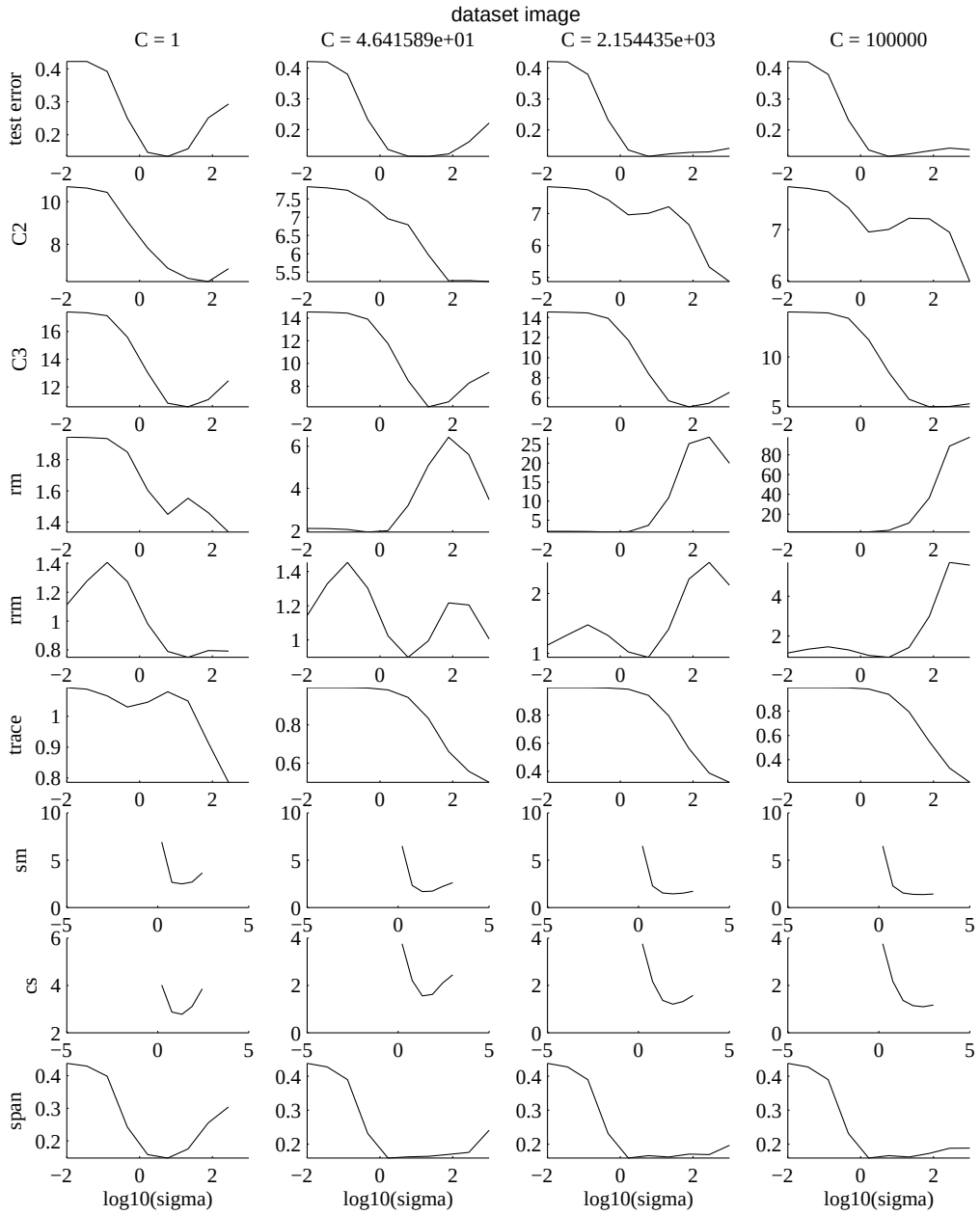


Figure 6, continued

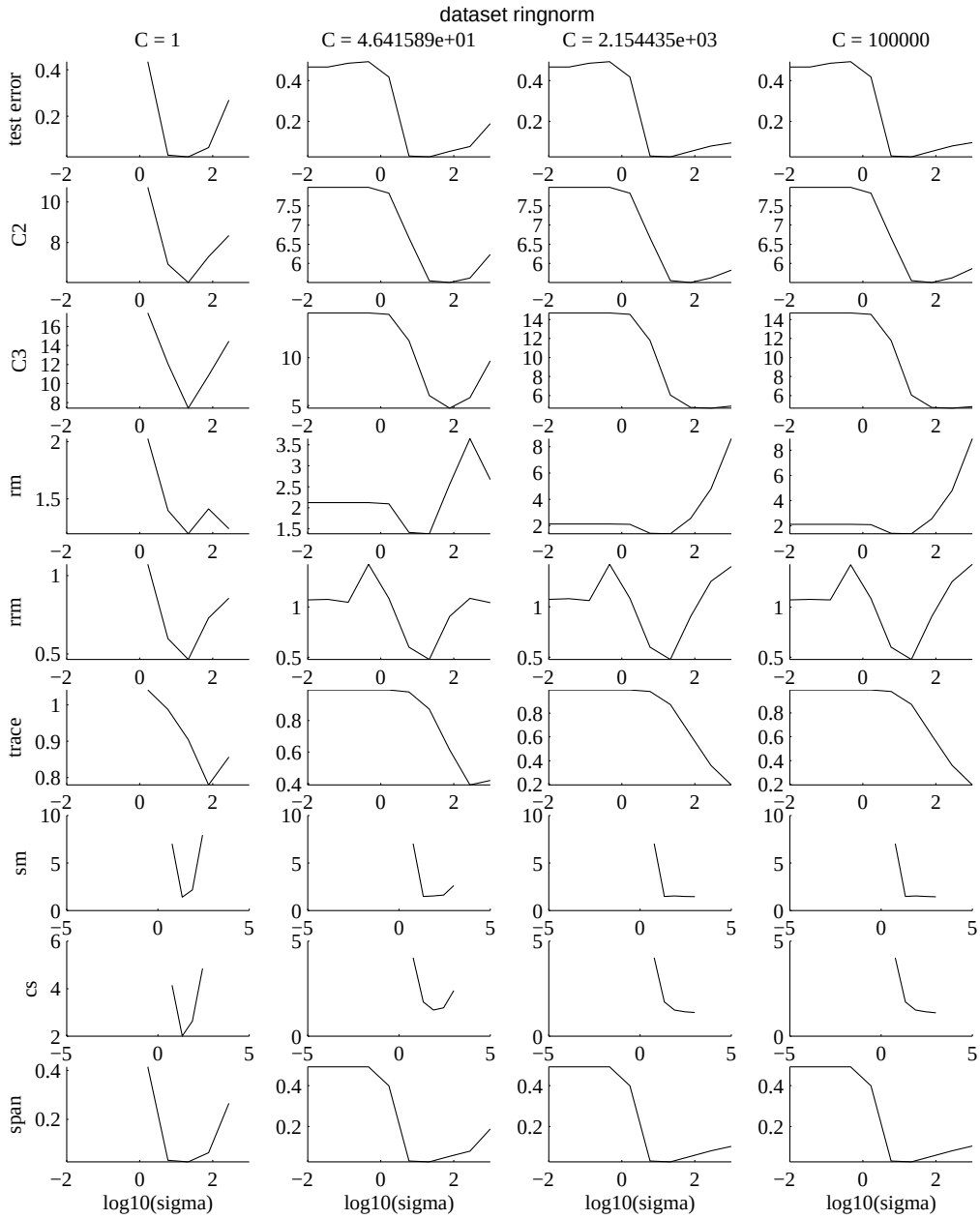


Figure 6, continued

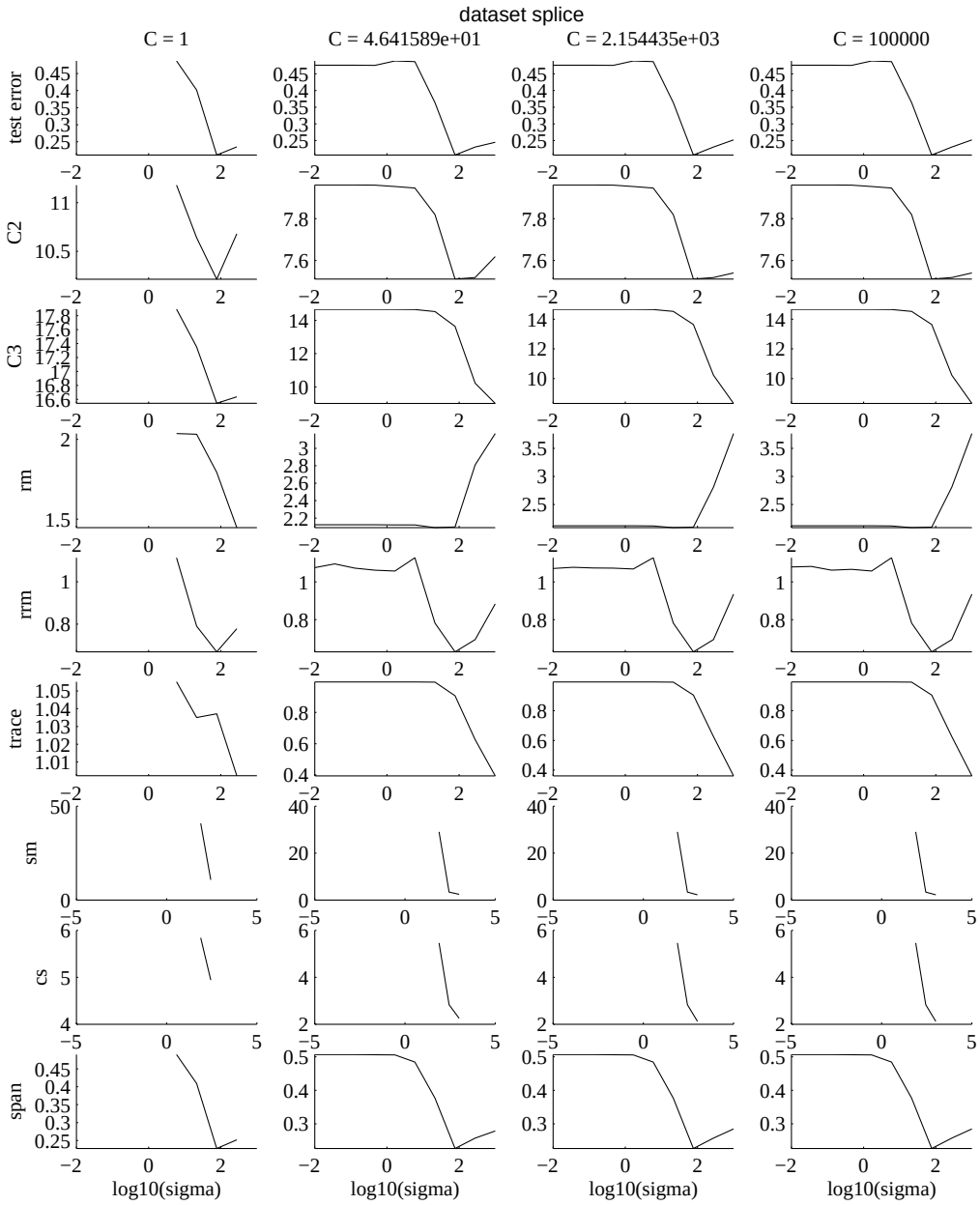


Figure 6, continued

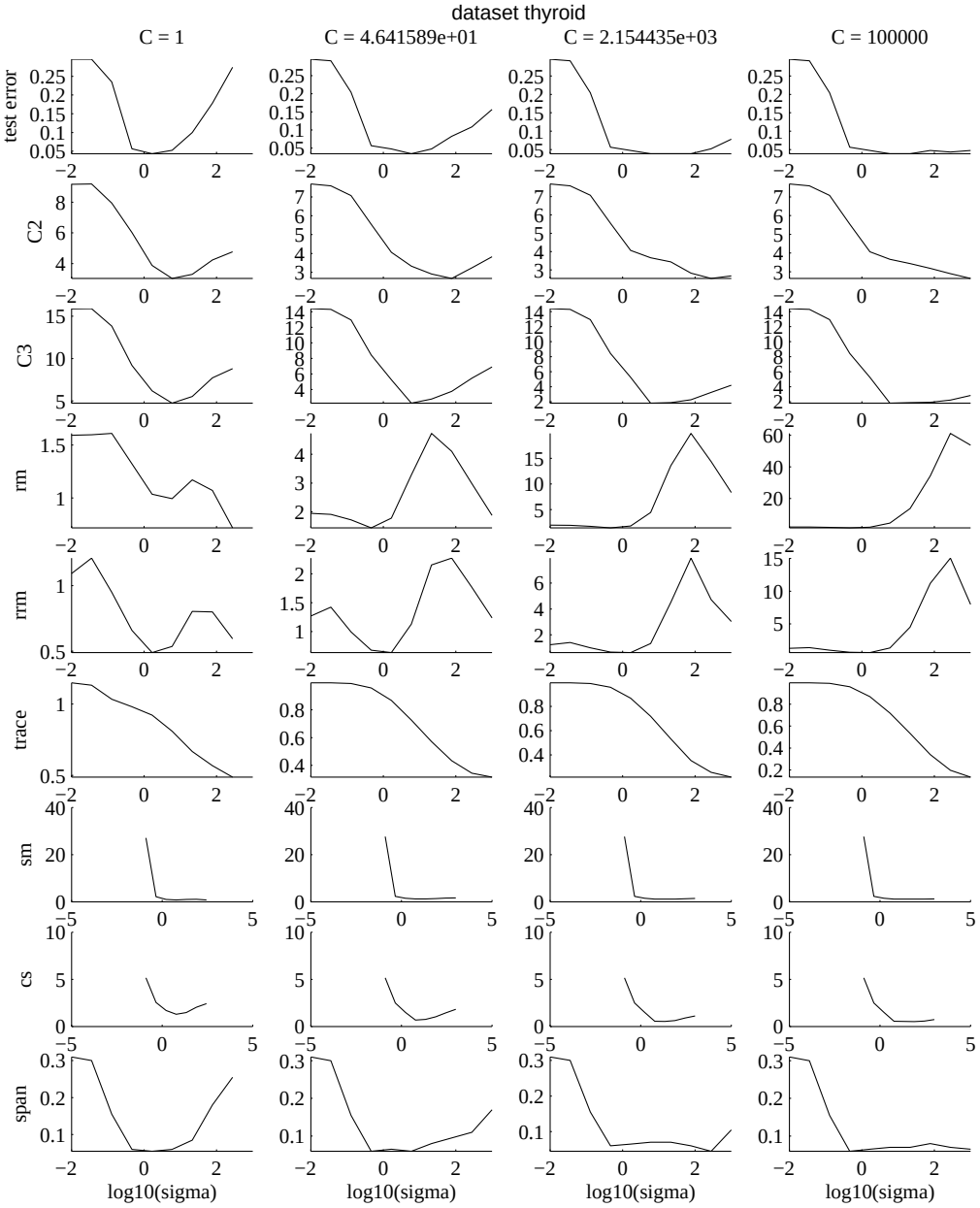


Figure 6, continued

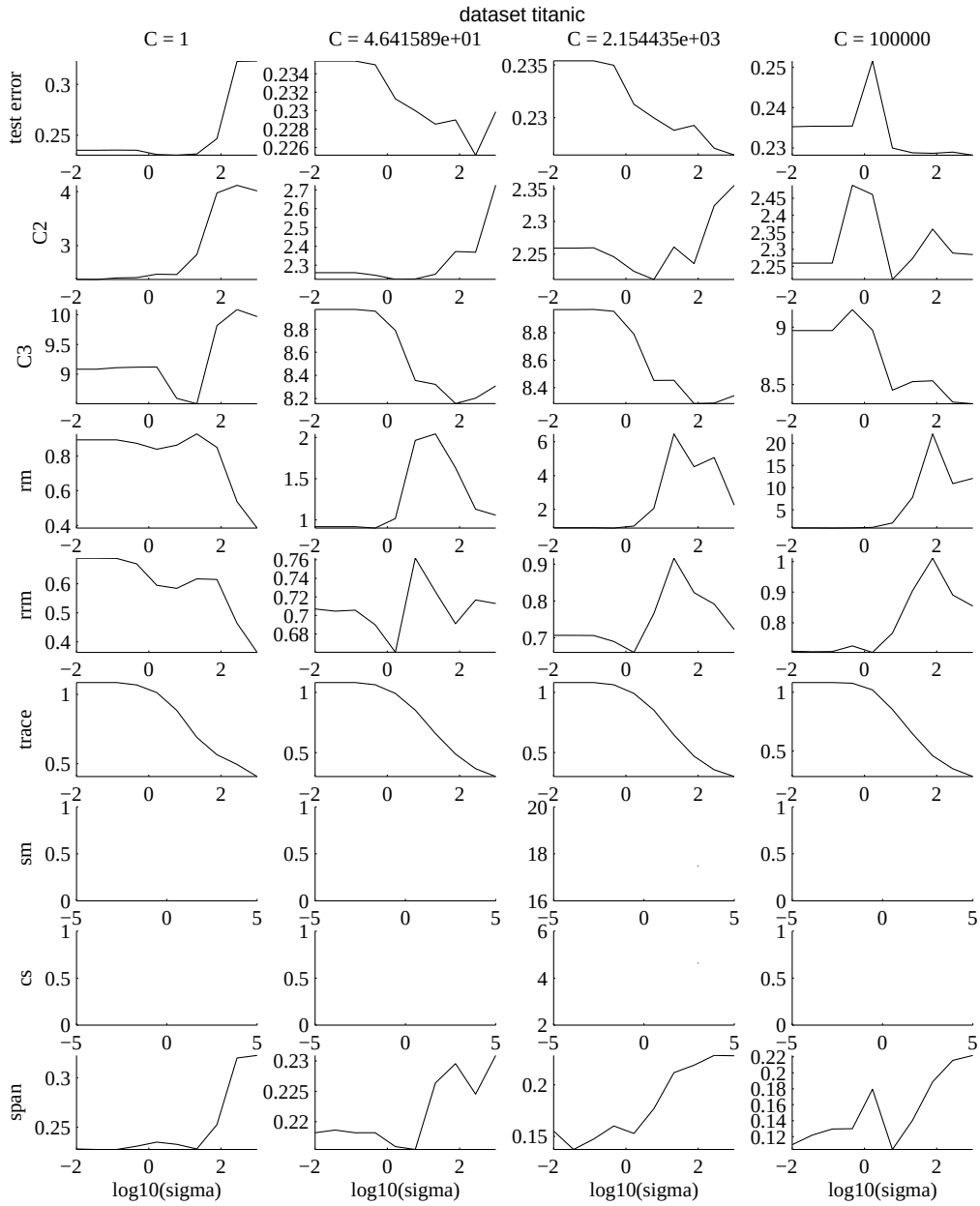


Figure 6, continued

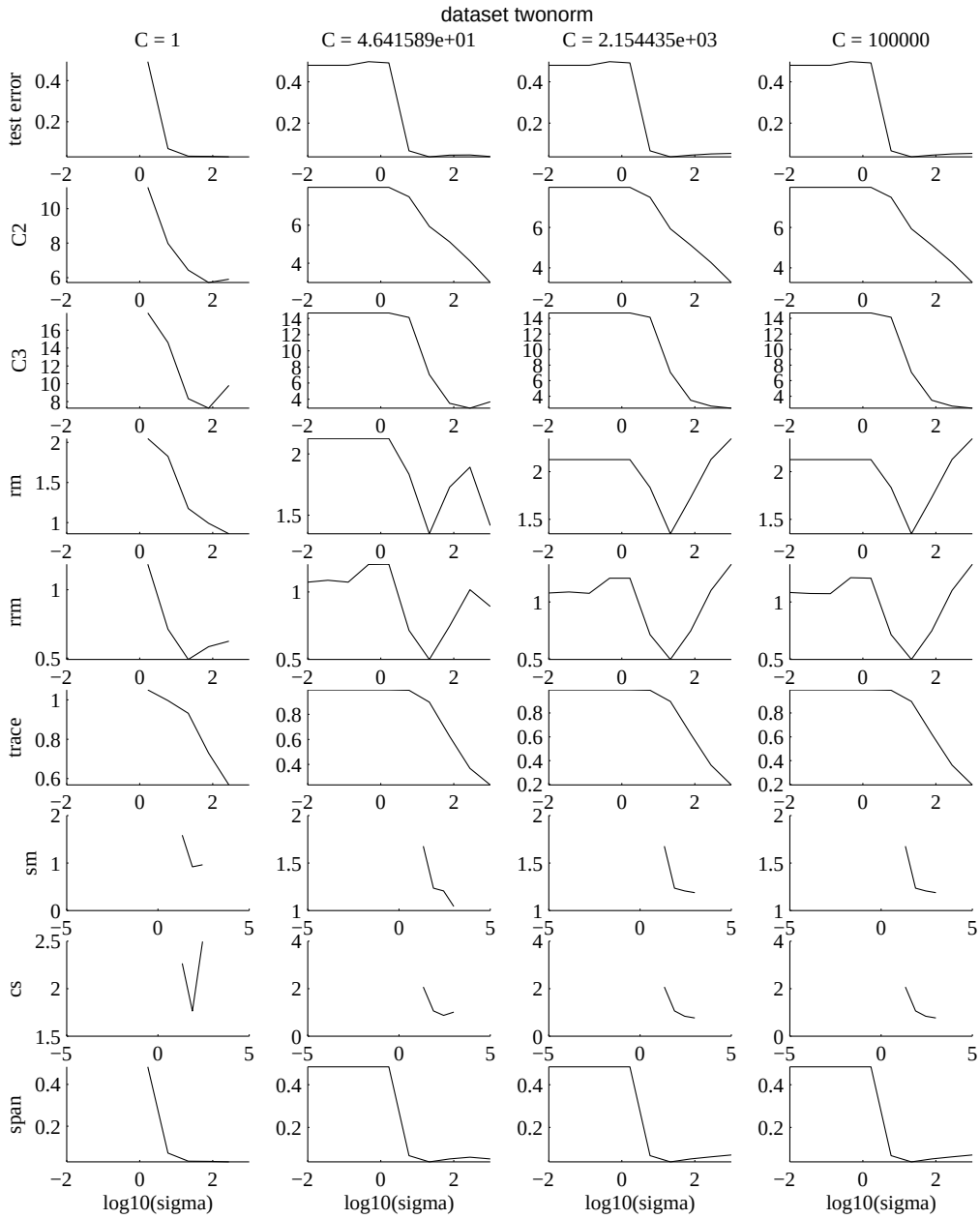


Figure 6, continued

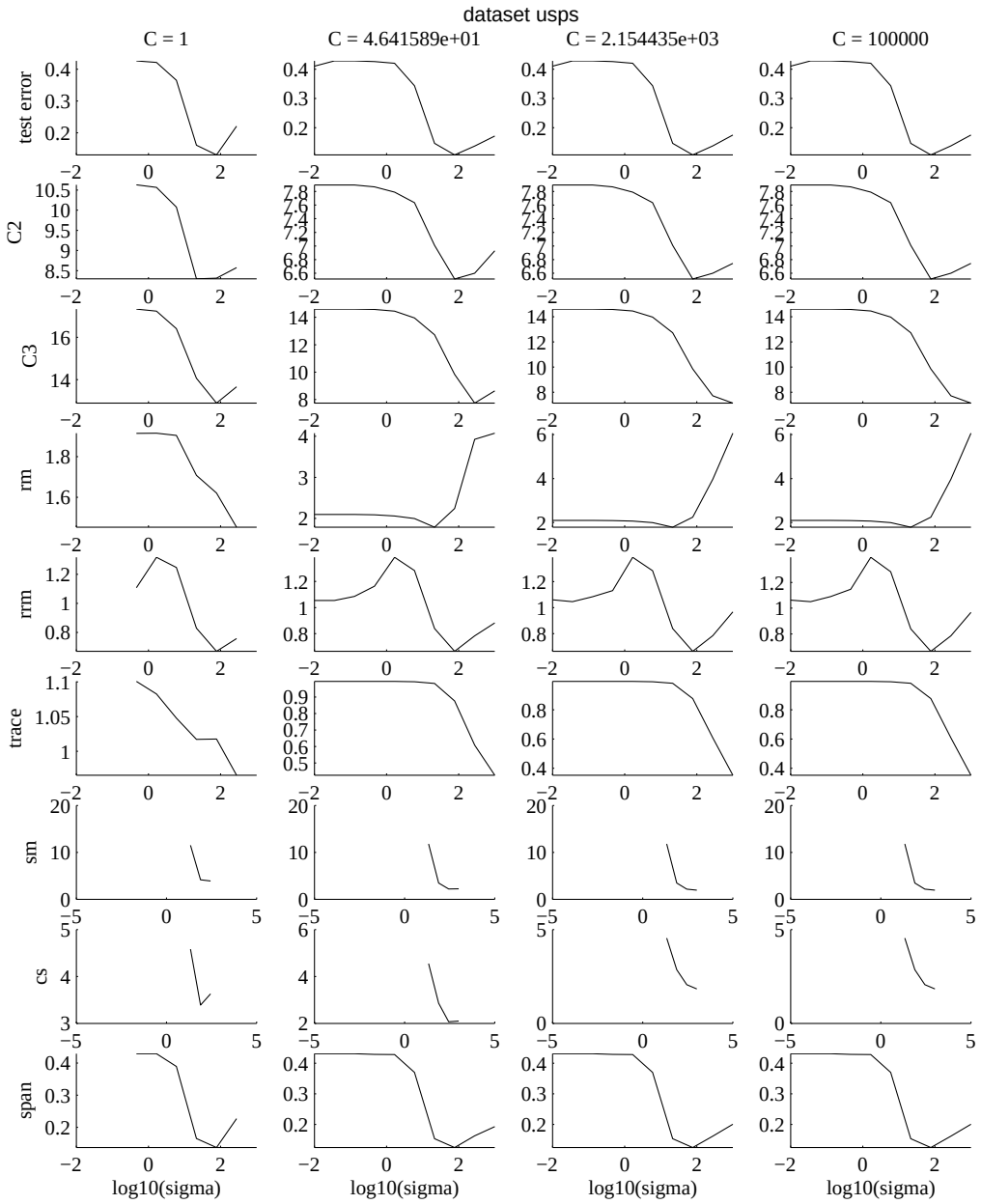


Figure 6, continued

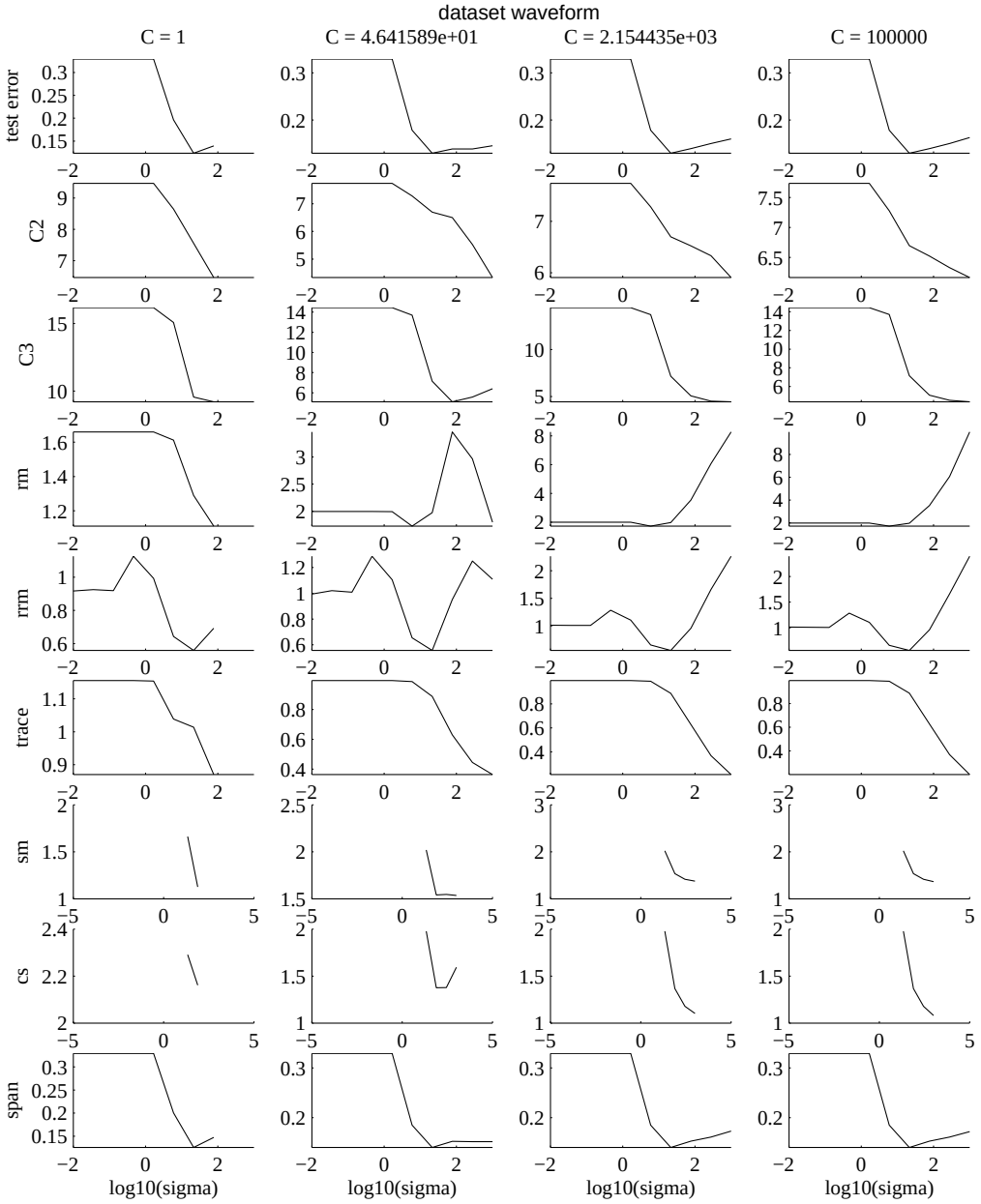


Figure 6, continued

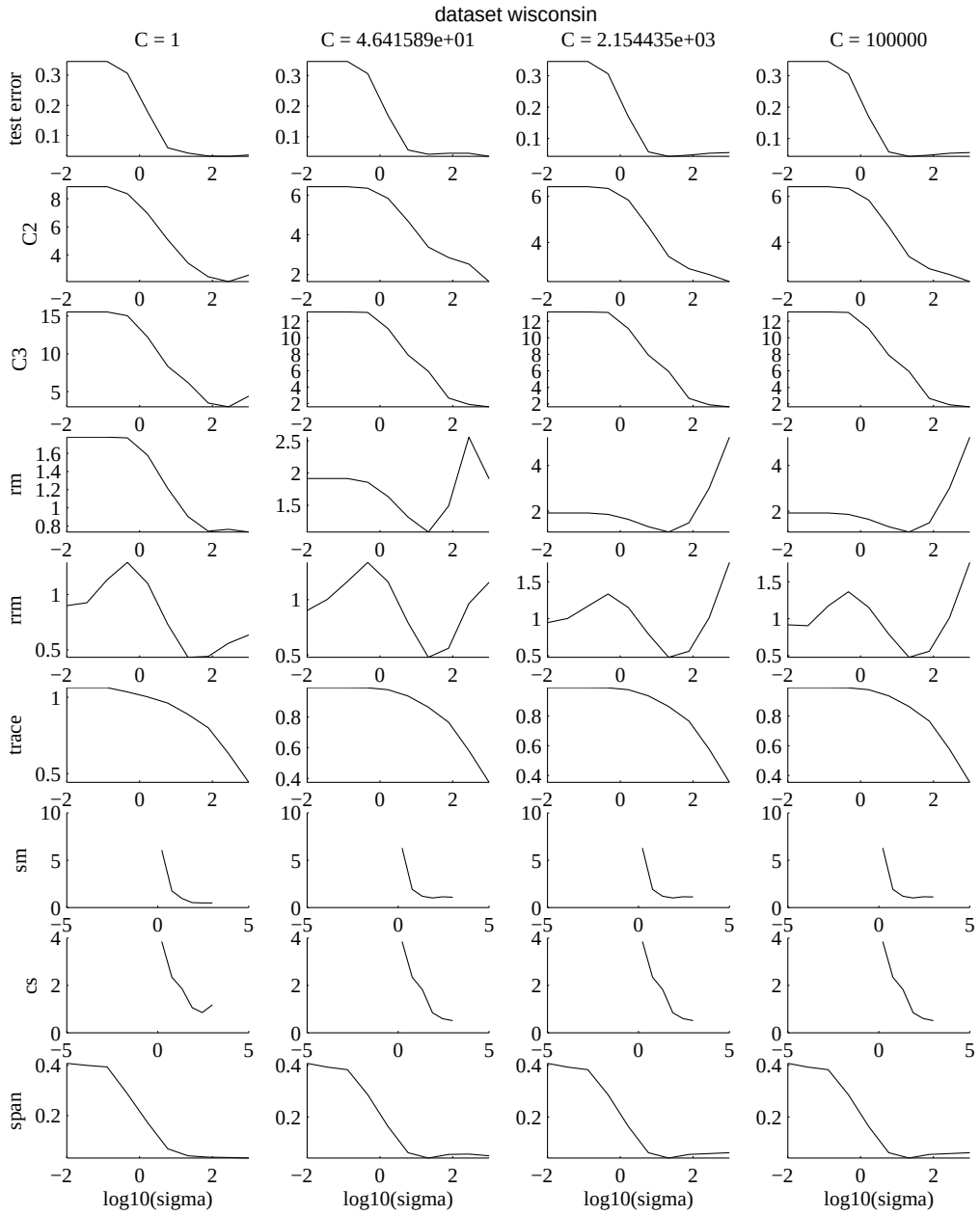


Figure 6, continued

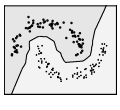
Notation

General:

$\mathcal{X}$	input space
$\mathcal{Y}$	output space
x_i or X_i	training patterns
y_i or Y_i	training labels
n	number of training points
$\mathbb{1}$	constant-1 function (or vector)
$\mathbb{0}$	constant-0 function (or vector)
$\mathbb{1}_A$	characteristic function of the set A
$(C(\mathcal{X}), \|\cdot\|_\infty)$	space of continuous functions on $\mathcal{X}$ with infinity norm
P	probability distribution on the data space
P_n	empirical distribution
ℓ	loss function
$\mathcal{R}$	true risk (p. 70)
$\mathcal{R}_{\text{emp}}$	empirical risk (p. 70)
$\mathcal{R}_{\text{reg}}$	regularized risk (p. 71)
R_n	Rademacher complexity (p. 71)
$\tilde{R}_n$	maximum discrepancy (p. 71)
$\hat{R}_n$	empirical Rademacher complexity (p. 71)
$N(\mathcal{F}, \varepsilon, d)$	ε-covering numbers of the space $\mathcal{F}$ with respect to metric d (p. 94)

Chapter II:

k	similarity function
K_n	similarity or kernel matrix of size $n \times n$ (p. 22)
D_n	degree matrix (p. 22)
L_n	unnormalized graph Laplacian (p. 22)
L'_n	symmetrically normalized graph Laplacian (p. 22)
L''_n	row-sum normalized graph Laplacian (p. 22)



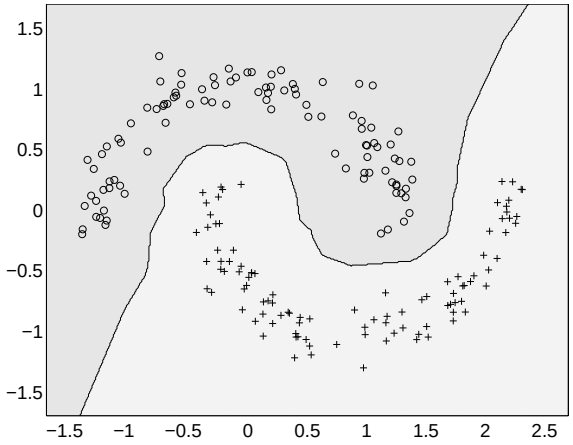
H'_n	symmetrically normalized similarity matrix (p. 23)
H'_n	row-sum normalized similarity matrix (p. 23)
d and d_n	degree functions, true and empirical (p. 35)
h and h_n	symmetrically normalized similarity functions, true and empirical (p. 36)
g and g_n	row-sum normalized similarity functions, true and empirical (p. 37)
$S_n \xrightarrow{p} S$	pointwise convergence (p. 32)
$S_n \xrightarrow{c} S$	compact convergence (p. 32)
$S_n \xrightarrow{cc} S$	collectively compact convergence (p. 32)
$S_n \xrightarrow{\|\cdot\|} S$	convergence in operator norm (p. 32)
$S_n \xrightarrow{+-} S$	convergence up to a change of sign (p. 29)

Chapter III:

$(\mathcal{X}, d)$	metric space
$(\mathcal{X}_0, d_0)$	extended metric space (p. 81)
diam	diameter of a metric space (p. 80)
$L(f)$	Lipschitz constant of function f (p. 79)
$(\mathrm{Lip}(\mathcal{X}), \|\cdot\|_L)$	Lipschitz function space (p. 80)
$(\mathrm{Lip}_0(\mathcal{X}_0), L(\cdot))$	another Lipschitz function space (p. 80)
$(AE(\mathcal{X}), \|\cdot\|_{AE})$	Arens-Eells space (p. 81)

Chapter IV:

log	logarithm to base 2
R	radius of the smallest ball sphere the training data
ρ	margin
d	dimension of the feature space
$C_1 - C_5$	compression coefficients



The little movie in the right bottom corner of the pages illustrates the convergence of normalized spectral clustering on the two moons data set (courtesy of Dengyong Zhou). For each page, we randomly draw one additional data point and show the resulting clustering. As similarity function we used the Gaussian kernel with kernel width $\sigma = 0.2$. The separation line between the classes is computed by assigning to each point in the space the label of the nearest training point.

Bibliography

P. Anselone. *Collectively compact operator approximation theory*. Prentice-Hall, Englewood Cliffs, New Jersey, 1971.

M. Anthony. Uniform Glivenko-Cantelli theorems and concentration of measure in the mathematical modelling of learning. CDAM Research Report LSE-CDAM-2002-07, 2002.

R. Arens and J. Eells. On embedding uniform and topological spaces. *Pacific J. Math.*, 6:397–403, 1956.

C. Baker. *The numerical treatment of integral equations*. Oxford University Press, 1977.

S. Barnard, A. Pothen, and H. Simon. A spectral algorithm for envelope reduction of sparse matrices. *Numerical Linear Algebra with Applications*, 2(4):317–334, 1995.

A. Barron, J. Rissanen, and B. Yu. The minimum description length principle in coding and modeling. *IEEE Trans. Inform. Theory*, 44(6):2743 – 2760, 1998.

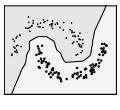
P. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. In D. Helmbold and B. Williamson, editors, *Proceedings of the 14th annual conference on Computational Learning Theory*, pages 273–288, 2001.

P. Bartlett and S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *JMLR*, 3:463–482, 2002.

P. L. Bartlett and J. Shawe-Taylor. Generalization performance of support vector machines and other pattern classifiers. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 43–54. MIT Press, Cambridge, MA, 1999.

M. Belkin. *Problems of Learning on Manifolds*. PhD thesis, University of Chicago, 2003.

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, June 2003a.



M. Belkin and P. Niyogi. Using manifold structure for partially labeled classification. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, Cambridge, MA, 2003b. MIT Press.

Y. Bengio, P. Vincent, J.-F. Paiement, O. Delalleau, M. Ouimet, and N. Le Roux. Spectral clustering and kernel PCA are learning eigenfunctions. Technical Report TR 1239, University of Montreal, 2003.

K. Bennett and E. Bredensteiner. Duality and geometry in SVM classifiers. In P. Langley, editor, *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 57–64. Morgan Kaufmann, San Francisco, 2000.

R. Bhatia. *Matrix Analysis*. Springer, New York, 1997.

T. De Bie, M. Momma, and N. Cristianini. Efficiently learning the metric with side-information. In *Algorithmic Learning Theory, 14th International Conference, ALT 2003, Sapporo, Japan, October 2003, Proceedings*, volume 2842 of *Lecture Notes in Artificial Intelligence*, pages 175–189. Springer, 2003.

M. Birman and M. Solomjak. *Spectral theory of self-adjoint operators in Hilbert space*. Reidel Publishing Company, Dordrecht, 1987.

A. Blum and J. Langford. PAC-MDL bounds. In B. Schölkopf and M.K. Warmuth, editors, *Learning Theory and Kernel Machines*, pages 344–357. 16th Annual Conference on Learning Theory, Springer, Berlin, 2003.

M. Bolla. Relations between spectral and classification properties of multigraphs. Technical Report 91-27, DIMACS Technical Report, 1991.

L. Bottou and Y. Bengio. Convergence properties of the k-means algorithm. In D. Touretzky, M. Mozer, and M. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*. MIT Press, Denver, 1995.

O. Bousquet. *Concentration Inequalities and Empirical Processes Theory Applied to the Analysis of Learning Algorithms*. PhD thesis, Ecole Polytechnique, 2002.

O. Bousquet, O. Chapelle, and M. Hein. Measure based regularization. In S. Thrun, L. Saul, and B. Schölkopf, editors, *Advances in Neural Information Processing Systems 16*. MIT Press, Cambridge, MA, 2004.

O. Bousquet and D. Herrmann. On the complexity of learning the kernel matrix. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems, 15*, Cambridge, MA, 2003. MIT Press.

Pierre Brémaud. *Markov chains: Gibbs fields, Monte Carlo simulation, and queues*, volume 31 of *Texts in Applied Mathematics*. Springer-Verlag, New York, 1999.

O. Chapelle and V. Vapnik. Model selection for support vector machines. In S. A. Solla, T. K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*. MIT Press, Cambridge, MA, 2000.

F. Chatelin. *Spectral Approximation of Linear Operators*. Academic Press, New York, 1983.

F. Chung. *Spectral graph theory*, volume 92 of *CBMS Regional Conference Series in Mathematics*. Published for the Conference Board of the Mathematical Sciences, Washington, DC, 1997.

J. Conway. *A Course in Functional Analysis*. Springer, New York, 1985.

T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley, New York, 1991.

T. Cox and A. Cox. *Multidimensional Scaling*. Chapman & Hall, London, 2nd edition, 2001.

N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, Cambridge, UK, 2000.

A. Cuevas, M. Febrero, and R. Fraiman. Cluster analysis: a further approach based on density estimation. *Computational Statistics and Data Analysis*, 36:441–459, 2001.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, New York, 1996.

L. Devroye and G. Lugosi. *Combinatorial Methods in Density Estimation*. Springer, New York, 2001.

V. Dobric and J. Yukich. Asymptotics for transportation costs in high dimensions. *J. Theor. Probab.*, 8(1):97–118, 1995.

W. E. Donath and A. J. Hoffman. Lower bounds for the partitioning of graphs. *IBM J. Res. Develop.*, 17:420–425, 1973.

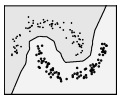
R. M. Dudley. Universal Donsker classes and metric entropy. *Ann. Probab.*, 15(4): 1306–1326, 1987.

N. Dunford and J. Schwartz. *Linear Operators, Part I*. Interscience Publishers, New York, 1957.

M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Math. J.*, 23:298–305, 1973.

S. Floyd and Manfred K. Warmuth. Sample compression, learnability, and the Vapnik-Chervonenkis dimension. *Machine Learning*, 21(3):269–304, 1995.

L. Goldfarb. A new approach to pattern recognition. In L. Kanal and A. Rosenfeld, editors, *Progress in Pattern Recognition*, volume 2, pages 241–402. Elsevier, North-Holland, 1985.



J. Gower. Measures of similarity, dissimilarity, and distance. In S. Kotz and N. Johnson, editors, *Encyclopedia of Statistical Sciences*, volume 5, pages 397–405. Wiley, New York, 1985.

J. Gower. Metric and Euclidean properties of dissimilarity coefficients. *Journal of classification*, 3:5–48, 1986.

T. Graepel, R. Herbrich, P. Bollmann-Sdorra, and K. Obermayer. Classification on pairwise proximity data. In M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11, pages 438–444. MIT Press, Cambridge, MA, 1999a.

T. Graepel, R. Herbrich, B. Schölkopf, A. Smola, P. Bartlett, K. Müller, K. Obermayer, and R. Williamson. Classification of proximity data with LP machines. In *Proceedings of the Ninth International Conference on Artificial Neural Networks*, pages 304–309, 1999b.

S. Guattery and G. L. Miller. On the quality of spectral separators. *SIAM Journal of Matrix Anal. Appl.*, 19(3), 1998.

L. Hagen and A.B. Kahng. New spectral methods for ratio cut partitioning and clustering. *IEEE Trans. Computer-Aided Design*, 11(9):1074–1085, 1992.

M. Hansen and B. Yu. Model selection and the principle of minimum description length. *Journal of the American Statistical Association*, 96(454):746–774, 2001.

J. Hartigan. *Clustering algorithms*. Wiley, New York, 1975.

J. Hartigan. Consistency of single linkage for high-density clusters. *JASA*, 76(374): 388–394, 1981.

J. Hartigan. Statistical theory in clustering. *Journal of classification*, 2:63–76, 1985.

M. Hein and O. Bousquet. Maximal margin classification for metric spaces. In M. Warmuth B. Schölkopf, editor, *Proceedings of the 16. Annual Conference on Computational Learning Theory*, pages 72–86. Springer Verlag, Heidelberg, 2003.

B. Hendrickson and R. Leland. An improved spectral graph partitioning algorithm for mapping parallel computations. *SIAM J. on Scientific Computing*, 16:452–469, 1995.

R. Herbrich, T. Graepel, and J. Shawe-Taylor. Sparsity vs. large margins for linear classifiers. In N. Cesa-Bianchi and S. Goldman, editors, *Proceedings of the Thirteenth Annual Conference on Computational Learning Theory*, pages 304–308. Morgan Kaufmann, San Francisco, 2000.

D. Higham and M. Kibble. A unified view of spectral clustering. Mathematics Research Report 2, University of Strathclyde, 2004.

D. Hochbaum and D. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of Operations Research*, 10(2):180–184, 1985.

N. Jardine and R. Sibson. *Mathematical taxonomy*. Wiley, London, 1971.

R. Kannan, S. Vempala, and A. Vetta. On clusterings - good, bad and spectral. Technical report, Computer Science Department, Yale University, 2000.

T. Kato. *Perturbation theory for linear operators*. Springer, Berlin, 1966.

J. Kleinberg. An impossibility theorem for clustering. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 446–453. MIT Press, Cambridge, MA, 2003.

A. N. Kolmogorov and V. M. Tihomirov. ε-entropy and ε-capacity of sets in functional space. *Amer. Math. Soc. Transl. (2)*, 17:277–364, 1961.

V. Koltchinskii. Asymptotics of spectral projections of some random matrices approximating integral operators. *Progress in Probabilty*, 43, 1998.

V. Koltchinskii and E. Giné. Random matrix approximation of spectra of integral operators. *Bernoulli*, 6(1):113 – 167, 2000.

G. Lanckriet, N. Cristianini, P. Bartlett, L. El Ghaoui, and M. Jordan. Learning the kernel matrix with semidefinite programming. *JMLR*, 5:27–72, 2004.

J. Lember. On minimizing sequences for k-centres. *Journal of Approximation Theory*, 120:20–35, 2003.

G. Lugosi and A. Nobel. Consistency of data-driven histogram methods for density estimation and classification. *Ann. Statist.*, 24(2):687–706, 1996.

K. Mardia, J. Kent, and J. Bibby. *Multivariate Analysis*. Academic Press, London, 1979.

D. McAllester. Some PAC–Bayesian theorems. *Machine Learning*, 37(3):355–363, 1999.

R. Megginson. *An introduction to Banach space theory*. Springer, New York, 1998.

M. Meila and J. Shi. A random walks view of spectral segmentation. In *8th International Workshop on Artificial Intelligence and Statistics*, 2001.

S. Mendelson and R. Vershynin. Entropy and the combinatorial dimension. *Inventiones Mathematicae*, 152(1):37–55, 2003.

B. Mohar. The Laplacian spectrum of graphs. In *Graph theory, combinatorics, and applications. Vol. 2 (Kalamazoo, MI, 1988)*, Wiley-Intersci. Publ., pages 871–898. Wiley, New York, 1991.

A. Ng, M. Jordan, and Y. Weiss. On spectral clustering: Analysis and an algorithm. In T. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*. MIT Press, 2001.

P. Niyogi and N. K. Karmarkar. An approach to data reduction and clustering with theoretical guarantees. In P. Langley, editor, *Proceedings of the Seventeenth International Conference on Machine Learning*. Morgan Kaufmann, San Francisco, 2000.

M. Opper and O. Winther. Gaussian processes and SVM: mean field and leave-one-out. In A. J. Smola, P. L. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 311–326. MIT Press, Cambridge, MA, 2000.

E. Pekalska, P. Paclík, and R. Duin. A generalized kernel approach to dissimilarity based classification. *Journal of Machine Learning Research*, 2:175–211, 2001. Special Issue "New Perspectives on Kernel Based Learning Methods".

V. Pestov. Free Banach spaces and representations of topological groups. *Funct. Anal. Appl*, 20:70–72, 1986.

D. Pollard. Strong consistency of k-means clustering. *Annals of Statistics*, 9(1): 135–140, 1981.

J. Puzicha, T. Hofmann, and J. Buhmann. A theory of proximity based clustering: Structure detection by optimization. *Pattern Recognition*, 33(4):617–634, 2000.

G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for AdaBoost. *Machine Learning*, 42(3):287–320, March 2001.

S. Roweis and L. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290:2323–2326, 2000.

W. Rudin. *Functional Analysis*. McGraw-Hill Inc., Singapore, 2nd edition, 1991.

I. Schoenberg. Metric spaces and positive definite functions. *TAMS*, 44:522–536, 1938.

B. Schölkopf. The kernel trick for distances. In T. G. Dietterich T. K. Leen and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*. MIT Press, Cambridge, MA, 2001.

B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

J. Shawe-Taylor, C. Williams, N. Cristianini, and J. Kandola. On the eigenspectrum of the Gram matrix and its relationship to the operator eigenspectrum. In N. Cesa-Bianchi, M. Numao, and R. Reischuk, editors, *Proceedings of the 13th International Conference on Algorithmic Learning Theory*. Springer, Heidelberg, 2002.

J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

A. J. Smola, B. Schölkopf, and K.-R. Müller. The connection between regularization operators and support vector kernels. *Neural Networks*, 11:637–649, 1998.

D. Spielman and S. Teng. Spectral partitioning works: planar graphs and finite element meshes. In *37th Annual Symposium on Foundations of Computer Science (Burlington, VT, 1996)*, pages 96–105. IEEE Comput. Soc. Press, Los Alamitos, CA, 1996.

J. M. Steele. *Probability theory and combinatorial optimization*, volume 69 of *CBMS-NSF Regional Conference Series in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1997.

M. Talagrand. The Ajtai-Komlós-Tusnády matching theorem for general measures. In *Probability in Banach spaces, 8 (Brunswick, ME, 1991)*, volume 30 of *Progr. Probab.*, pages 39–54. Birkhäuser Boston, Boston, MA, 1992.

A. Taylor. *Introduction to Functional Analysis*. Wiley, New York, 1958.

J. Tenenbaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2323, 2000.

A. W. van der Vaart and J. A. Wellner. *Weak Convergence and Empirical Processes*. Springer, New York, 1996.

R. Van Driessche and D. Roose. An improved spectral bisection algorithm and its application to dynamic load balancing. *Parallel Comput.*, 21(1), 1995.

V. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, New York, 1995.

V. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

V. Vapnik and O. Chapelle. Bounds on error expectation for support vector machines. *Neural Computation*, 12(9):2013–2036, 2000.

R. Veltkamp and M. Hagedoorn. Shape similarity measures, properties and constructions. In *Advances in Visual Information Systems, Proceedings of the 4th International Conference*, pages 467–476. Springer, 2000.

D. Verma and M. Meila. A comparison of spectral clustering algorithms. Technical Report UW CSE Technical report 03-05-01, University of Washington, 2003.

U. von Luxburg and O. Bousquet. Distance-based classification with Lipschitz functions. In B. Schölkopf and M.K. Warmuth, editors, *Proceedings of the 16th Annual Conference on Learning Theory (COLT)*, pages 314–328. Springer, 2003. This article won the prize for the best student paper.

U. von Luxburg and O. Bousquet. Distance-based classification with Lipschitz functions. *Journal for Machine Learning Research*, 5:669–695, 2004.

U. von Luxburg, O. Bousquet, and M. Belkin. Limits of spectral clustering. In *Advances in Neural Information Processing Systems 18*, 2004a. to appear.

U. von Luxburg, O. Bousquet, and M. Belkin. On the convergence of spectral clustering on random samples: the normalized case. In J. Shawe-Taylor and Y. Singer, editors, *Proceedings of the 17th Annual Conference on Learning Theory (COLT)*, pages 457–471. Springer, 2004b.

U. von Luxburg, O. Bousquet, and B. Schölkopf. A compression approach to support vector model selection. *Journal for Machine Learning Research*, 5:293–323, 2004c.

S. Watson. The classification of metrics and multivariate statistical analysis. *Topology and its Applications*, 99:237–261, 1999.

N. Weaver. *Lipschitz algebras*. World Scientific, Singapore, 1999.

J. Weidmann. *Linear Operators in Hilbert spaces*. Springer, New York, 1980.

Y. Weiss. Segmentation using eigenvectors: A unifying view. In *Proceedings of the International Conference on Computer Vision*, pages 975–982, 1999.

C. K. I. Williams and M. Seeger. The effect of the input density distribution on kernel-based classifiers. In P. Langley, editor, *Proceedings of the 17th International Conference on Machine Learning*, pages 1159–1166. Morgan Kaufmann, San Francisco, 2000.

W. Wright. A formalization of cluster analysis. *Pattern Recognition*, 5:273–282, 1973.

E. Xing, A. Ng, M. Jordan, and S. Russell. Distance metric learning with application to clustering with side-information. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 505–512. MIT Press, Cambridge, MA, 2003.

S. Zhong and J. Ghosh. A unified framework for model-based clustering. *JMLR*, 4: 1001–1037, 2003.

D. Zhou and B. Schölkopf. Learning from labeled and unlabeled data using random walks. Technical report, Max Planck Institute for Biological Cybernetics, March 2004.

D. Zhou, B. Xiao, H. Zhou, and R. Dai. Global geometry of SVM classifiers. Technical Report 30-5-02, Institute of Automation, Chinese Academy of Sciences, 2002.

X. Zhu, Z. Ghahramani, and J. Lafferty. Semi-supervised learning using Gaussian fields and harmonic functions. In T. Fawcett and N.Mishra, editors, *Proceedings of the 20th International Conference of Machine Learning*. AAAI Press, 2003.