Learning with Kernels

vorgelegt von
Diplom–Physiker
Alexander Johannes Smola

Vom Fachbereich 13 — Informatik
der Technischen Universität Berlin
zur Erlangung des akademischen Grades

Doktor der Naturwissenschaften
– Dr. rer. nat. –

Promotionsausschuss:

Vorsitzender: Prof. Dr. A. Biedl
Berichter: Prof. Dr. S. Jähnichen
Berichter: Prof. Dr. J. Shawe–Taylor

Tag der wissenschaftlichen Aussprache:
12. November 1998, 16.00 Uhr

Berlin 1998
– D 83 –

Foreword

The present thesis can take its place among the numerous doctoral theses and other publications that are currently revolutionizing the area of machine learning. The author's basic concern is with kernel–based methods and in particular Support Vector algorithms for regression estimation for the solution of inverse, often ill–posed problems. However, Alexander Smola's thesis stands out from many of the other publications in this field. This is due in part to the author's profound theoretical penetration of his subject–matter, but also and in particular to the wealth of detailed results he has included.

Especially neat and of particular relevance are the algorithmic extensions of Support Vector Machines, which can be combined as building blocks, thus markedly improving the Support Vectors. Of substantial interest is also the very elegant unsupervised method for nonlinear feature extraction, which applies the kernel-based method to classical Principal Component Analysis (kernel PCA). And although only designed to illustrate the theoretical results, the practical applications the author gives us from the area of high-energy physics and time-series analysis are highly convincing.

In many respects the thesis is groundbreaking, but it is likely to soon become a frequently cited work for numerous innovative applications from the field of statistical machine learning and for improving our theoretical understanding of Support Vector Machines.

Stefan Jähnichen, Professor, Technische Universität Berlin
Director, GMD Berlin

Alex Smola's thesis has branched out in at least five novel directions broadly based around kernel learning machines: analysis of cost functions, relations to regularization networks, optimization algorithms, extensions to unsupervised learning including regularized principal manifolds, entropy numbers for linear operators and applications to bounding covering numbers. I will highlight some of the significant contributions made in each of these areas.

Cost Functions This section presents a very neat coherent view of cost functions and their effect on the overall algorithmics of kernel regression. In particular, it is shown how using a general convex cost function still allows the problem to be cast as a convex programming problem solvable via the dual. Experiments show that choosing the right cost function can improve performance. The section goes on to describe a very useful approach to choosing the ϵ for the ϵ-insensitive loss measure, based on traditional statistical methods. Further refinements arising from this approach give a new algorithm termed ν-SV regression.

Kernels and Regularization The chapter covers the relation between kernels used in Support Vector Machines and Regularization Networks. This connection is a very valuable contribution to understanding the operation of SV machines and in particular their generalization properties. The analysis of particular kernels and experiments showing the effects of their regularization properties are very illuminating. Consideration of higher dimensional input spaces is made and the case of dot product kernels studied in some detail. This leads to the introduction of Conditionally Positive Definite Kernels and semiparametric estimation, both of which are new in the context of SV machines.

Optimization Algorithms This section takes the interior point methods and implements them for SV regression and classification. By taking into account the specifics of the problem efficiency savings have been made. The consideration then turns to subset selection to handle large data sets. This introduces among other techniques, SMO or sequential minimal optimization. This approach is generalized to the regression case and proves an extremely efficient method.

Unsupervised Learning The extension to Kernel PCA is a nice idea which appears to work well in practice. The further work on Regularized Principal Manifolds is very novel and opens up a number of interesting problems and techniques.

Entropy numbers, kernels and operators The estimation of covering numbers via techniques from operator theory is another major contribution to the state-of-the-art. Many new results are presented, among others the generalization bounds for Regularized Principal Manifolds are given.

John Shawe–Taylor, Professor, Royal Holloway, University of London

To my parents

Abstract

Support Vector (SV) Machines combine several techniques from statistics, machine learning and neural networks. One of the most important ingredients are kernels, i.e. the concept of transforming linear algorithms into nonlinear ones via a map into feature spaces. The present work focuses on the following issues:

- Extensions of Support Vector Machines.
- Extensions of kernel methods to other algorithms such as unsupervised learning.
- Capacity bounds which are particularly well suited for kernel methods.

After a brief introduction to SV regression it is shown how the classical ε–insensitive loss function can be replaced by other cost functions while keeping the original advantages or adding other features such as automatic parameter adaptation.

Moreover the connection between kernels and regularization is pointed out. A theoretical analysis of several common kernels follows and criteria to check Mercer's condition more easily are presented. Further modifications lead to semiparametric models and greedy approximation schemes. Next three different types of optimization algorithms, namely interior point codes, subset selection algorithms, and sequential minimal optimization (including pseudocode) are presented. The primal–dual framework is used as an analytic tool in this context.

Unsupervised learning is an extension of kernel methods to new problems. Besides Kernel PCA one can use the regularization to obtain more general feature exractors. A second approach leads to regularized quantization functionals which allow a smooth transition between the Generative Topographic Map and Principal Curves.

The second part deals with uniform convergence bounds for the algorithms and concepts presented so far. It starts with a brief self contained overview and an introduction to functional analytic tools which play a crucial role in this problem. By viewing the class of kernel expansions as an image of a linear operator one may give bounds on the generalization ability of kernel expansions even when standard concepts like the VC dimension fail or give too conservative estimates.

In particular it is shown that it is possible to compute the covering numbers of the given hypothesis classes *directly* instead of taking the detour via the VC dimension. Applications of the new tools to SV machines, convex combinations of hypotheses (i.e. boosting and sparse coding), greedy approximation schemes, and principal curves conclude the presentation.

Keywords Support Vectors, Regression, Kernel Expansions, Regularization, Statistical Learning Theory, Uniform Convergence;

Support Vektor (SV) Maschinen verbinden verschiedene Techniken der Statistik, des maschinellen Lernens und Neuronaler Netze. Eine Schlüsselposition fällt den Kernen zu, d.h. dem Konzept, lineare Algorithmen durch eine Abbildung in Merkmalsräume nichtlinear zu machen. Die Dissertation behandelt folgende Aspekte:

- Erweiterungen des Support Vektor Algorithmus
- Erweiterungen und Anwendungen kernbasierter Methoden auf andere Algorithmen wie das unüberwachte Lernen
- Abschätzungen zur Generalisierungsfähigkeit, die besonders auf kernbasierte Methoden abgestimmt sind

Nach einer kurzen Einführung in die SV Regression wird gezeigt, wie die ε unempfindliche Kostenfunktion durch andere Funktionen ersetzt werden kann, während gleichzeitig die Vorteile des ursprünglichen Algorithmus erhalten bleiben, oder auch neue Eigenschaften wie automatische Parameteranpassung hinzugefügt werden.

Weiterhin wird die Verbindung zwischen Kernen und Regularisierung aufgezeigt. Es folgt eine theoretische Analyse verschiedener häufig verwendeter Kerne, nebst Kriterien zur leichten Überprüfung von Mercers Bedingung. Weitere Veränderungen führen zu semiparametrischen Modellen sowie "geizigen" Näherungsverfahren. Abschließend werden drei Optimierungsalgorithmen vorgestellt, nämlich die Methode der inneren Punkte, Auswahlalgorithmen und sequentiell minimale Optimierung. Als analytisches Werkzeug fungiert hier das primär–duale Konzept der Optimierunge. Auch Pseudocode wird in diesem Zusammenhang zur Verfügung gestellt.

Unüberwachtes Lernen ist ein Anwendungsfall kernbasierter Methoden auf neue Probleme. Neben Kern PCA kann man das Regularisierungskonzept dazu verwenden, allgemeinere Mermalsextraktoren zu erhalten. Ein zweiter Ansatz führt zu einem stufenlosen Übergang zwischen der der erzeugenden topographischen Abbildung (GTM) und Hauptkurven.

Der zweite Teil der Dissertation beschäftigt sich mit Abschätzungen zur uniformen Konvergenz für die bisher vorgestellten Algorithmen und Konzepte. Dazu wird zuerst kurz ein Überblick über existierende Techniken zur Kapazitätskontrolle und Funktionalanalysis gegeben. Letztere spielen eine entscheidende Rolle, da die Klasse der Kernentwicklungen als Bild unter einem linearen Operator aufgefaßt werden kann, was Abschätzungen der Generalisierungsfähigkeit sogar in den Fällen ermöglicht, in denen klassische Ansätze wie die VC Dimension versagen bzw. zu konservative Abschätzungen geben.

Insbesondere wird gezeigt, daß es möglich ist, die Überdeckungszahlen einer gegebenen Hypothesenklasse *direkt* zu berechnen, ohne den Umweg über die Berechnung der VC Dimension zu nehmen. Anwendungen finden die neuen Methoden bei Support Vektor Maschinen, Konvexkombinationen von Hypothesen (z.B. Boosting und spärliche Kodierung), "geizigen" Näherungsverfahren und Hauptkurven.

Schlagworte Support Vektoren, Regression, Kernentwicklungen, Regularisierung, Statistische Lerntheorie, Uniforme Konvergenz;

Preface

The goal of this thesis is to give a self contained overview over Support Vector Machines and similar kernel based methods, mainly for Regression Estimation. It is, in this sense, complementary to Bernhard Schölkopf's work on Pattern Recognition. Yet it also contains new insights in capacity control which can be applied to classification problems as well.

It is probably best to view this work as a technical description of a toolset, namely the building blocks of a Support Vector Machine. The first part describes its basic machinery and the possible add–ons that can be used for modifying it, just like the leaflet one would get from a car dealer with a choice of all the 'extras' available.

In this respect the second part could be regarded as a list of operating instructions, namely how to effectively carry out capacity control for a class of systems of the SV type.

How to read this Thesis

I tried to organize this work both in a self contained, and modular manner. Where necessary, proofs have been moved into the appendix of the corresponding chapters and can be omitted if the reader is willing to accept some results on faith. Some fundamental results, however, if needed to understand the further way of reasoning, are derived in the main body.

How not to read this Thesis

This is the work of a physicist who decided to do applied statistics, ended up in a computer science department, and sometimes had engineering applications or functional analysis in mind. Hence it provides a mixture of techniques and concepts from several domains, sufficient to annoy many readers, due to the lack of mathematical rigor (from a mathematician's point), the sometimes rather theoretical reasoning and some technical proofs (from a practicioner's point), the lack of hardly any connection with physics, or some algorithms that work, but have not (yet) been proven to be optimal or to terminate in a finite number of steps, etc. However, I tried to split the nuisance equally among the disciplines.

Acknowledgements

I would like to thank the researchers at the Neuro group of GMD FIRST with whom I had the pleasure of carrying out research in an excellent environment. In particular the discussions with Peter Behr, Thilo Frieß, Jens Kohlmorgen, Steven Lemm, Sebastian Mika, Takashi Onoda, Petra Philips, Gunnar Rätsch, Andras Ziehe and many others were often inspiring. Besides that, the system administrators Roger Holst and Andreas Schulz helped in many ways.

The second lab to be thanked is the machine learning group at ANU, Canberra. Peter Bartlett, Jon Baxter, Shai Ben–David, and Lew Mason helped me in getting a deeper understanding of Statistical Learning Theory. The research visit at ANU was also one of the most productive periods of the past two years.

Next to mention are two departments at AT&T and Bell Laboratories, with Leon Bottou, Chris Burges, Yann LeCun, Patrick Haffner, Craig Nohl, Patrice Simard, and Charles Stenard. Much of the work would not have been possible, if I had not had the chance to learn about Support Vectors in their, then, joint research facility in Holmdel, USA. I am particularly grateful to Vladimir Vapnik in this respect. His "hands on" approach to statistics was a reliable guide to interesting problems.

Moreover I had the fortune to discuss with and get help from people like Bernd Carl, Nello Cristianini, Leo van Hemmen, Adam Krzyzak, David MacKay, Heidrun Mündlein, Noboru Murata, Manfred Opper, John Shawe–Taylor, Mark Stitson, Sara Solla, Grace Wahba, and Jason Weston. Chris Burges, Andre Eliseeff, Ralf Herbrich, Olvi Mangasarian, Klaus–Robert Müller, Bernhard Schölkopf, John Shawe–Taylor, Anja Westerhoff and Robert Williamson gave helpful comments on the thesis and found many errors.

Special thanks go the three people with whom most of the results of this thesis were obtained — Klaus-Robert Müller, Bernhard Schölkopf, and Robert Williamson: thanks to Klaus in particular for always reminding me of the necessity that theoretical advances have to be backed by experimental evidence, for advice and discusions about learning theory and neural networks, and for allowing me to focus on research, quite undisturbed from administrative chores; thanks to Bernhard for many discussions which were a major source of ideas and for ensuring that the proofs stayed theoretically sound; thanks to Bob for teaching me many things about statistical learning theory and functional analysis, and many valuable discussions.

Close collaboration is only possible if complemented by friendship. Many of these researchers, in particular Bernhard, became good friends, far beyond the level of scientific cooperation.

Finally I would like to thank Stefan Jähnichen. He provided as head of GMD and dean of the TU Berlin computer science department the research environment in which new ideas could be developed. His wise advice and guidance in scientific matters and academic issues was very helpful.

This work was made possible through funding by ARPA, grants of the DFG JA 379/71 and JA 379/51, funding from the Australian Research Council, travel grants from the NIPS foundation and NEuroNet, and support of NeuroCOLT 2.

Contents

List of Figures

List of Tables

List of Symbols

b	constant offset (or threshold)
$B_n(x)$	B–Splines of order n
$\mathcal{B}$	index set for regularized principal manifolds
$c(x, y, f(x))$	cost function at (x, y) for the output $f(x)$
C	regularization constant ($C = 1/\lambda$)
$\mathfrak{c}$	Williamson's constant, currently < 103 but probably < 1.86
C_k	kernel constant, i.e. upper bound for $\lvert\phi_i(x)\rvert$
$\mathcal{C}^p$	p–times continously differentiable functions
d	dimensionality of input space
$d(x, y)$	distance between x and y in the metric of d
D_{ij}	Regularization Matrix $\langle Pk(x_i, \cdot), Pk(x_j, \cdot)\rangle$
$E[\xi]$	expected value of random variable ξ
$E_m[\xi]$	empirical mean (m sample) of random variable ξ
$e_n(X)$	n-th dyadic entropy number of set X
$\mathcal{E}$	ellipsoid containing $\Phi(\mathcal{X})$
$\mathcal{F}$	the set of functions contained in the hypothesis class
$\mathfrak{F}$	structure of hypothesis classes
G	Green's function, usually of P^*P
H	Hilbert Space
$H(x)$	Heavyside function, i.e. 1 if $x > 0$ and 0 otherwise
h	VC dimension
fat–h	(level) fat shattering VC dimension
i, j	index variables (i is sometimes the imaginary unit)
k	(Mercer) kernel
K	kernel matrix
K_{ij}	matrix entries $k(x_i, x_j)$
$\mathbb{K}$	either $\mathbb{R}$ or $\mathbb{C}$

$\mathfrak{L}(E, F)$	space of linear operators between Banach spaces E, F
m	number of training examples
$\mathbf{n}$	multi index
$\mathbb{N}$	the set of natural numbers
$\mathcal{N}$	covering number of a set / class of functions
$Pr\{x\}$	probablity of event x
$p(x)$	probability density function (pdf)
$p_{\mathrm{emp}}(x)$	empirical density function
$\mathfrak{p}$	famliy of densities of same type and different level
P	regularization operator
P^*	adjoint operator of P
$P(\omega)$	Fourier representation of P^*P
$Q[f]$	regularization term
$R[f]$	expected risk of f
$R_{\mathrm{q}}[f]$	expected quantization error (risk)
$R_{\mathrm{emp}}[f]$	empirical risk
$R_{\mathrm{q,emp}}[f]$	empirical quantization error
$R_{\mathrm{reg}}[f]$	regularized risk
$R_{\mathrm{q,reg}}[f]$	regularized quantization error
$\mathbb{R}$	the set of reals
$\mathfrak{R}(\mathcal{F}, X, \delta)$	deviation bound on $R[f]$ wrt. $R_{\mathrm{emp}}[f]$
$\mathfrak{S}$	feature space
$T(\xi)$	term in the dual formulation stemming from the cost function
$U_r(x)$	ball with radius r around x
V_{SV}	unit ball of admissible vectors for Kernel Feature Analysis
V_{LP}	linear programming "unit ball" of admissible vectors for KFA
w	weight vector
$\mathcal{X}$	space of input patterns
X	set of training patterns (sometimes also X^m)
x_i	the i–th pattern ($x_i \in X \subset \mathcal{X}$)
$\mathcal{Y}$	space of target values
Y	set of target values (sometimes also Y^m)
y_i	the i–th pattern ($y_i \in X \subset \mathcal{X}$)
Z	set of pairs from X and Y

α_i	Lagrange multiplier, also expansion coefficients
α	vector of all Lagrange multipliers
β_i, β	auxiliary expansion coefficients
δ	variational δ
$\delta_{x_i}(x)$	delta distribution centered at x_i
δ_{ij}	Kronecker symbol
∇	Gradient operator
$\epsilon_n(X)$	n-th entropy number of set X
ε	parameter of the ε-insensitive loss function
$\gamma(\cdot)$	parametrized curve
λ	regularization constants ($\lambda = 1/C$)
Λ	upper bound on $Q[f]$
$\mu(x)$	measure of x, also value of the KKT target
η_i	Lagrange multipliers corresponding to ξ_i
Δ	Laplace operator
Φ	map into feature space
ϕ_i	basis functions for feature map
σ	variance, also kernel width
ξ_i	slack variables
ν	trade off parameter for ν–SV regression
ω	(spatial) frequency
Ω	support of frequency domain

$\tilde{f}(\omega)$	Fourier transform of $f(x)$, sometimes also $F[f](\omega)$
$\langle x, y\rangle$	dot product between patterns x and y
$\|.\|$	norm (normally Euclidean distance $\|x\| := \sqrt{\langle x, x\rangle}$)
$\|\cdot\|_p$	p–norm, e.g. $\left(\sum_i \|x_i\|^p\right)^{1/p}$
$\|A\|$	operator norm of A, i.e. $\sup_x \|Ax\|/\|x\|$
$\vec{1}$	vector of ones
$\mathbf{1}$	identity operator, also unit matrix

$\mathrm{card}(X)$	cardinality of the set X
$\mathrm{co}\,(X)$	set of convex combinations of the set X
$\dim(X)$	dimensionality of X
$\ln$	logarithm to base e
$\log_2$	logarithm to base 2
$\mathrm{sinc}\,x$	sinc function $(\sin x)/x$
$\mathrm{span}\,(X)$	linear span of the set X
$\mathrm{Pow}[f](\omega)$	power spectrum of f
$\mathrm{vol}\,(X)$	volume of the set X
max	maximum of a set (or maximum operation)
argmax	argument of max
min	minimum of a set (or minimum operation)
argmin	argument of min

I Algorithms

"The algorithms for constructing the separating hyperplane considered above will be utilized for developing a battery of programs for pattern recognition."

[Vapnik, 1982, p. 364]

In recent years the number of different Support Vector (SV) algorithms [Vapnik, 1995, Vapnik et al., 1997, Smola and Schölkopf, 1998a, Smola et al., 1998b, Schölkopf et al., 1998a, 1996, Bennett, 1999, Weston et al., 1999] and other kernel based methods [Schölkopf et al., 1998a] has grown rapidly. This is due to both, the success of the method [Burges and Schölkopf, 1997], and the need to adapt it to particular problems. Consequently, Support Vector methods have been proposed for classification, regression estimation, solution of inverse problems, general cost functions, multiple operators, arbitrary kernel expansions, modified regularization methods, etc. — just to name a few of them.

It is therefore sometimes hard to find the basic idea that generates them all, or to custom tailor a SV like algorithm that fits ones needs best, without causing unnecessary overhead. Whilst learning theoretic concerns will be relegated to the second part of the thesis (chapters 7 to 9), the following chapters will give an overview over the basic idea of SVMs and possible directions for an extension of the framework.

This will include several ways how to extend or modify the standard SV framework. In particular it will cover three different ways how to restrict the set of admissible functions via regularization functionals, several different cost functions and how these can be dealt with, adaptive estimates and their asymptotic optimality, extensions to semiparametric modelling by changing the regularization functional and considerations on regularization operators for vector valued functions. Optimization algorithms are given to efficiently solve the problems described above. Finally, the ideas are applied to unsupervised learning via kernel methods.

Most of these concepts may be combined much in a way like the options of a menu (i.e. many choices go together quite well, other combinations may not yield overly satisfactory results, however the taste may vary). Due to the combinatorial nature, it is impossible (and of little use) to present all combinations one might think of explicitly. New algorithms therefore should be considered whether they may be filed into one of the categories presented below, or show a new way for even further modifications.

Roadmap

Chapter 1 explains the basic SV regression setting as introduced by Vapnik [1995]. This is the common base all further extensions and enhancements build on. Moreover it is (still) the most widespread setting, and also relatively simple to deal with. Hence it will serve as point of reference for new techniques. Some fundamental calculations are carried out with respect to this setting, such as the Wolfe dualization approach, which is by itself central to the convex programming setting of the problems. Basic notions of statistical learning theory are mentioned (they will be dealt with in full detail in part II of this work). In particular, the concept of risk functionals, both empirical and regularized, is explained. Finally, techniques to restrict the model class are described.

The next chapter (2) shows how SV machines can be adapted to a wider class of convex loss functions: the optimization problem is set up in a way to suit arbitrary convex functions and subsequently optimization equations are given for this extension. Experimental evidence shows that, not surprisingly, this increased degree of freedom may contribute to improved generalization. In a second step, an answer is sought to the question, which cost function, or more specifically, which setting of parameters, would be best to achieve good estimates. The answer can be stated in the context of asymptotical statistics. Experiments show that these findings carry over to the finite sample size case. To make this choice automatic, adaptive noise model selection is introduced. One can show that in this setting the choice of parameters, like the ε–insensitive loss zone, can be reduced to finding one parameter per class of noise models.

To make the (so far linear) estimators nonlinear, kernels are introduced in chapter 3. After a brief overview on admissible SV kernels and feature spaces, this chapter focuses on the regularization properties of SV machines. It is shown that the SV kernels correspond to implicit regularization by matching regularization operators, hence that SV machines and regularization networks are closely related. This finding is exploited to analyze some commonly used translation invariant kernels such as B_{2n+1} splines, Gaussian kernels, and the (somewhat underperforming) Dirichlet kernel. Some considerations on kernels when prior knowledge is available and an analysis of translation invariant kernels in higher dimensions conclude this section. After the analysis of scalar valued functions a uniqueness theorem regarding vector valued functions is stated. It shows that all regularization operators for vector valued functions, satisfying certain homogeneity and permutation symmetries, have to be scalar. This rules out the construction of possibly very elaborate extensions of the scalar case (hence saves time seeking nonexistent extensions). As polynomial kernels are rather relevant in practice, they are also looked at in the subsequent section. It provides a regularization theoretic interpretation of (in)homogeneous kernels. Finally the class of admissible kernels is extended to conditionally positive definite ones, thus largely extending the SV setting to cases like (thin plate) splines and other expansions widely used in interpolation theory. This leads to a modification of the standard SV algorithm to account for these changes. The new algorithm leads directly to what can be called semiparametric SV estimation. It exploits the fact that one can easily modify the nullspace of operators in an explicit way (as done before when dealing with conditionally positive definite functions) by adding the corresponding basis functions. This also leads to (mixed) linear programming regularizers and sparse coding versions of SV machines. A worked through example of a particular regularization operator ends the discussion on kernels and regularization.

Chapter 4 may be of particular interest for practicioners as it provides an overview over three popular techniques for solving convex mathematical programming problems of the SV type. After a brief overview over some general properties of convex optimization problems a couple of useful tricks is pointed out, which may greatly decrease the computational cost and simplify control of the algorithms. Next, interior point algorithms are presented, including the particular implications for SV regression, and also why it would be rather inefficient to use "off the shelf" optimizers in this case. The other benefit from presenting these techniques is that many of the considerations may be adapted to other optimization techniques as well. Unfortunately, for medium and large sized problems (more than 3000 samples) interior point methods become numerically quite expensive $O(m^3)$ and one has to seek cheaper methods to (approximately) solve the problem. Chunking and other working set algorithms are such techniques. After an overview of selection rules for the latter an adaptation of the Sequential Minimal Optimization (SMO) algorithm to regression is presented. Besides that SMO is extended to a more flexible setting of regularization properties, and a more realistic stopping rule, using the size of the feasibility gap, is stated. Pseudo code of the SMO regression algorithm is given.

After this rather lengthy discourse on supervised learning, some concepts of unsupervised learning are briefly mentioned in chapter 5. Two ways of gathering information are discussed in this context: a feature extracting approach leading to Kernel PCA and kernel feature analysis algorithms, where the goal is to find highly reliable feature extractors; secondly, a data descriptive approach where the goal is to minimize a quantization functional not unlike vector quantization or principal curves. The latter leads to a new algorithm, regularized principal manifolds (a hybrid between the generative topographic map and principal curves), which has nice practical and theoretical properties. Moreover the regularized quantization functional setting offers further extensions of this algorithm, to be explored in the future.

Applications of the methods described so far are presented in chapter 6. It is shown that SV machines yield excellent performance in time series prediction tasks. Moreover SV classification (a testbed for development of algorithms) is applied to data from particle physics. It is shown that even without much prior knowledge, SV machines yield state of the art performance, even in the high noise regime.

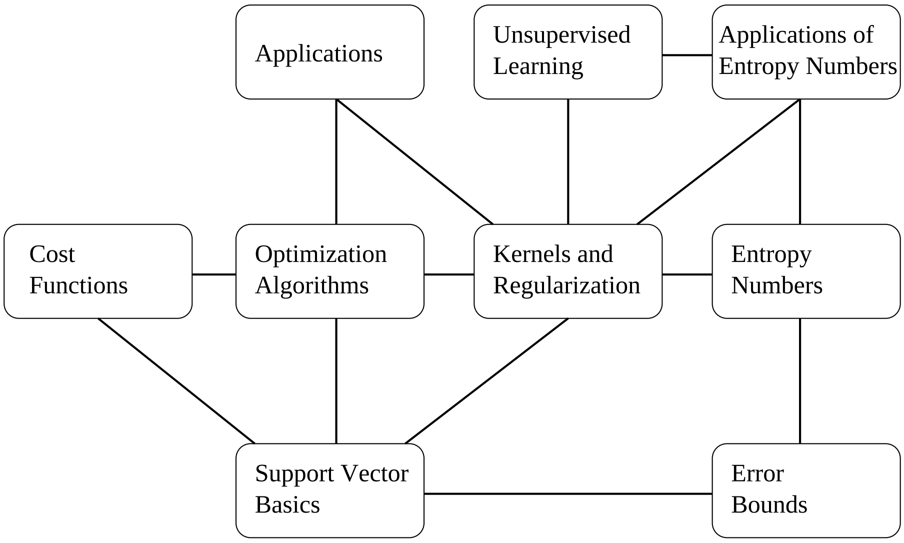

Figure 1 Connections and Dependencies between the Chapters.

Shortcuts

I tried to make each chapter as self contained as possible, such that combinations of the different techniques can be achieved even without having read all other chapters. However it appears reasonable to read chapter 1, the introduction of chapter 2 and section 3.1 before proceeding to more advanced topics. In particular, some knowledge about kernel expansions is required in nearly all formulations of the optimization problem. Figure 1 depicts the connections between the chapters.

1 Support Vector Basics

For the sake of convenience the first chapter gives a brief overview of the basic SV equations for the simplest case, namely linear regression with bounded loss. Many of the properties of this simplest of all SV regression algorithms can be found in the more complicated and elaborate settings as well. Hence it serves as a point of reference for further developments. Moreover many of the results presented here, directly carry over to classification. However, for the ease of presentation, results on the latter will be omitted. See the works of Cortes [1995], Vapnik [1995], Schölkopf [1997], Vapnik [1998] for details.

Roadmap

The first section takes an algorithmic approach to SV regression. This is done in favour of readers familiar with the practical details of the SV machines rather than their statistical implications. Readers with a learning theory background may want to swap the order and read section 1.2 first.

First the basic setting of finding the flattest linear function that approximates given data with ε precision is introduced. Next, this setting is transformed into a convex optimization problem. After a slight modification of the problem to suit for errors larger than ε, the Wolfe dual of the convex minimization problem is computed, resulting in a quadratic programming problem. Basic properties like the expansion of the weight vector w in terms of the training patterns, the Karush–Kuhn–Tucker conditions (for optimality), and a simple way to compute the constant threshold b are discused in this context.

Beyond the algorithmic viewpoint, the SV problem can also be considered as a special instance of the problem of risk minimization. This aspect illuminates the interaction between the class of models, the empirical risk functional and the expected risk. It is shown that regularization, i.e. restriction of the hypothesis class, is beneficial for two reasons — uniform convergence and well posedness of the optimization problem. Three different settings of regularization are discussed subsequently, namely the SV constraint, a convexity constraint used in linear programming, and a weight decay constraint in parameter space. Finally it is shown that minimization of the regularized risk functional is equivalent to minimizing the empirical risk for a *fixed* class of models, or minimization of the model complexity for a *fixed* amount of errors.

1.1 Introduction

1.1.1 The Basic Setting

Let $\mathcal{X}$ and $\mathcal{Y}$ be (subsets of) normed vector spaces, normally $\mathbb{R}^d$ and $\mathbb{R}$. Suppose one is given some "training" inputs

$$X := \{x_1, \ldots, x_m\} \subset \mathcal{X} \tag{1.1}$$

together with corresponding observations

$$Y := \{y_1, \ldots, y_m\} \subset \mathcal{Y} \tag{1.2}$$

(sometimes also briefly denoted as $Z := \{(x_1, y_1), \ldots, (x_m, y_m)\} \subset \mathcal{X} \times \mathcal{Y}$) drawn independently identically distributed (iid) according to some probability measure $dP(x, y)$.

The goal is to *learn* the data Z. This could be done, for instance, by seeking a function that has the least mean squared error. Another possible goal would be to find a function $f(x)$ that has at most ε deviation from the actually obtained targets y_i for all the training data Z, and at the same time, is as flat as possible. The present section is about methods for satisfying this goal. In other words, one does not bother about errors, provided they are smaller than ε, but will not accept any deviation larger than that.

An example is data that has been measured by some digital device (e.g. a voltmeter) with a rather limited number of digits. Roundoff errors, i.e. uniform noise, are exactly the type of additive noise, an ε–insensitive loss zone can account for, best. The data is given subject to a range $[y - \varepsilon, y + \varepsilon]$. Another example might be the case where Z is some exchange rate of a currency measured at subsequent days together with corresponding econometric indicators. There one wants to be sure not to lose more than ε money when dealing with the exchange rates.[1]

For the sake of simplicity assume that we are only interested in linear functions f (the extension to nonlinear functions will follow in chapter 3). For this purpose we have to make the additional assumption that $\mathcal{X}$ is an inner product space to define dot products. We assume

$$f(x) = \langle w, x \rangle + b \text{ with } w \in \mathcal{X}, b \in \mathcal{Y} \tag{1.3}$$

where $\langle \cdot, \cdot \rangle$ denotes the dot product in $\mathcal{X}$.

1. Of course one would not like to lose *any* money at all — this is what everyone would hope for — but that may not be possible. Morover one might tolerate bounded losses in the strong L_∞ metric rather than in the L_2 metric of least mean squares.

1.1.2 A Convex Optimization Problem

In order to solve the problem posed above, it is necessary to transform it into a constrained convex optimization problem. This is done as follows: choosing the flattest function means that one wants the factor w to be as "small" as possible. This may mean, for instance, that $\frac{1}{2}\|w\|^2$ is small (there exist other criteria for choosing "small" w, which will be futher discussed in examples 1.1, 1.2, and 1.3). This criterion is chosen in analogy to SV classification, where $\frac{2}{\|w\|}$ determines the width of the margin. At the same time the constraints have to be satisfied, namely that the error $|f(x_i) - y_i| \leq \varepsilon$ for all $i \in \{1, \ldots, m\}$. Formally this is equivalent to solving:

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|w\|^2 \\ \text{subject to} & \left\{ \begin{array}{lll} y_i - \langle w, x_i \rangle - b & \leq & \varepsilon \\ \langle w, x_i \rangle + b - y_i & \leq & \varepsilon \end{array} \right. \end{array} \tag{1.4}$$

The silent assumption in (1.4) is that such a function f actually exists, that approximates all pairs (x_i, y_i) with ε precision, or in other words, that the convex optimization problem is *feasible*. In some cases, however, this may not be the case, or one actually might want to allow for some errors, thus trading off errors by flatness of the estimate. Analogously to the soft margin loss function of Cortes and Vapnik [1995] one can introduce slack variables to cope with otherwise infeasible optimization problems. This leads to the formulation stated in [Vapnik, 1995].

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{m} (\xi_i + \xi_i^*) \\ \text{subject to} & \left\{ \begin{array}{lll} y_i - \langle w, x_i \rangle - b & \leq & \varepsilon + \xi_i \\ \langle w, x_i \rangle + b - y_i & \leq & \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* & \geq & 0 \end{array} \right. \end{array} \tag{1.5}$$

The constant $C > 0$ determines the trade off between the flatness of f and the amount up to which deviations larger than ε are tolerated. Eq. (1.5) corresponds to dealing with a so called ε–insensitive loss function $|\xi|_\varepsilon$ [Vapnik, 1995] described by

$$|\xi|_\varepsilon := \left\{ \begin{array}{ll} 0 & \text{if } |\xi| \leq \varepsilon \\ |\xi| - \varepsilon & \text{otherwise.} \end{array} \right. \tag{1.6}$$

Viewing (1.5) in this way also shows how to extend the basic SV problem to cost functions other than (1.6). This will be done in Sec. 2. Fig. 1.1 depicts the situation graphically:[2] only the points outside the shaded region contribute to the cost insofar, as the deviations are penalized in a linear fashion. It turns out that the optimization problem (1.5) can be solved more easily in its dual formulation.

2. Courtesy of Bernhard Schölkopf.

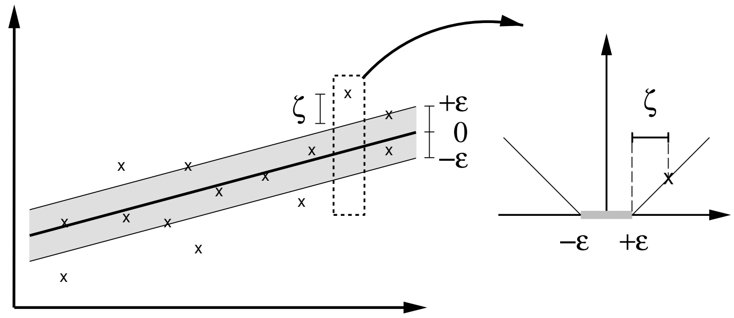

Figure 1.1 The soft margin loss setting corresponds for a linear SV machine.

Moreover, as will be seen in chapter 3, the dual formulation provides the key for extending SV machine to nonlinear functions. Therefore a standard dualization approach by means of Lagrange multipliers is taken (cf. Fletcher [1989]).

1.1.3 Dual Formulation and Quadratic Programming

The key idea is to construct a Lagrange function from both the objective function (it will be called the *primal* objective function in the rest of this article) and the corresponding constraints, by introducing a dual set of variables. It can be shown that this function has a saddle point with respect to the primal and dual variables at the optimal solution. For details see e.g. [Goldstein, 1986, Mangasarian, 1969, McCormick, 1983, Vanderbei, 1997]. Hence one proceeds as follows:

$$\begin{aligned} L := \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}(\xi_i + \xi_i^*) - \sum_{i=1}^{m}\alpha_i(\varepsilon + \xi_i - y_i + \langle w, x_i\rangle + b) \\ - \sum_{i=1}^{m}\alpha_i^*(\varepsilon + \xi_i^* + y_i - \langle w, x_i\rangle - b) - \sum_{i=1}^{m}(\eta_i\xi_i + \eta_i^*\xi_i^*) \end{aligned} \tag{1.7}$$

It is understood that the dual variables in (1.7) have to satisfy positivity constraints, i.e. $\alpha_i, \alpha_i^*, \eta_i, \eta_i^* \geq 0$. It follows from the saddle point condition (this is where the primal objective function is minimal, and the dual is maximized) that the partial derivatives of L with respect to the primal variables (w, b, ξ_i, ξ_i^*) have to vanish for optimality.

$$\partial_b L = \sum_{i=1}^{m}(\alpha_i^* - \alpha_i) = 0 \tag{1.8}$$

$$\partial_w L = w - \sum_{i=1}^{m}(\alpha_i - \alpha_i^*)x_i = 0 \tag{1.9}$$

$$\partial_{\xi_i^{(*)}} L = C - \alpha_i^{(*)} - \eta_i^{(*)} = 0 \quad (*) \text{ denotes variables with and without } * \tag{1.10}$$

Substituting (1.8) and (1.10) into (1.7) lets the terms in b and ξ vanish (they only contribute in a linear fashion anyway). Consequently also (1.10) can be transformed into $\alpha_i \in [0, C]$. Finally substitution of (1.9) into (1.7) shows that the expressions in w, the now quadratic terms in α_i, can be collected into one term, thus yielding the dual optimization problem.

$$\begin{aligned} &\text{maximize} && \begin{cases} -\frac{1}{2}\sum\limits_{i,j=1}^{m}(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)\langle x_i, x_j\rangle \\ -\varepsilon\sum\limits_{i=1}^{m}(\alpha_i+\alpha_i^*)+\sum\limits_{i=1}^{m}y_i(\alpha_i-\alpha_i^*) \end{cases} \\ &\text{subject to} && \begin{cases} \sum\limits_{i=1}^{m}(\alpha_i-\alpha_i^*) = 0 \\ \alpha_i, \alpha_i^* \in [0, C] \end{cases} \end{aligned} \tag{1.11}$$

In deriving (1.11) the dual variables η_i, η_i^* were already eliminated through condition (1.10), as these variables did not appear in the dual objective function anymore, but only were present in the dual feasibility conditions. Eq. (1.9) can be rewritten as follows:

$$w = \sum_{i=1}^{m}(\alpha_i-\alpha_i^*)x_i \text{ and therefore } f(x) = \sum_{i=1}^{m}(\alpha_i-\alpha_i^*)\langle x_i, x\rangle + b. \tag{1.12}$$

This is the well known Support Vector expansion, i.e. w can be completely described as a linear combination of the training patterns x_i. Note that this expansion is independent of both the dimensionality of the input space $\mathcal{X}$ and the sample size m. Moreover the complete algorithm can be described in terms of dot products between the data. Even when evaluating $f(x)$, by virtue of (1.12) one need not compute w explicitly. The latter is of course more computationally efficient in the linear setting, but these observations will come handy in the formulation of a nonlinear extension in chapter 3.

1.1.4 Computing b

So far the issue of computing b was neglected. It can be calculated by exploiting the so called Karush–Kuhn–Tucker (KKT) conditions [Karush, 1939, Kuhn and Tucker, 1951]. These state that at the optimal solution the product between dual variables and constraints has to vanish. In the SV case this means (cf. [Müller et al., 1997])

$$\begin{array}{rclcrcl} \alpha_i(\varepsilon+\xi_i-y_i+\langle w, x_i\rangle+b) & = & 0 & \text{and} & (C-\alpha_i)\xi_i & = & 0 \\ \alpha_i^*(\varepsilon+\xi_i^*+y_i-\langle w, x_i\rangle-b) & = & 0 & & (C-\alpha_i^*)\xi_i^* & = & 0 \end{array} \tag{1.13}$$

In other words

$$\begin{array}{rcll} b & = & y_i - \langle w, x_i\rangle - \varepsilon & \text{for } \alpha_i \in (0, C) \\ b & = & y_i - \langle w, x_i\rangle + \varepsilon & \text{for } \alpha_i^* \in (0, C) \end{array} \tag{1.14}$$

Another way of computing b will be discussed in the context of interior point optimization. There b turns out to be a by-product of the optimization process. See section 4.2.3 and 4.6.2 for more details on this topic.

1.2 A (more) Statistical View

Besides the practical aspect of SV regression, it will be helpful to consider the problem of regression and function estimation from a more statistical perspective. In the framework of risk minimization, as presented in [Vapnik, 1982], the problems regression, classification and density estimation reduce to the same fundamental class of problems.

1.2.1 The Problem of Risk Minimization

The general goal is to estimate a function

$$f : \mathcal{X} \to \mathcal{Y}, \qquad y = f(x) \tag{1.15}$$

that has minimal expected error on $\mathcal{X} \times \mathcal{Y}$. More precisely, deviations will be penalized according to a cost function $c(x, y, f(x))$. Without loss of generality assume

$$c(\cdot, \cdot, \cdot) \geq 0 \text{ and } c(\cdot, y, y) = 0 \tag{1.16}$$

In other words one will neither win by making extra good predictions nor will correct predictions incur any loss. Hence the problem to be solved can be written as follows:

$$\text{minimize } R[f] := \int_{\mathcal{X} \times \mathcal{Y}} c(x, y, f(x)) dP(x, y) \tag{1.17}$$

Eq. (1.17) is the fundamental problem, one tries to solve in statistical learning theory. Unfortunately it cannot be minimized exactly, as this would require knowledge of the probability measure $dP(x, y)$. Hence the problem is unsolvable in this formulation. The only way out is to approximate the latter by the empirical probability density function (pdf)

$$p_{\mathrm{emp}}(x, y) := \frac{1}{m} \sum_{i=1}^{m} \delta(x - x_i)\delta(y - y_i) \tag{1.18}$$

leading directly to the empirical risk functional

$$R_{\mathrm{emp}}[f] := \int_{\mathcal{X} \times \mathcal{Y}} c(x, y, f(x)) p_{\mathrm{emp}}(x, y) dx dy = \frac{1}{m} \sum_{i=1}^{m} c(x_i, y_i, f(x_i)). \tag{1.19}$$

One might think that finding a function f which minimizes $R_{\mathrm{emp}}[f]$ would ensure that $R[f]$ is at least close to minimal, too. However this is not true since f may be chosen arbitrarily.

Just imagine choosing f such that $f(x_i) = y_i$ for all $x_i \in X$ and $f(x) = 0$ for $x \in \mathcal{X} \setminus X$. It is very unlikely that such a function will be a minimizer of $R[f]$. Moreover one can show that the attempt to minimize $R_{\mathrm{emp}}[f]$ generally is an ill posed problem (except for very restrained model classes) [Tikhonov and Arsenin, 1977, Morozov, 1984, Vapnik, 1982, Bickel et al., 1994], i.e. the map from the training data X, Y to f may not be continuous, and can lead to a behaviour known as overfitting in the Neural Networks literature [Duda and Hart, 1974, Ripley, 1994, Bishop, 1995]. This raises the question under which conditions at least some statement can be made about $R[f]$.

1.2.2 Uniform Convergence

The situation is not as desperate as it may seem. First of all one can render the problem of minimizing $R_{\text{emp}}[f]$ well posed, i.e. make the map from X, Y to f continuous. This is done by restricting the set from which f is chosen to some *compact* set $\mathcal{F}$. This settles the algorithmic question. One can show by virtue of the operator inversion lemma [Tikhonov and Arsenin, 1977] that in this case the problem of empirical risk minimization becomes well posed. In particular one uses the following theorem.

Theorem 1.1 Operator Inversion Lemma, e.g. Riesz and Nagy [1955] Let X be a compact set and let the map $f : X \to Y$ be continuous. Then there exists an inverse map $f^{-1} : f(X) \to X$ that is also continous.

Considering $R_{\text{emp}}[f]$ as a continuous map from the space of admissible functions $\mathcal{F}$ into $\mathbb{R}$, one may construct an inverse map, and in particular obtain $\operatorname{argmin}_{f \in \mathcal{F}} R_{\text{emp}}[f]$, which is then well posed as required above. Section 1.2.3 will introduce basic concepts for constructing feasible sets $\mathcal{F}$.

Restricting $\mathcal{F}$ has yet another advantage. One can show [Vapnik and Chervonenkis, 1971] that if $\mathcal{F}$ is sufficiently well behaved, i.e. has finite covering number (a quantity introduced in chapter 7), the empirical risk $R_{\text{emp}}[f]$ will converge to $R[f]$ for increasing sample size m, i.e.

$$Pr\left\{\sup_{f \in \mathcal{F}} |R[f] - R_{\text{emp}}[f]| \geq \varepsilon\right\} \to 0 \quad \text{for} \quad m \to \infty \quad \text{and} \quad \varepsilon > 0. \tag{1.20}$$

The sup is taken over all functions in $\mathcal{F}$, as one would like to make a statement independently of the actual function that is chosen. Vapnik and Chervonenkis [1991] show that such a "worst case" statement is necessary and sufficient to give uniform convergence bounds (the "worst" function determines the behaviour of the whole class). Note that there is an additional subtlety which goes unnoticed quite often: effectively it is not the complexity of $\mathcal{F}$ but of the loss function induced class, i.e. $c(f(x), y, x)$ with $f \in \mathcal{F}$ that determines the overall complexity. This can be seen quite easily: assume $c = 0$ for all possible arguments. Here errorbounds are easily obtained independent of $\mathcal{F}$. However, under normal circumstances, $\mathcal{F}$ and the loss function induced class $c_{|\mathcal{F}}$ are related — see section 7.3.4 for more details.

Often function classes satisfying (1.20) are also called generalized Glivenko Cantelli classes. These issues will be discussed in greater detail in chapter 7. In particular, tools for deriving generalization bounds will be presented. For the moment assume that the classes $\mathcal{F}$ actually do satisfy (1.20). Finally note that usually (1.20) imposes an overly strict condition. For practical purposes all one needs is that the probablity $Pr\{\sup_{f\in\mathcal{F}}|R[f]-R_{\text{emp}}[f]|\geq\varepsilon\}\leq\eta$ for some $\eta<1$ rather than the fact that the probablilty converges to 0.

1.2.3 Regularized Risk Functional(s)

A large part of the elegance in SV machines and related kernel based methods comes from the fact that the problem is reduced to a linear setting, i.e. one tries to estimate linear functions in some space which need not be the input space at all. Most of the ideas on feature spaces developed in SV learning are based on a result in [Aizerman et al., 1964], however also note similar approaches by Nilsson [1965]. Hence in the following assume the case of (1.3).

As mentioned in section 1.2.2 one may have to restrict $\mathcal{F}$ even further. In practice this is done by imposing a convex penalty term on some quantity related to f. These functionals of f will be called $Q[f]$ subsequently. Two requirements will be imposed on $Q[f]$: it has to be convex and continuous (in order not to mess up the optimization problem) and should restrict the function class in such a way that uniform convergence bounds can be stated (this issue will be relegated to part II of the thesis). In this view one can rewrite (1.5) as follows:

$$\begin{array}{ll}\text{minimize} & Q[f]+C\frac{1}{m}\sum\limits_{i=1}^{m}(\xi_i+\xi_i^*)\\ \text{subject to} & \left\{\begin{array}{lll} y_i-\langle w,x_i\rangle-b & \leq & \varepsilon+\xi_i\\ \langle w,x_i\rangle+b-y_i & \leq & \varepsilon+\xi_i^*\\ \xi_i,\xi_i^* & \geq & 0.\end{array}\right.\end{array} \tag{1.21}$$

It is easy to check that the convex programming problem (1.21) can be rewritten as

$$\text{minimize}\quad R_{\text{reg}}[f]=\lambda Q[f]+\frac{1}{m}\sum_{i=1}^{m}|f(x_i)-y_i|_\varepsilon \tag{1.22}$$

with $\lambda=\frac{1}{C}\in\mathbb{R}^+$. This is the default notation when dealing with regularization networks. The following three choices of regularization terms are quite common in statistical learning theory.

Example 1.1 Support Vector Constraint
As already shown above in SV machines [Vapnik and Lerner, 1963, Boser et al., 1992, Smola and Schölkopf, 1998b] and regularization networks [Girosi et al., 1993] the regularization term is

$$Q[f]:=\frac{1}{2}\langle w,w\rangle=\frac{1}{2}\|w\|^2. \tag{1.23}$$

It leads to quadratic programming problems as shown above.

Example 1.2 Convexity Constraint
The convexity regularization is most used in Mathematical Programming [Mangasarian, 1964, Bradley et al., 1998, Bennett, 1999]. There, the assumption is that the weight vector w can be expanded as a linear combination of 'basis' patterns x_i,

usually the training set, such that $w = \sum_{i=1}^m \alpha_i x_i$. A bound on the absolute sum of the expansion coefficients is imposed, i.e.

$$Q[f] := \sum_{i=1}^{m} |\alpha_i| \tag{1.24}$$

Similar methods for regularization have been proposed in the context of sparse decompositions [Olshausen and Field, 1996] and basis pursuit denoising [Chen, 1995, Chen et al., 1995]. See also [Vapnik, 1998] for a similar approach.

Example 1.3 Weight Decay Constraint

Weight Decay is a well known principle from the Neural Networks literature (e.g. [Rumelhart et al., 1986, Hinton, 1989]). Again assume that $w = \sum_{i=1}^m \alpha_i x_i$ and regularization to be imposed indirectly via the coefficients α_i. The incremental learning rule to minimize $R_{\text{emp}}[f]$ is formulated as follows

$$\Delta\alpha_i = -\eta \left(\partial_{\alpha_i} R_{\text{emp}}[f] + \lambda\alpha_i\right). \tag{1.25}$$

Here λ, a small positive constant, is the so called *weight decay* parameter and η is the learning rate. Minimizing $R_{\text{emp}}[f]$ according to (1.25) is equivalent to a squared norm constraint on the weights, as can be seen by integrating (1.25) over α_i.[3] Hence one has

$$Q[f] := \frac{1}{2} \sum_{i=1}^{m} \alpha_i^2, \tag{1.26}$$

or in a more general version allowing for coupling between separate patterns

$$Q[f] := \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j D_{ij} \tag{1.27}$$

for some positive matrix D.

In all three examples presented above restricting f to $Q[f] \leq \Lambda$ for some $\Lambda > 0$ results in compact function classes $\mathcal{F}_\Lambda$ for which the problem of empirical risk minimization becomes well posed.[4] As will be shown in chapter 7, examples 1.1 and 1.2 lead to good means of capacity control, whereas example 1.3 may imply diverging bounds on the generalization error under some circumstances.

Both example 1.2 and 1.3 may give rise to the question why only indirect measures of regularization have been imposed. In fact, this is unnecessary when dealing with

---

3. Note that this imposes the weight decay constraint on the *expansion coefficients* rather than on the actual weight w. The approach presented here leads to regularizers of ridge regression type.

4. Actually this is not quite true — the parameter b was neglected. However one can show that an additional single free parameter does not spoil the overall setting. The calculations could also be carried out without this parameter, see corollary 7.18 for details. Yet, most papers use b for various, sometimes historic, reasons. Thus omitting b would lead to results, less familiar in notation.

finitely dimensional Euclidean spaces like $\mathbb{R}^d$, one could directly use $\|w\|_1$ instead of $\|\alpha\|_1$ to obtain similar results. In general, however, these norms may not even be defined (e.g. a $\|\cdot\|_1$ norm in feature space) and the Hilbert space where the dot products $\langle\cdot,\cdot\rangle$ are computed may only be indirectly accessible. This point will be clarified by the introduction of kernels in chapter 3.

1.2.4 Three Algorithms in one Picture

According to the considerations in the previous section the risk minimization problem for $f \in \mathcal{F}$ should be formulated as follows

$$\text{minimize} \quad R_{\text{emp}}[f] \quad \text{with} \quad Q[f] \leq \Lambda. \tag{1.28}$$

In other words, one minimizes $R_{\text{emp}}[f]$ while keeping the model complexity fixed by enforcing an upper bound on the measure of complexity (regularization term) $Q[f]$. This is what should be done when following the empirical risk minimization principle. In many cases, however, one tries to solve a different problem (with $\lambda > 0$)

$$\text{minimize} \quad R_{\text{reg}}[f] := R_{\text{emp}}[f] + \lambda Q[f]. \tag{1.29}$$

The advantage of this formulation is that it results in an optimization problem that can be solved more easily by numerical means. Therefore, unless stated otherwise, it will be the standard setting in this thesis. The function to be minimized $R_{\text{reg}}[f]$ is also called regularized risk functional. Finally one could also think of a third problem:

$$\text{minimize} \quad Q[f] \quad \text{with} \quad R_{\text{emp}}[f] \leq \Lambda'. \tag{1.30}$$

Algorithmically all three problems are (provided $Q[f]$ and $R_{\text{emp}}[f]$ are convex) by themself convex optimization problems which can be solved rather efficiently. The obvious question is whether, and under which conditions, (1.28), (1.29), and (1.30) are equivalent. Figure 1.2 depicts these three different choices of a learning algorithm.[5] As the functional $Q[f]$ defines a structure of nested subsets on $\mathcal{F}$ it is immediately obvious that the minimum of $R_{\text{emp}}[f]$ decreases monotonically with increasing $Q[f]$. One could, thus, parametrize the solutions with minimal empirical risk $R_{\text{emp}}[f]$ chosen from the set $\mathcal{F}_\Lambda := \{f \in \mathcal{F} | Q[f] \leq \Lambda\}$ by Λ.

The three algorithms in figure 1.2 are thus equivalent. One may specify the model class (1), i.e. $\mathcal{F}_\Lambda$, or the maximum training error (3), or determine the tradeoff between $R_{\text{emp}}[f]$ and $Q[f]$ beforehand (2). In the latter case λ is the negative slope of $R_{\text{emp}}[f]$ with respect to $Q[f]$ at the optimal solution: there the derivative of

5. A similar curve is known as the L–curve in statistics Hansen [1992] and is used to perform model selection. The difference in that approach is that instead of plotting $Q[f]$ vs. $R_{\text{emp}}[f]$ one plots the log thereof in order to avoid scaling dependencies. The optimal model is assumed to lie in the "kink" of the L that is formed by the correponding curve.

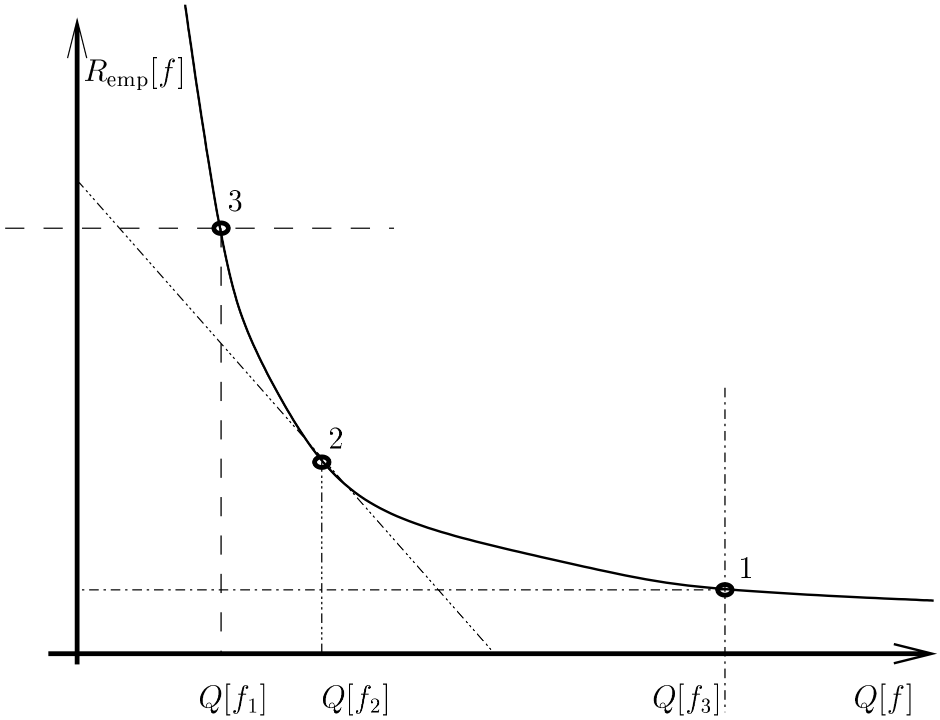

Figure 1.2 Three algorithms for risk minimization. (1) $Q[f]$ is fixed (see intercept with the Q axis) and $R_{\text{emp}}[f]$ is minimized subject to $Q[f] = \Lambda$. (2) The tradeoff λ, (the slope of the graph) between $Q[f]$ and $R_{\text{emp}}[f]$ is specified, thus the solution adapts to the actual learning problem. (3) The empirical risk $R_{\text{emp}}[f]$ is fixed (see intercept with the R axis) and $Q[f]$ is minimized subject to this constraint.

$R_{\text{reg}}[f]$ wrt. $Q[f]$ has to vanish, hence

$$\frac{\delta R_{\text{reg}}[f_{\text{opt}}]}{\delta Q[f_{\text{opt}}]} = \frac{\delta R_{\text{emp}}[f_{\text{opt}}]}{\delta Q[f_{\text{opt}}]} + \lambda \frac{\delta Q[f_{\text{opt}}]}{\delta Q[f_{\text{opt}}]} = \frac{\delta R_{\text{emp}}[f_{\text{opt}}]}{\delta Q[f_{\text{opt}}]} + \lambda = 0. \tag{1.31}$$

In other words, finding some f' that decreases $R_{\text{emp}}[f]$ by some δ would at least increase $Q[f]$ by δ/λ. This tangent condition is unique if the overall functional is strictly convex. Then there exists only one optimal solution for each given λ.

In the case where $R_{\text{emp}}[f]$ is not differentiable on a finite subset, one may still use the convexity of $R_{\text{emp}}[f]$ to obtain a similar statement. In that case, the left and right sided derivatives form an interval containing λ.

Finally, one might seek even further ways of combining $R_{\text{emp}}[f]$ and $Q[f]$ into one overall functional to be minimized, instead of a simple linear combination as in (1.29). There are two reasons why this might not be too desirable at all. First, nonlinear combinations might render minimization of the functional more difficult from an algorithmic point of view. Secondly, besides an initial guess of the trade off factor (= regularization constant) λ one will have to vary this parameter anyway, in order to find the particular hypothesis class (associated with λ) that promises best generalization error. Hence a new (and possibly more clever) way of minimizing a regularized risk functional might have virtually no influence on the final hypothesis that is chosen.

1.3 Summing Up

Basic concepts like regression with finite precision were introduced, and it was shown how this can be connected with regularization to obtain well posed problems. In particular, it was pointed out how regularization can be beneficial in two regards: rendering the optimization problem well posed and ensuring uniform convergence.

Moreover, techniques from optimization theory such as dualization and feasibility conditions were formulated, showing central properties of the SV solution such as the expansion of the solution in terms of the training patterns (independent of the dimensionality). Means for computing thresholds and for translating similar constrained optimization problems into problems with a trade off between model complexity and empirical error were pointed out.

What will be done in the following is an extension of the basic SV setting to cost functions other than the ε–insensitive loss, while keeping the property of being solvable by a convex optimization program. This is the topic of the next chapter.

Nonlinearities and a more thorough analysis of regularization are the issues of chapter 3. There a deeper connection between dot products and function spaces is pointed out and further methods to regularize functions are given.

Finally, the algorithmic details will be dealt with in chapter 4, presenting three different algorithms for solving the optimization equations.

Applications of these techniques to unsupervised learning and to real world problems conclude the first part. The second part of the thesis, in turn, will deal with the theoretical aspect of minimizing regularized risk functionals by assessing the capacity of such estimators.

2 Cost Functions

In order to construct algorithms for the problems (1.28), (1.29), and (1.30) one has to specify which cost function $c(x, y, f(x))$ to choose. This thesis will only consider **convex** cost functions. The practical reason being that in this case the problems mentioned above can be proven to have a unique minimum (cf. Fletcher [1989]). For many nonconvex cost functions the attempt to solve the corresponding risk minimization problems results in combinatorial optimization settings which are NP-hard as they exhibit many local minima.[1]

Roadmap

After a brief review of the connection between cost functions and noise models the first section focuses on deriving the optimization equations for arbitrary convex cost functions. This lays the foundations for decoupling the SV approach from the specific choice of a cost function made by Vapnik [1995]. Sparsity will be preserved as long as there exists an ε insensitive zone in the loss function. Robustness, conversely, as long as the maximum slope of the cost function c with respect to $f(x)$ is bounded.

For several cases of loss functions (linear, polynomial, piecewise polynomial, hard margin, etc.), explicit optimization problems are given in a directly implementable way. Experiments show that the increased degree of freedom in choosing a model may, in fact, contribute to improved generalization.

This, however, raises the question of what to do with this increased degree of freedom. In a restricted toy model of a SV machine with ε–insensitive loss, an answer can be given by means of asymptotical statistics. Astonishingly, despite the rather crude assumptions made in deriving the result, experimental findings confirm the linear scaling behaviour between external noise level and the (asymptotically) optimal degree of ε–insensitivity. Unfortunately these results hold only in the case when both *level and type* of the external noise are known.

A modification of the basic SV algorithm to allow for automatic tuning of the margin can be used to adjust ε in an (asymptotically) optimal fashion, provided, the *type* of the external noise is known. This result builds on the scaling behaviour between optimal ε and noise level. Parameters for polynomial noise are given.

1. It is possible to find reasonable convex proxies (cf. Wahba [1999] for the case of pattern recognition) for most real world cost functions, though.

Maximum likelihood takes the following approach to choosing cost functions.

Remark 2.1 Cost Functions and Maximum Likelihood
Under the assumption that the samples were generated by an underlying functional dependency plus additive noise $y_i = f_{\text{true}}(x_i) + \xi_i$ with density $p(\xi)$ the optimal cost function in a maximum likelihood sense would be

$$c(x, y, f(x)) = -\log p(y - f(x)). \tag{2.1}$$

Proof The likelihood[2] of an estimate

$$Z_f := \{(x_1, f(x_1)), \dots, (x_m, f(x_m))\} \tag{2.2}$$

under the assumption of additive noise which is independent of x and iid data is

$$\Pr\{Z_f|Z\} = \prod_{i=1}^{m} \Pr\{f(x_i)|(x_i, y_i)\} = \prod_{i=1}^{m} Pr\{f(x_i)|y_i\} = \prod_{i=1}^{m} p(y_i - f(x_i)). \tag{2.3}$$

Maximizing $Pr\{Z_f|Z\}$ is equivalent to minimizing $-\log Pr\{Z_f|Z\}$. Using (2.1) yields

$$-\log Pr\{Z_f|Z\} = \sum_{i=1}^{m} c(x_i, y_i, f(x_i)). \tag{2.4}$$

which proves the statement. ■

However, the cost function resulting from this reasoning might be nonconvex. In this case one would have to find a convex proxy in order to deal with the situation efficiently (i.e. to find an efficient implementation of the corresponding optimization problem).

Moreover, the situation of regression as such, i.e. without any knowledge of cost functions, is not properly defined from the viewpoint of structural *risk* minimization: *risk* can only be minimized if it can be quantified via a cost function (i.e. a penalty for deviations).

Finally, given a specific cost function from a real world problem, one should try to find as close a proxy to this cost function as possible, as it is the performance wrt. this particular cost function that matters ultimately.

2.1 Convex Programming Problems (again)

This section lays the foundations for general convex optimization problems. For this purpose one has to generalize the quadratic programming problems stated in (1.5) and (1.21). By replacing the slack variables ξ_i, ξ_i^* by $c_i(\xi_i)$ and $c_i^*(\xi_i^*)$ one obtains

2. We are not looking for the *probability* of f which would require the knowledge of a *prior* on f. Hence the present reasoning is in order. With slight abuse of notation we denote the likelihood by $\Pr\{\}$ just in the same way as the probability.

the following optimization problem.

$$\begin{array}{ll} \text{minimize} & Q[f] + C \sum_{i=1}^{m} (c_i(\xi_i) + c_i^*(\xi_i^*)) \\ \text{subject to} & \left\{ \begin{array}{lcl} y_i - \langle w, x_i \rangle - b & \leq & \varepsilon_i + \xi_i \\ \langle w, x_i \rangle + b - y_i & \leq & \varepsilon_i^* + \xi_i^* \\ \xi_i, \xi_i^* & \geq & 0 \end{array} \right. \end{array} \tag{2.5}$$

where

$$\begin{array}{lcl} c_i(\xi) & := & c(x_i, y_i, y_i + \varepsilon_i + \xi) \\ c_i^*(\xi) & := & c(x_i, y_i, y_i - \varepsilon_i^* - \xi) \end{array} \tag{2.6}$$

and $\varepsilon_i, \varepsilon_i^*$ are chosen such that $c(x_i, y_i, y_i + \xi) = 0$ for $\xi \in [-\varepsilon_i^*, \varepsilon_i]$. For squared loss this would imply[3] $\varepsilon_i^{(*)} = 0$, $c_i^{(*)}(\xi) = \frac{1}{2}\xi^2$.

2.1.1 Common Choices

The initial choice in Vapnik [1995], as already discussed in Sec. 1.1 is the ε–insensitive loss function, cf. (1.6), with

$$c(x, y, f(x)) = |y - f(x)|_\varepsilon \quad \text{hence} \quad c_i^{(*)}(\xi) = \xi \tag{2.7}$$

It follows directly from [Huber, 1981] that this cost function is robust in the class of uniform densities 'polluted' by an additional small arbitrary density. This is also backed by empirical results [Müller et al., 1997] for SV machines.

Other possible loss functions include Laplacian loss, leading to median type approximations, squared loss, being optimal for normal additive noise, several polynomial loss functions, and their robust counterparts where the maximum steepness has been limited. Table 2.1 summarizes common loss functions and the corresponding density models as defined by (2.1); figure 2.1 contains graphs of the corresponding functions.

Besides these standard choices one may enounter some more problem specific functions in practical situations.

Example 2.1 Locally Dependent ε_i

Assume the precision of the measurements depends on the location of the measurement. Hence it appears reasonable to take advantage of the latter. This can be done by defining

$$c(x, y, f(x)) := |y - f(x)|_{\varepsilon(x)}, \tag{2.8}$$

---

3. It is understood that expressions written with $(*)$ are assumed to hold for the corresponding quantities with and without the asterisk $*$. Moreover note that c_i is the ε–*clipped* version of the original cost function and should not be confused with the ε–insensitive loss itself. In the latter case $c_i(\xi_i) = \xi$.

	loss function $c(\xi)$	density model
ε–insensitive	$\|\xi\|_\epsilon$	$p(\xi) = \frac{1}{2(1+\varepsilon)} \exp(-\|\xi\|_\varepsilon)$
Laplacian	$\|\xi\|$	$p(\xi) = \frac{1}{2} \exp(-\|\xi\|)$
Gaussian	$\frac{1}{2}\xi^2$	$p(\xi) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{\xi^2}{2})$
Huber's robust loss	$\begin{cases} \frac{1}{2\sigma}(\xi)^2 & \text{if } \|\xi\| \leq \sigma \\ \|\xi\| - \frac{\sigma}{2} & \text{otherwise} \end{cases}$	$p(\xi) \propto \begin{cases} \exp(-\frac{\xi^2}{2\sigma}) & \text{if } \|\xi\| \leq \sigma \\ \exp(\frac{\sigma}{2} - \|\xi\|) & \text{otherwise} \end{cases}$
Polynomial	$\frac{1}{p}\|\xi\|^p$	$p(\xi) = \frac{p}{2\Gamma(1/p)} \exp(-\|\xi\|^p)$
Piecewise polynomial	$\begin{cases} \frac{1}{p\sigma^{p-1}}(\xi)^p & \text{if } \|\xi\| \leq \sigma \\ \|\xi\| - \sigma\frac{p-1}{p} & \text{otherwise} \end{cases}$	$p(\xi) \propto \begin{cases} \exp(-\frac{\xi^p}{p\sigma^{p-1}}) & \text{if } \|\xi\| \leq \sigma \\ \exp(\sigma\frac{p-1}{p} - \|\xi\|) & \text{otherwise} \end{cases}$

Table 2.1 Common loss functions and corresponding density models

and in particular $c(x_i, y_i, f(x_i)) := |y_i - f(x_i)|_{\varepsilon_i}$. Note that there is no direct need to define the function c on $\mathcal{X}$. In order to minimize the regularized risk, or solve one of the related problems it suffices to know c on the training set Z. Though, if c was allowed to vary too much, uniform convergence could not be satisfied even with a very simple class of models. On the other hand, if c was fixed before the analysis is performed, this problem vanishes.

One may think of even more complicated cost functions, even in very generic settings like pattern recognition.

Example 2.2 Soft Margin in Classification [Cortes and Vapnik, 1995]

There is no reason why c should only depend on the difference between y and $f(x)$. In the case of binary classification one has $y_i \in \{-1, 1\}$, and therefore

$$c(x, y, f(x)) := \max\{0, 1 - yf(x)\}. \tag{2.9}$$

2.1.2 Solving the Equations

For practical issues it is crucial to bring (2.5) into a form which is more suitable to solve. Unfortunately there is (so far) no general way of doing this for arbitrary regularization functionals $Q[f]$. In fact, one has to solve the equations separately for any of the three choices (1.23), (1.24), and (1.26).

For the sake of simplicity additionally assume c, as described by (2.6) to have at most two (for symmetry) discontinuities in the first derivative at $-\varepsilon_i^*$ and ε_i (with $\varepsilon_i^{(*)} \geq 0$) and to be zero in the interval between. This is not a major restriction, as all loss functions from table 2.1 belong to this class. For nonzero cost functions in the interval $[-\varepsilon_i^*, \varepsilon_i]$ use an additional pair of slack variables. At the expense of additional Lagrange multipliers in the dual formulation, additional discontinuities also can be taken care of.

Assume $Q[f] = \frac{1}{2}\|w\|^2$. Now again, by standard Lagrange multiplier techniques, exactly in the same manner as in the $|\cdot|_\varepsilon$ case, one can compute the dual

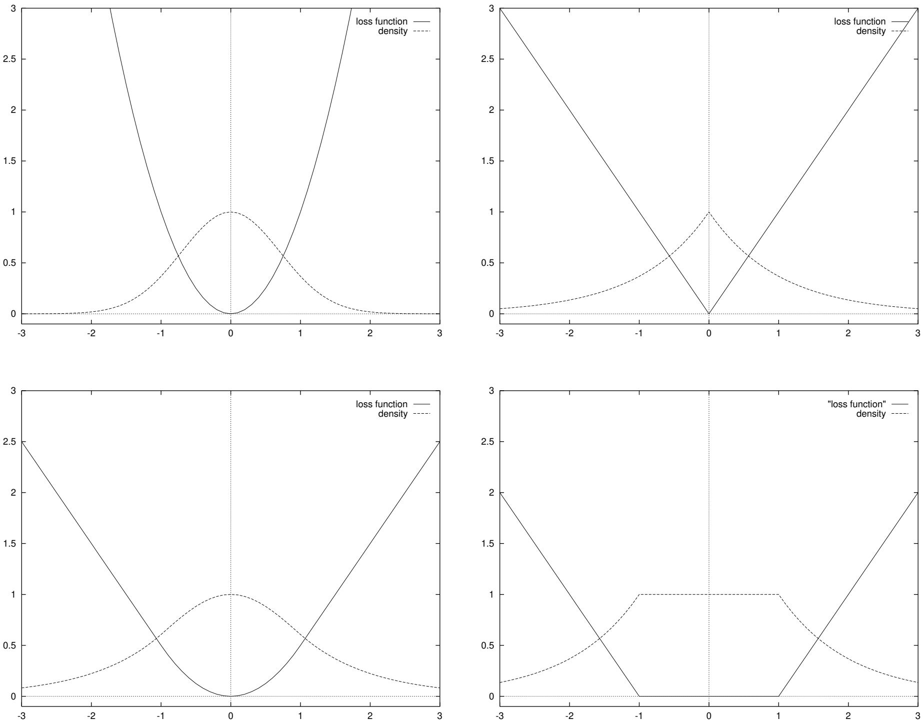

Figure 2.1 Graphs of loss functions and corresponding density models. upper left: Gaussian, upper right: Laplacian, lower left: Huber's robust, lower right: ε–insensitive

optimization problem of (2.5). Indices $_i$ and * are omitted where their meaning is obvious. This yields

$$\text{maximize} \begin{cases} -\frac{1}{2}\sum_{i,j=1}^{m}(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)\langle x_i, x_j\rangle \\ +\sum_{i=1}^{m}\left(y_i(\alpha_i-\alpha_i^*)-\varepsilon_i\alpha_i+\varepsilon_i^*\alpha_i^*+\frac{1}{\lambda}(T_i(\xi_i)+T_i^*(\xi_i^*))\right) \end{cases} \tag{2.10}$$

$$\text{where} \begin{cases} w & = & \sum_{i=1}^{m}(\alpha_i-\alpha_i^*)x_i \\ T_i^{(*)}(\xi) & := & c_i^{(*)}(\xi)-\xi\partial_\xi c_i^{(*)}(\xi) \end{cases} \tag{2.11}$$

$$\text{subject to} \begin{cases} \sum_{i=1}^{m}(\alpha_i-\alpha_i^*) & = & 0 \\ \alpha & \leq & \frac{1}{\lambda}\partial_\xi c(\xi) \\ \xi & = & \inf\{\xi \mid \frac{1}{\lambda}\partial_\xi c \geq \alpha\} \\ \alpha, \xi & \geq & 0 \end{cases} \tag{2.12}$$

The proof is straightforward, rather technical, and follows from [Smola and Schölkopf, 1998a]. Now consider the examples of table 2.1. The two following examples show, how (2.12) can be further simplified to bring it into a form that is practically useful.

Example 2.3 Vapnik's Loss Function
As already shown before in the ε–insensitive case, i.e. $c_i(\xi) = \xi$ one gets $T(\xi) = \xi - \xi \cdot 1 = 0$. Morover one can conclude from $\partial_\xi c_i(\xi) = 1$ that $\xi = \inf\{\xi \mid \frac{1}{\lambda} \geq \alpha\} = 0$ and hence $\alpha \in \left[0, \frac{1}{\lambda}\right]$.

Example 2.4 Piecewise Polynomial Loss
Here one has to distinguish two different cases: $\xi \leq \sigma$ and $\xi > \sigma$. The first case yields

$$T(\xi) = \frac{1}{p\sigma^{p-1}}\xi^p - \frac{1}{\sigma^{p-1}}\xi^p = -\tfrac{p-1}{p}\sigma^{1-p}\xi^p \tag{2.13}$$

and $\xi = \{\xi \mid \frac{1}{\lambda}\sigma^{1-p}\xi^{p-1} \geq \alpha\} = \sigma\lambda^{\frac{1}{p-1}}\alpha^{\frac{1}{p-1}}$. Therefore

$$T(\xi) = -\tfrac{p-1}{p}\sigma\lambda^{\frac{p}{p-1}}\alpha^{\frac{p}{p-1}}. \tag{2.14}$$

In the second case $(\xi \geq \sigma)$

$$T(\xi) = \xi - \sigma\tfrac{p-1}{p} - \xi = -\sigma\tfrac{p-1}{p} \tag{2.15}$$

and

$$\xi = \inf\left\{\xi \mid \tfrac{1}{\lambda} \geq \alpha\right\} = \sigma \text{ and hence } \alpha \in \left[0, \tfrac{1}{\lambda}\right]. \tag{2.16}$$

These two cases can be combined into one as

$$\alpha \in \left[0, \tfrac{1}{\lambda}\right] \text{ and } T(\alpha) = -\tfrac{p-1}{p}\sigma\lambda^{\frac{p}{p-1}}\alpha^{\frac{p}{p-1}}. \tag{2.17}$$

Table 2.2 contains a summary of the various conditions on α and formulas for $T(\alpha)$ for different cost functions. Note that the maximum slope of c_i determines the region of feasibility of α, i.e. $s := \sup_{\xi \in \mathbb{R}^+} \partial_\xi c_i(\xi) < \infty$ leads to compact intervals $[0, \frac{1}{\lambda}s]$ for α. This means that the influence of a single pattern is bounded, leading to robust estimators [Huber, 1972]. One can also observe experimentally that the performance of a SV machine depends significantly on the cost function [Müller et al., 1997, Smola et al., 1998b].

A cautionary remark is necessary regarding the use of cost functions other than the ε–insensitive one. Unless $\varepsilon > 0$ one will lose the advantage of a sparse decomposition. This may be acceptable in the case of few data, but will render the prediction step computationally quite expensive otherwise. Hence one will have to trade off a potential loss in prediction accuracy for faster predictions. Note, however, that a reduced set algorithm like in [Burges, 1996, Burges and Schölkopf, 1997, Osuna and Girosi, 1999, Schölkopf et al., 1998b] could be applied to alleviate this problem.

	ε	α	$\frac{1}{\lambda}T(\alpha)$
ε–insensitive	$\varepsilon \neq 0$	$\alpha \in [0, \frac{1}{\lambda}]$	$\frac{1}{\lambda}T(\alpha) = 0$
Laplacian	$\varepsilon = 0$	$\alpha \in [0, \frac{1}{\lambda}]$	$\frac{1}{\lambda}T(\alpha) = 0$
Gaussian	$\varepsilon = 0$	$\alpha \in [0, \infty)$	$\frac{1}{\lambda}T(\alpha) = -\frac{1}{2}\lambda\alpha^2$
Huber's robust loss	$\varepsilon = 0$	$\alpha \in [0, \frac{1}{\lambda}]$	$\frac{1}{\lambda}T(\alpha) = -\frac{1}{2}\sigma\lambda\alpha^2$
Polynomial	$\varepsilon = 0$	$\alpha \in [0, \infty)$	$\frac{1}{\lambda}T(\alpha) = -\frac{p-1}{p}\lambda^{\frac{1}{p-1}}\alpha^{\frac{p}{p-1}}$
Piecewise polynomial	$\varepsilon = 0$	$\alpha \in [0, \frac{1}{\lambda}]$	$\frac{1}{\lambda}T(\alpha) = -\frac{p-1}{p}\sigma\lambda^{\frac{1}{p-1}}\alpha^{\frac{p}{p-1}}$

Table 2.2 Terms of the convex optimization problem depending on the choice of the loss function.

2.1.3 Experiments

A close look at table 2.2 shows that one may distinguish two different cases. Laplacian, Gaussian, Huber's robust, and the ε–insensitive cost function lead to quadratic programming problems which can be solved by standard quadratic programming methods. Other cost functions lead to convex programming problems which are more difficult to minimize. Whilst the algorithmic issue will be considered in chapter 4, one also has to answer the question whether additional generalization performance can be gained by further extending the choice of available cost functions.

One can see that this is the case, indeed. For the experimental setup artificial data generated by an additive noise model with $p(\xi) \propto \exp(-|\xi|^{1.5})$ and standard deviation 0.5 was used. Clearly in this case, neither an L^1 nor an L^2 cost function would be optimal in a maximum likelihood sense. The generalization error is measured in terms of the $\|.\|_{1.5}$ norm.

To cope with potential nonlinearities in the training data, Gaussian rbf–kernels[4] are used with $\sigma^2 = 0.5$; the analyzed cost functions are of the piecewise polynomial type with $\epsilon = 0$, and polynomial degree $p \in [1, 2]$. The functional dependency is $y = \text{sinc}(x)$, $x \in \mathbb{R}^1$ and the data equidistantly distributed over the interval $[-4\pi, 4\pi]$ with sample size 50. In order to obtain reliable results the generalization error was averaged over 100 runs where the best regularization parameter $\lambda = 1/C$ was selected by crossvalidation for each run independently. The exponent of the cost function is varied to measure how well each exponent performs. Figure 2.2 exhibits a minimum close to $p = 1.5$, which was the exponent of the exponential distribution that generated the noise. Observe the improvement with respect to L^1 ($p = 1$) and Huber's ($p = 2$) robust loss, i.e. the values at the boundary of the interval.

This experimentally shows that the additional freedom in adapting the noise model can improve generalization performance in SV regression. The large error

4. See section 3 for details.

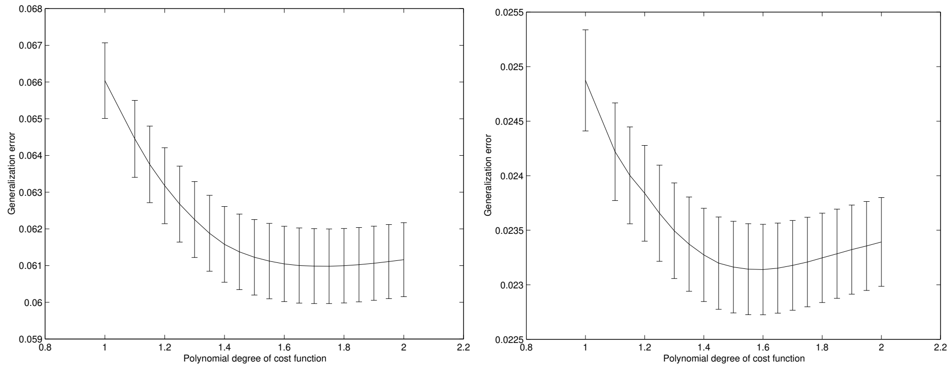

Figure 2.2 Generalization error depending on the noise model. Left: sample size $m = 100$, right: sample size $m = 200$; The noise had unit variance and zero mean, distributed $\propto \exp(-|\xi|^{1.5})$.

bars are due to the overall variation of the generalization error in different runs. This in turn is caused by the rather small sample size ($m = 100, 200$) and signal to noise ratio (0.65). Measuring the *relative* variance (i.e. for each fixed test and training set separately) would have led to much smaller error bars.

2.2 Asymptotically Optimal Choice of ε

The above reasoning still leaves one with the question which cost function to choose, and even for a fixed cost function, say the ε–insensitive, it is not clear, how to adjust the parameter ε. One might think that setting $\varepsilon = 0$ would lead to best results as it "pays most attention" to the data. However, in general this is not true.

For the sake of simplicity let us consider a toy model of a SV machine — the estimation of a location parameter setting, i.e. the attempt to estimate one single parameter by using the ε–insensitive cost function.

This section follows largely [Smola et al., 1998a]. One can show that there is a nontrivial choice of ε for which the statistical efficiency of estimating a location parameter using the ε–insensitive loss function is maximized and that the optimal ε is proportional to the variance of the random variable (here additive noise) under consideration.

2.2.1 Statistical Preliminaries

For this purpose one has to introduce some statistical notations. Denote by $\hat{\alpha}(Z)$ an estimator of the parameters α (not to be confused with the Lagrange multipliers of the previous section) based on the sample Z and let Z (in this section) be drawn according to some probability measure $dP(Z, \alpha)$ (also parametrized by α). Finally denote by $E[\xi]_\alpha$ the expectation of the random variable ξ with respect to $p(Z, \alpha)$.

Now one can define an unbiased estimator $\hat{\alpha}(Z)$ by requiring

$$E\left[\hat{\alpha}(Z)\right]_{\bar{\alpha}} = \bar{\alpha}. \tag{2.18}$$

Moreover one may introduce the Fisher information matrix I denoting

$$I_{ij} := E\left[\partial_{\alpha_i} \ln p(Z, \bar{\alpha}) \cdot \partial_{\alpha_j} \ln p(Z, \bar{\alpha})\right]_{\bar{\alpha}}, \tag{2.19}$$

and the covariance matrix B of the estimator $\hat{\alpha}$ by

$$B_{ij} := E\left[(\hat{\alpha}_i - E[\hat{\alpha}_i]_{\bar{\alpha}})(\hat{\alpha}_j - E[\hat{\alpha}_j]_{\bar{\alpha}})\right]_{\bar{\alpha}}. \tag{2.20}$$

The Cramér–Rao inequality [Rao, 1973] states that $\det IB \geq 1$ for all possible estimators $\hat{\alpha}$. This allows us to define the statistical efficiency e of an estimator as

$$e := \frac{1}{\det IB}. \tag{2.21}$$

Comparing the quality of unbiased estimators in this context can be reduced to comparing their statistical efficiencies. For a special class where $\hat{\alpha}$ is defined by

$$\hat{\alpha}(Z) := \operatorname*{argmin}_{\alpha} d(Z, \alpha) \tag{2.22}$$

and d is a two times differentiable function in α one can show [Murata et al., 1994, Lemma 3] that asymptotically $B = Q^{-1} G Q^{-1}$ with

$$G_{ij} := \mathrm{Cov}_{\bar{\alpha}}\left[\partial_{\alpha_i} d\left(Z, \bar{\alpha}\right), \partial_{\alpha_j} d\left(Z, \bar{\alpha}\right)\right] \text{ and } Q_{ij} := E\left[\partial^2_{\alpha_i \alpha_j} d(Z, \bar{\alpha})\right]_{\bar{\alpha}} \tag{2.23}$$

and therefore $e = (\det Q)^2/(\det IG)$. This allows one to formulate the following proposition.

Proposition 2.2 Asymptotically Optimal Choice of ε
Denote by ϕ a noise model and ξ_i random variables drawn iid from ϕ. In the estimation of a location parameter case with the ε–insensitive loss as cost function, the estimator achieves its maximal efficiency for ε chosen as

$$\varepsilon_{\mathrm{opt}} = \operatorname*{argmin}_{\varepsilon} \frac{(\phi(-\varepsilon) + \phi(\varepsilon))^2}{\left(1 - \int_{-\varepsilon}^{\varepsilon} \phi(\alpha) d\alpha\right) E\left[\left(\partial_\alpha \ln \phi(\alpha)\right)^2\right]_\phi}. \tag{2.24}$$

Proof For estimation of a location parameter one has to deal with a one–parametrical model and estimate the mean of a distribution. Denote by $p_\varepsilon(\xi)$ the noise model given by the ε–insensitive loss function,

$$p_\varepsilon(\xi) = c_0 \exp(-|\xi|_\varepsilon) = \frac{1}{2(1+\varepsilon)} \begin{cases} 1 & \text{if } |\xi| \leq \varepsilon \\ \exp(\varepsilon - |\xi|) & \text{otherwise} \end{cases}. \tag{2.25}$$

Without loss of generality assume the location parameter to be 0 or formally

$E[Z_i]_\phi = 0$.[5] Then the maximum likelihood estimator $\hat{\alpha}(Z)$ is given by setting

$$d(Z,\alpha) = -\frac{1}{m}\sum_{i=1}^{m} \ln p_\varepsilon(Z_i - \alpha). \tag{2.26}$$

Substituting (2.25) and (2.26) into the definitions of I, G, and Q yields

$$I = E\left[(\partial_\alpha \ln \phi(\alpha))^2\right]_\phi \tag{2.27}$$

$$G = E\left[(\partial_\alpha \ln p_\varepsilon(\alpha))^2\right]_\phi = 1 - \int_{-\varepsilon}^{\varepsilon} \phi(\alpha) d\alpha \tag{2.28}$$

$$Q = E\left[\partial_\alpha^2 \ln p_\varepsilon(\alpha)\right]_\phi = \phi(-\varepsilon) + \phi(\varepsilon) \tag{2.29}$$

In (2.28) we exploited that ϕ is symmetric and hence the $E[\partial_\alpha \ln \phi(\alpha)]$ term cancels out. Exploiting $e = Q^2/(GI)$ and noting that optimal efficiency is obtained (by definition) for maximal $e(\varepsilon)$ proves the statement. ■

2.2.2 Scaling Properties for different Noise Models

To make things more explicit, consider different choices of ϕ and the corresponding values of e.

Example 2.5 Gaussian Noise
Set $\phi(\xi) = 1/\sqrt{2\pi\sigma^2} \exp\left(-\xi^2/\left(2\sigma^2\right)\right)$. Then $I = \sigma^{-2}, G = 1 - \operatorname{erf}\left(\frac{\varepsilon}{\sqrt{2}\sigma}\right)$ and therefore

$$\frac{1}{e} = \frac{\det GI}{\det Q^2} = 2\pi \exp(\varepsilon^2/\sigma^2)\left(1 - \operatorname{erf}\left(\frac{\varepsilon}{\sqrt{2}\sigma}\right)\right). \tag{2.30}$$

The maximum of e is obtained for $\varepsilon/\sigma = 0.6120$ and therefore one has a linear dependency between ε and the noise level.

A sanity check is to compute the optimal ε for Laplacian noise.

Example 2.6 Laplacian Noise
Set $\phi(\xi) = 1/(2\sigma)\exp(-|\xi|/\sigma)$. Then $I = \sigma^{-2}, G = \exp(-\varepsilon/\sigma)$ and therefore $\frac{1}{e} = \exp(\varepsilon/\sigma)$. Here the maximum of e is achieved for $\varepsilon/\sigma = 0$, i.e. the case where p_ε degenerates to the L^1 loss, exactly matching the Laplacian noise. In this case the estimator is asymptotically efficient ($e = 1$).

Finally consider the general case. For the sake of simplicity assume a symmetric noise model. The only thing one has to show is that the efficiency is not optimal for $\varepsilon/\sigma = 0$ but depends on the coefficient $\tau := \varepsilon/\sigma$.

---

5. One always could redefine the problem for a nonzero location parameter by shifting all variables by the corresponding amount. We use the assumption for being able to compute the second order moment more easily.

Example 2.7 Arbitrary Symmetric Noise
One can derive general conditions for a linear scaling behaviour of ϵ. Assume $\phi(\xi)$ to be a symmetric density with unit variance. Hence $1/\sigma\phi(\xi/\sigma)$ has standard deviation σ. Now rewrite G and Q in terms of τ as

$$G = 2\int_{\epsilon}^{\infty} \frac{1}{\sigma}\phi(\xi/\sigma)d\xi = 2\int_{\tau}^{\infty} \phi(\xi)d\xi, \tag{2.31}$$

$I_\sigma = \sigma^{-2}$ and $Q = \frac{2}{\sigma}\phi(\tau)$. This leads to

$$e = \frac{\det Q^2}{\det GI} = \frac{4\sigma^{-2}\phi(\tau)}{\sigma^{-2}2\int_{\tau}^{\infty}\phi(\xi)d\xi} = \frac{2\phi^2(\tau)}{\int_{\tau}^{\infty}\phi(\xi)d\xi}. \tag{2.32}$$

What remains is to check that e does not have a maximum for $\tau = 0$. Computing the derivative of e with respect to τ for $\tau = 0$ yields the sufficient condition $\partial_\xi\phi(0) + \phi^2(0) > 0$. For instance any density with $\partial_\xi\phi(0) = 0$ and $\phi(0) \neq 0$ satisfies this property.

It is not directly possible to carry over the conclusions to the SV case due to two assumptions that may not be satisfied. Neither is one dealing with the asymptotic case nor is a SV machine a model of a single parameter. Instead, one has finite a sample size and estimates a function. Experiments illustrate, however, that the conclusions obtained from this simplified situation are still approximately valid in the more complex SV case.

2.2.3 Experiments

Consider a toy example, namely $f(x) = 2\mathrm{sinc}\,(2\pi(x-5))$ on $[0,10]$. 50 and 100 datapoints x_i were drawn iid from a uniform distribution on $[0,10]$, and generated the sample via $y_i = f(x_i) + \xi_i$. Here ξ_i was a Gaussian random variable with zero mean and variance σ^2. We used a Gaussian rbf kernel (see section 3) of width $\frac{1}{4}$.

The purpose of these experiments was to exhibit the dependency on ε. Hence 'model–selection' for the regularization parameter is carried out in such a way to always choose the value which led to the smallest L^1 error (the same reasoning works for the L_2 error, too) on the test set. Thereby it was possible to exclude side effects of possible model selection criteria. For statistical reliability the results were averaged over 100 trials.

The left graph in figure 2.3 shows the behaviour of the inverse statistical efficiency of estimators of a location parameter with the ε–insensitive loss function. $1/e$ clearly exhibits a minimum at $\tau = \epsilon/\sigma = 0.612$. The right graph shows the $L_1([0,10])$ error of the optimal (wrt. C) estimate for different values of ϵ for fixed noise $\sigma = 0.4$ and sample sizes $m = 50$. It shows a minimum for $\epsilon = 0.265$, which is not too far away from the theoretically predicted minimum of $0.612 \cdot 0.4 = 0.245$. Moreover observe the qualitatively similar behaviour of both graphs.

Also note the approximately linear dependency in figure 2.4. Again, ϵ was chosen such that optimal generalization performance was achieved. It is a striking example for the quality of the calculations. It shows that over a wide range of different

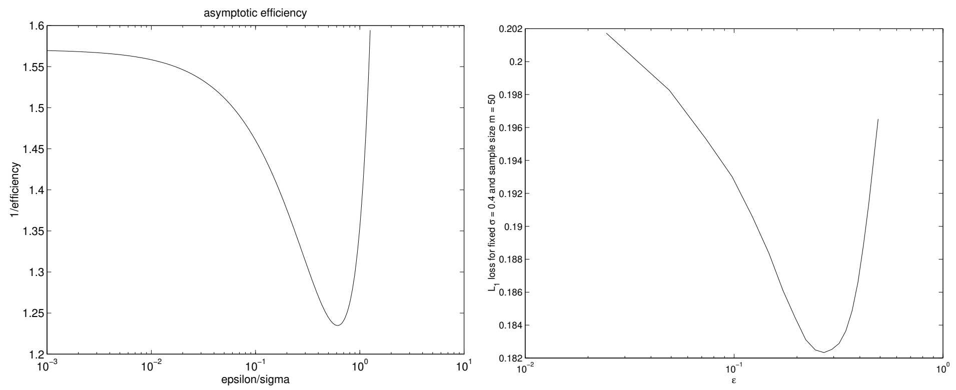

Figure 2.3 Left: Inverse asymptotic efficiency (2.30) for an ϵ–insensitive model and data with Gaussian noise. Right: L^1 loss for $m = 50$ and fixed noise level $\sigma = 0.4$ for different values of ϵ, averaged over 100 trials.

sample sizes and noise levels there exists a linear scaling behaviour which matches closely the $\epsilon = 0.612\sigma$ prediction from theory, even though the assumptions are not exactly satisfied.

Figure 2.5 shows this dependency again, but this time already rescaled for $\tau = \epsilon/(0.612\sigma)$ and relative to the best performance of the estimator for a given noise level. Observe that the contour lines (each line corresponds to a deterioration of 0.5%) are rather well centered around the theoretically optimal quotient $\epsilon/(0.612\sigma) = 1$, corresponding to the minimum of the relative error with respect to the optimal predictor for a given noise .

These findings are corroborated by results of Solla and Levin [1992], where it is shown that for linear Boltzmann machines the best performance is achieved when the "internal noise" matches the external noise.

2.3 Adaptive Noise Models

The considerations about asymptotically optimal choice still leave one with the problem of having to guess a parameter, now the variance of the additive noise. Thus, this result by itself, is not particularly useful in practice. What one wants is an automatic method to determine ϵ. This can be found as follows.

It is convenient to remember that asymptotically for fixed ϵ the number of Support Vectors converges to the number of samples deviating from the *true* function more than ϵ (this is true under the assumption that one is dealing with a uniform convergent estimator, and SV regression is assumed to be one). Now, if one could construct a modified algorithm that could be proven to have an asymptotically fixed (previously chosen) fraction of SVs one would only have to adjust this fraction such that it yields the corresponding setting of ϵ. This is what will be done in the following, but first some details of the algorithm ...

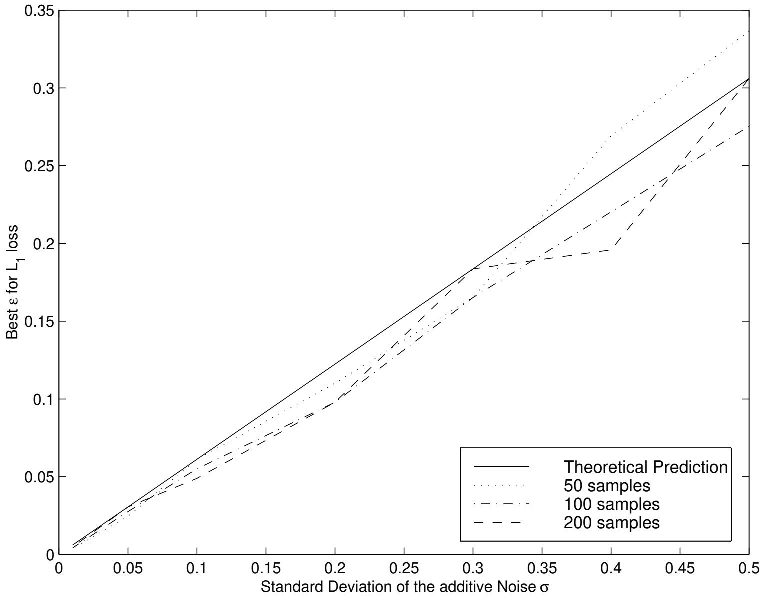

Figure 2.4 Experimentally optimal choice of ε and τ for different levels of Gaussian noise vs. their theoretically predicted best value ($\ell = 100$).

2.3.1 The Basic Idea

Assume the case of ε–insensitive loss. In this section denote by $R_{\text{emp}}[f, \varepsilon]$ the corresponding empirical risk functional. Schölkopf et al. [1998a], Schölkopf et al. [1999b] modify the regularized risk functional $R_{\text{reg}}[f]$ to automatically adapt ε.

$$R_{\text{reg}}[f, \nu] := R_{\text{emp}}[f, \varepsilon] + \nu\varepsilon + \lambda Q[f] \tag{2.33}$$

Depending on the choice of the regularizer $Q[f]$ one obtains Boosting type algorithms, Linear Programming settings, or for $Q[f] = \frac{1}{2}\|w\|^2$ a new SV regression algorithm. Rewriting (2.33) as a constrained convex optimization problem yields

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\|w\|^2 + C\left(\nu\varepsilon + \frac{1}{m}\sum_{i=1}^{m}(\xi_i + \xi_i^*)\right) \\ \text{subject to} & \left\{\begin{array}{lcl} y_i - \langle w, x_i\rangle - b & \leq & \varepsilon + \xi_i \\ \langle w, x_i\rangle + b - y_i & \leq & \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* & \geq & 0 \end{array}\right. \end{array} \tag{2.34}$$

Now one has to compute the Wolfe dual (for details see Schölkopf et al. [1998a]) just as in the standard SV regression setting. As ε is a variable of the optimization problem this time and contributes only linearly it drops out in the dual. An equality

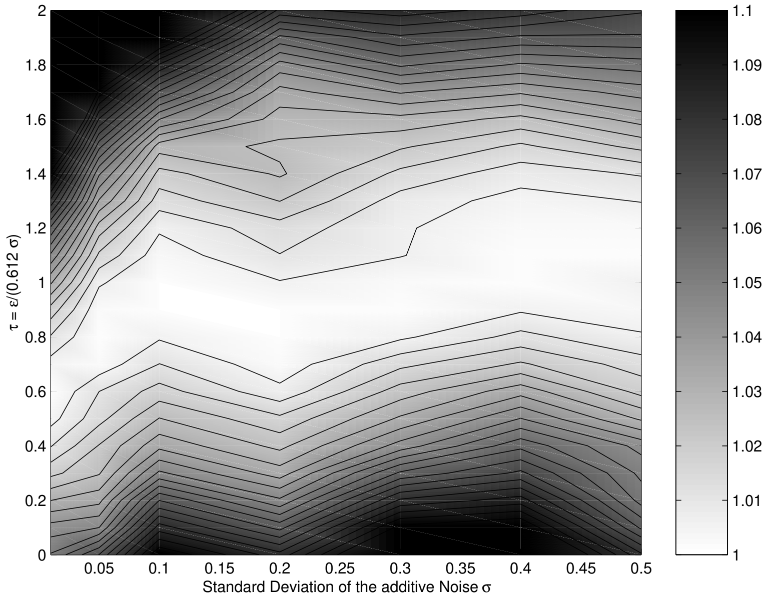

Figure 2.5 Relative L_1 error for different values of ε and different levels of Gaussian noise. For each fixed noise level the performance was normalized to the best performing setting (ε), i.e. the minimum of the error valley is fixed to 1 independently of the noise level. The contour lines indicate the shape of the valley — each line corresponds to a deterioration of 0.5% of the performance of the estimator.

constraint takes its place and we obatin

$$\begin{array}{ll} \text{maximize} & -\frac{1}{2}\sum\limits_{i,j=1}^{m}(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)\langle x_i,x_j\rangle+\sum\limits_{i=1}^{m}y_i(\alpha_i-\alpha_i^*) \\ \text{subject to} & \left\{\begin{array}{lcl}\sum\limits_{i=1}^{m}(\alpha_i-\alpha_i^*) & = & 0 \\ \sum\limits_{i=1}^{m}(\alpha_i+\alpha_i^*) & = & C\nu \\ \alpha_i,\alpha_i^* & \in & [0,\frac{C}{m}]\end{array}\right. \end{array} \tag{2.35}$$

with the standard expansion of f in terms of the Lagrange multipliers α_i, α_i^*, and b. The nice property of this setting is that it still can be solved by a standard optimization algorithm, just with an additional constraint. Just as b also ε can be computed by making direct use of the KKT conditions just as in section 1.1.4. However (also like b) it is more convenient to exploit primal–dual properties directly and obtain this parameter as a by product (i.e. the dual variable of the second equality constraint) of the optimization algorithm.

Further modifications to (non)parametric models of cost functions instead of a constant margin (i.e. heteroscedastic noise) exist. Moreover also the classification problem can be rewritten in a similar way to specify *asymptotically* the number of SVs beforehand.[6] A thorough analysis of these issues can be found in Schölkopf et al. [1998b].

2.3.2 Properties of ν SV Regression

The following proposition captures some fundamental properties of ν–regression (this name is chosen to distinguish this algorithm from the standard ε regression approach).

Proposition 2.3 Schölkopf, Bartlett, Smola, and Williamson [1998a]
Assume $\varepsilon > 0$. The following statements hold:

1. ν is an upper bound on the fraction of errors, i.e. number of samples lying outside the ε tube.
2. ν is a lower bound on the fraction of SVs.
3. Suppose the data Z were generated iid from a distribution $P(x, y) = P(x)P(y|x)$ with $P(y|x)$ continuous. With probability 1, asymptotically ν equals both the fraction of SVs and the fraction of errors.

The first two claims follow immediately from the dual optimization problem. The third claim can be proven via a uniform convergence argument. Hence ν SV regression is an algorithm that can be used to (asymptotically) specify the number of SVs beforehand.

2.3.3 Asymptotically Optimal Choice of ν

Combining the previous results leads immediately to the following proposition which solves asymptotically the choice of an optimal ϵ for given classes of noise models. This allows to compute ν “once and for ever”, given that the noise model is known.

Remark 2.4 Optimal Choice of ν
Denote by $\mathfrak{p}$ a probability density with unit variance, and by $\mathfrak{P}$ a famliy of noise models generated from $\mathfrak{p}$ by

$$\mathfrak{P} := \left\{ p \left| p = \tfrac{1}{\sigma}\mathfrak{p}\left(\tfrac{y}{\sigma}\right) \right. \right\}. \tag{2.36}$$

Moreover assume that the data were generated iid from a distribution $p(x, y) = p(x)p(y - f(x))$ with $p(y - f(x))$ continuous, i.e. generated by an underlying

6. Specifying the number of SVs beforehand may appear strange, especially in separable cases. However, by increasing the regularization parameter λ, i.e. decreasing C, also “separable” problems become “nonseparable.” This is not an unreasonable approach as when using e.g. Gaussian rbf–kernels all problems would be separable for a certain regularization strength on. See chapter 3 for details.

functional dependency f, corrupted by additive noise. Then under the assumption of uniform convergence, the asymptotically optimal adaptation parameter ν is

$$\nu = 1 - \int_{-\varepsilon}^{\varepsilon} \mathfrak{p}(t)dt \text{ where } \varepsilon := \operatorname*{argmin}_{\tau} \frac{1}{(\mathfrak{p}(-\tau) + \mathfrak{p}(\tau))^2}\left(1 - \int_{-\tau}^{\tau} \mathfrak{p}(t)dt\right) \tag{2.37}$$

Proof For a uniformly convergent algorithm the estimate $\hat{f}$ converges to the underlying functional dependency f. Assume that SV regression satisfies this property.[7] Then the probability of a deviation larger than ε can be computed as

$$Pr\left\{|y - \hat{f}(x)| > \varepsilon\right\} = Pr\left\{|y - f(x)| > \varepsilon\right\} \tag{2.38}$$

$$= \int_{\mathcal{X}\times\{\mathbb{R}/[-\varepsilon,\varepsilon]\}} p(x)p(\xi)dxd\xi \tag{2.39}$$

$$= 1 - \int_{-\varepsilon}^{\varepsilon} p(\xi)d\xi. \tag{2.40}$$

This is also the fraction of samples that will (asymptotically) become SVs. Therefore an algorithm generating a fraction $\nu = 1 - \int_{-\varepsilon}^{\varepsilon} p(\xi)d\xi$ SVs will correspond to an algorithm with a tube of size ε. The consequence is that, given a noise model $p(\xi)$, one can compute the optimal ε for it, and then, by using (2.38), compute the corresponding optimal value ν. This leads to an algorithm with asymptotically optimal choice of ν.

Next one exploits the linear scaling behaviour between the standard deviation σ of a distribution p and the optimal ε, established in section 2.2 for the estimation of a location parameter context. This means that one has to consider only distributions of unit variance, say, $\mathfrak{p}$, to compute an optimal value of ν that holds for the whole class of distributions $\mathfrak{P}$ generated via (2.36).

The last step is to use $e\left(\frac{\varepsilon}{\sigma}\right) = \frac{Q^2}{GI}$ where the Fisher information is independent of ε and thus by (2.28) and (2.29) (the det was dropped as Q, G, I are scalar quantities in the present case)

$$\frac{1}{e(\varepsilon)} \propto \frac{G}{Q^2} = \frac{1}{(\mathfrak{p}(-\varepsilon) + \mathfrak{p}(\varepsilon))^2}\left(1 - \int_{-\varepsilon}^{\varepsilon} \mathfrak{p}(t)dt\right) \tag{2.41}$$

The minimum of (2.41) yields the optimal choice of ε, which allows computation of the corresponding ν and thus proves the statement. ■

Note that this reasoning only holds in the *asymptotical* case, with the *approximation* of a SVM by the estimation of a location parameter. Thirdly we assumed the SVM to be *uniformly convergent* to the true model. Hence the "proof" is by no means rigorous. Still the basic properties hold.

---

7. Ways to show this are pointed out in part II of this thesis.

2.3.4 Polynomial Noise

Arbitrary polynomial noise models ($\propto e^{-|\xi|^p}$) with unit variance can be written as

$$\mathfrak{p}(\xi) = \frac{1}{2}\sqrt{\frac{\Gamma\left(\frac{3}{p}\right)}{\Gamma\left(\frac{1}{p}\right)}}\frac{p}{\Gamma\left(\frac{1}{p}\right)}\exp\left(-\left(\sqrt{\frac{\Gamma\left(\frac{3}{p}\right)}{\Gamma\left(\frac{1}{p}\right)}}|\xi|\right)^p\right) \tag{2.42}$$

where $\Gamma(x)$ is the gamma function.[8] Figure 2.6 shows the optimal value of ν for polynomial degrees in the interval $[1, 10]$. For convenience, the explicit numerical values are repeated in table 2.3.

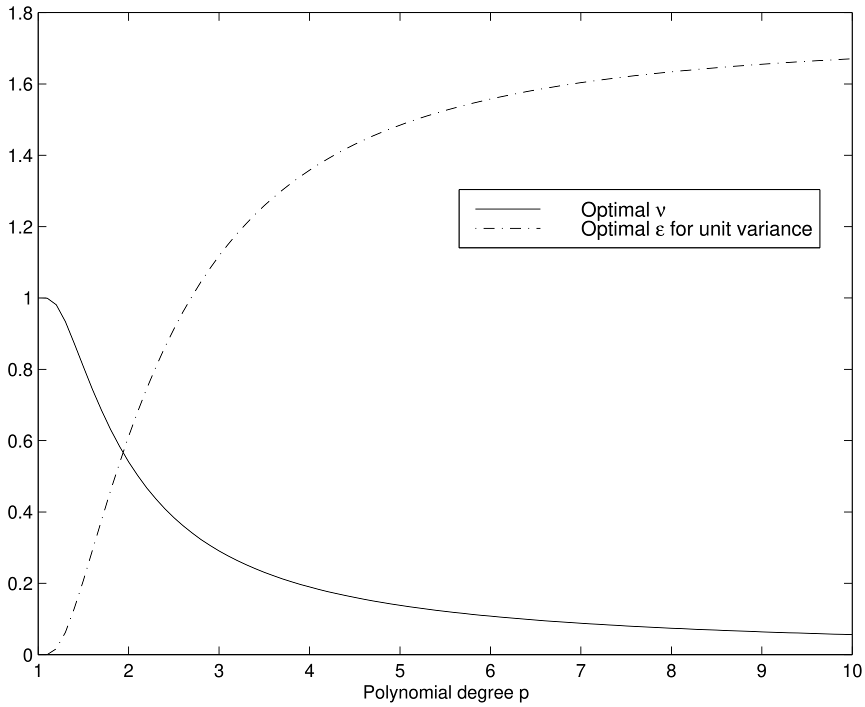

Figure 2.6 Optimal ν and ϵ for various degrees of polynomial additive noise.

Observe that the "lighter–tailed" the distribution becomes, the smaller ν are optimal. This is reasonable as only for very long tails of the distribution (data with many outliers) it appears reasonable to use an early cutoff on the influence of the data (by basically giving all data equal influence via $\alpha_i = C/m$).

8. Equation (2.42) can be checked using some computer algebra program, e.g. Maple.

Polynomial Degree p	1	2	3	4	5
Optimal ν	1	0.5405	0.2909	0.1898	0.1384
Optimal ϵ for unit variance	0	0.6120	1.1180	1.3583	1.4844
Polynomial Degree p	6	7	8	9	10
Optimal ν	0.1080	0.0881	0.0743	0.0641	0.0563
Optimal ϵ for unit variance	1.5576	1.6035	1.6339	1.6551	1.6704

Table 2.3 Optimal ν and ϵ for various degrees of polynomial additive noise.

2.4 Summing Up

Contrary to common belief there is no explicit inherent connection between the overall setting of SV machines and the particular ε–insensitive or soft margin cost functions mainly used in this context. In fact, the extension of the model of admissible cost functions to arbitrary convex ones *can* lead to improved generalization performance, while leaving the computational complexity rather unchanged (at least for up to medium sized problems of up to 3000 patterns).

The problem of choosing an optimal cost function from this (now even larger) family can be solved in the asymptotic case of large sample size for a toy model of one estimating one parameter. Astonishingly, the predictions still work in the finite sample size case quite well. However one has to compute the optimal parameter setting beforehand, knowing both the variance and the noise model of the data. Thus, by itself, this additional insight is not very useful.

However, it can be quite effective when combined with a new variant of SV regression, where the margin is adjusted automatically. As shown in the previous section, one can compute an asymptotically optimal tradeoff parameter for the margin for each class of noise models. This solves the problem of having to know the *variance* beforehand. However it does not address the problem of knowing the *noise model*, i.e. $\mathfrak{p}$. Future research will show whether also this problem can be come by as effectively.

3 Kernels and Regularization

All the considerations so far only led to *linear* functions due to the nature of the dot product in $\mathcal{X}$. A possible way to extend SV machines is to extract features, e.g. all quadratic monomials that can be constructed from $x \in \mathbb{R}^d$, and compute linear dot products there.[1]

Roadmap

The present chapter is central to this thesis, as it establishes a connection between regularization operators and linear expansions in feature spaces. This, together with Mercer's theorem constitutes the foundation of methods for capacity control for a wide range of regularizers.

Feature maps build the basis for the introduction of kernels with the definition of the latter as dot products between mapped data in feature space. It is shown that all basic equations from linear SV regression remain virtually unchanged by simply replacing dot products by kernel functions. Mercer's theorem states a condition under which kernels may be used in SV machines, thus obviating the need to know the feature map explicitly. This, however, raises the problem (to be answered in the subsequent section) that in many cases it is by no means obvious that linear functions in highdimensional spaces may exhibit good generalization performance.

The connection between kernels and regularization is made in the next section by showing that the kernels k are Green's functions of corresponding regularization operators. Thus an equivalence between SV regression and regularization networks is proven. Examples for both continuous and discrete expansions in eigensystems provide the necessary tools for the analysis of some particular popular kernel functions. Moreover the techniques also allow an analysis of ridge regression in the regularization context.

Translation invariant kernels are a special, yet important case of SV kernels. Thus they are analyzed first. It is shown that the Fourier transform diagonalizes the regularization operators in this context, i.e. that the Fourier transform of the kernel shows the filter properties of a SV machine in frequency domain. Detailed analysis is carried out for Gaussian, periodical Gaussian, B_n–spline, and Dirichlet

1. Nilsson [1965] proposed a similar approach, however with explicit evaluation of the maps into feature space.

kernels. Finally, in the case of prior knowledge about the power spectrum of the estimate, it is shown in the maximum likelihood context that a matching kernel should be given by the inverse Fourier transform of the power spectrum.

Whilst the previous exposition was restricted to functions defined in (subsets of) $\mathbb{R}$, the analytic methods are extended to multivariate regression. Again, translation invariant kernels are analyzed, this time Gaussian, exponential, and "damped harmonic oscillator" rbfs, and their regularization properties in Fourier space are stated. A note on invariances other than translation, and how corresponding kernels could be obtained, concludes the exposition of this chapter.

The other popular group of kernels consists of (in)homogeneous polynomial ones. A decomposition of the latter in terms of eigenfunctions, the identification of a corresponding regularization operator and rules how to construct further kernels based on these techniques conclude the sections on specific SV kernels.

The next sections aim at extensions of the basic framework. First a non existence statement of vector valued regularization operators is given which holds, provided some basic assumptions (like permutation symmetry) on the space of target values are satisfied. This rules out the search for possibly elaborate means of controlling capacity in the multi output case.

Using results from interpolation theory the class of kernels itself is extended to conditionally positive functions. This also leads to new criteria for testing which functions might satisfy Mercer's condition. Moreover, the requirement of orthogonality with respect to certain polynomial subspaces, which is needed for some of the new functions, leads to algorithmic modifications of the basic SV setting.

These very modifications show the way to yet another extension — semiparametric modelling with SV machines. In contrast to previous attempts of encoding prior knowledge *in* the kernel the idea is to encode this information by creating a nullspace of the regularization operator, i.e. to encode prior knowledge *outside* the kernel.

Relations of semiparametric modelling to other types of regularization such as linear programming constraints and a worked through example conclude the chapter.

3.1 Feature Maps

The SV equations are "non–linearized" in the following way. Define the map $\Phi : \mathcal{X} \to \mathfrak{S}$ where $\mathfrak{S}$ is *some* feature space. Now instead of (1.3) assume f to take on the form:

$$f(x) = \langle w, \Phi(x) \rangle + b \text{ with } w \in \mathfrak{S}, b \in \mathcal{Y} \tag{3.1}$$

where $\langle \cdot, \cdot \rangle$ denotes the dot product in feature space $\mathfrak{S}$. As all the equations concerning the SV optimization problem can be written in terms of dot products (cf. (1.11) and (1.12)), it is straightforward to replace every occurence of x by $\Phi(x)$, and obtain a nonlinear algorithm.

However, this plain setting has its limitations. Sure, one may compute many features from input data, but this is quite expensive, with computational cost increasing (at least) linearly in the number of features. Hence the dimensionality of $\Phi(x)$ is limited to a few thousand features. Moreover, it is difficult to compute the "right" features, requiring a serious amount of domain knowledge for the problem at hand. Finally, it is not always clear what flatness in "feature space" means for f which is defined on input space $\mathcal{X}$. A solution to these problems can be obtained by a simple change of notation.

$$\langle \Phi(x), \Phi(x') \rangle =: k(x, x') \tag{3.2}$$

Thus $k(x, x')$ replaces $\langle x, x' \rangle$. One obtains the equations for nonlinear SV regression:

$$\text{maximize} \quad \begin{cases} -\frac{1}{2} \sum\limits_{i,j=1}^{m} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) k(x_i, x_j) \\ -\varepsilon \sum\limits_{i=1}^{m} (\alpha_i + \alpha_i^*) + \sum\limits_{i=1}^{m} y_i (\alpha_i - \alpha_i^*) \end{cases} \tag{3.3}$$

$$\text{subject to} \quad \begin{cases} \sum\limits_{i=1}^{m} (\alpha_i - \alpha_i^*) & = & 0 \\ \alpha_i, \alpha_i^* & \in & [0, C] \end{cases} \tag{3.4}$$

with

$$f(x) = \sum_{i=1}^{m} (\alpha_i - \alpha_i^*) k(x_i, x) + b. \tag{3.5}$$

f can be expressed in terms of k alone (Φ appears only implicitly through the dot product in $\mathfrak{S}$). If $k(x, x')$ is a function that can be computed *easily*, it is reasonable to use k instead of $\Phi(x)$. A simple example shows this situation:

Example 3.1 Quadratic Features on $\mathbb{R}^2$ (cf. [Vapnik, 1995])
Define $\Phi : \mathbb{R}^2 \to \mathbb{R}^3$ as

$$\Phi(x) = (x_1^2, \sqrt{2} x_1 x_2, x_2^2). \tag{3.6}$$

In this case one obtains

$$k(x, x') = \langle \Phi(x), \Phi(x') \rangle = x_1^2 {x'}_1^2 + 2 x_1 x_2 {x'}_1 {x'}_2 + x_2^2 {x'}_2^2 = \langle x, x' \rangle^2 \tag{3.7}$$

It follows directly from [Poggio, 1975] that one can define analogous kernels of the type $k(x, x') = (\langle x, x' \rangle + 1)^p$ with $p \in \mathbb{N}$. In those cases, it is a significant improvement not having to compute $\Phi(x)$ explicitly, as the number of monomial features of order p in n dimensions grows in a combinatorial fashion. The elegance of using kernels lies in the fact that one can deal implicitly with spaces $\mathfrak{S}$ of arbitrary dimensionality without having to compute the map Φ explicitly.

3.1.1 Fundamental Properties of SV Kernels

The next question is whether it is possible to reverse the way of reasoning for kernels, i.e. under which conditions a symmetric kernel $k(x, x')$ corresponds to a dot product in some feature space $\mathfrak{S}$.

In [Mercer, 1909, Aizerman et al., 1964, Boser et al., 1992] an answer is given. Namely, a kernel corresponds to a dot product in $\mathfrak{S}$, if it satisfies Mercer's condition, i.e. if it generates a positive integral operator.

This follows directly from Mercer's theorem. The version stated below is a special case of the theorem proven in [König, 1986, p. 145]. In the following we will assume $(\mathcal{X}, \mu)$ to be a finite measure space, i.e. $\mu(\mathcal{X}) < \infty$. As usual, "almost all" means all elements of $\mathcal{X}^m$ except a set of μ^m-measure zero.

Theorem 3.1 Mercer
Suppose $k \in L_\infty(\mathcal{X} \times \mathcal{X})$ is a symmetric kernel (i.e. $k(x, x') = k(x', x)$) such that the integral operator

$$\begin{aligned} T_k &: L_2(\mathcal{X}) \to L_2(\mathcal{X}) \\ T_k &: f(\cdot) := \int_{\mathcal{X}} k(\cdot, y) f(y) d\mu(y) \end{aligned} \tag{3.8}$$

is positive. Let $\psi_j \in L_2(\mathcal{X})$ be the eigenfunction of T_k associated with the eigenvalue $\lambda_j \neq 0$ and normalized such that $\|\psi_j\|_{L_2} = 1$.

1. $(\lambda_j(T_k))_j \in \ell_1$.
2. $\psi_j \in L_\infty(\mathcal{X})$ and $\sup_j \|\psi_j\|_{L_\infty} < \infty$.
3. $k(x, y) = \sum_{j \in \mathbb{N}} \lambda_j \psi_j(x) \psi_j(y)$ (3.9)

holds for almost all (x, y), where the series converges absolutely and uniformly for almost all (x, y).

A kernel satisfying the conditions of this theorem will be called a *Mercer kernel*. k is a Mercer kernel if and only if it satisfies

$$\int_{\mathcal{X} \times \mathcal{X}} g(x) g(x') k(x, x') dx dx' \geq 0 \quad \text{for all} \quad g \in L_2(\mathcal{X}). \tag{3.10}$$

3.1.2 A Serious Problem

However, defining Φ implicitly through k also creates some serious problems. Mostly, this map and many of its properties are unknown. Even worse, this method does not generate any general rule about which kernel should be used, or why mapping into a very high dimensional space often provides good results, seemingly defying the "curse of dimensionality" [Bellman, 1961]. This dilemma can be resolved by showing that the kernels $k(x, x')$ correspond to regularization operators P, the link being that k is the Green's function of P^*P (with P^* denoting the adjoint operator of P). In other words — given a Mercer kernel find the corresponding regularization operator and vice versa.[2] For the sake of simplicity, only the case of regression will be dealt with — the considerations, however, are also valid for classification and the solution of inverse problems.

3.2 The Connection between SV Machines and Regularization Networks

As already stated above, the key to the questions raised, lies in the consideration of regularization operators. For this purpose it is necessary to briefly review the definition of the latter.

3.2.1 Regularization Networks

In regularization networks one minimizes the empirical risk functional $R_{\mathrm{emp}}[f]$ plus a regularization term, with

$$Q[f] := \frac{1}{2}\|Pf\|^2 \tag{3.11}$$

defined by a regularization operator P in the sense of Tikhonov and Arsenin [1977], i.e. P is a positive semidefinite operator mapping from the Hilbert Space $\mathcal{H}$ of functions f under consideration to a dot product space D such that the expression $\langle Pf \cdot Pg \rangle$ is well defined.[3] In the following the aim will be to show that (3.11) and (1.23) are equivalent (for some kernel k corresponding to P).

For instance by choosing a suitable operator that penalizes large variations of f one can reduce the well–known overfitting effect. Another possible setting also might be an operator P mapping from $L_2(\mathbb{R}^d)$ into some Reproducing Kernel Hilbert Space [Kimeldorf and Wahba, 1971, Girosi, 1998] (see also definition 3.13). In section 3.11, a worked through example (mainly taken from [Girosi et al., 1993]) is provided for a simple regularization operator to illustrate this reasoning.

Similar to (1.22), one minimizes

$$R_{\mathrm{reg}}[f] = R_{\mathrm{emp}} + \lambda\|Pf\|^2 = \frac{1}{m}\sum_{i=1}^{m} c(x_i, y_i, f(x_i)) + \lambda\|Pf\|^2. \tag{3.12}$$

Using an expansion of f in terms of some symmetric function $k(x_i, x_j)$ (note here, that k need not fulfill Mercer's condition),

$$f(x) = \sum_i \alpha_i k(x_i, x) + b, \tag{3.13}$$

2. The further exposition in this chapter follows largely [Smola et al., 1998e].
3. Formally one has the following definitions:

$$P : \mathcal{H} \to D, \quad P^* : D \to \mathcal{H}, \text{ and } P^*P : D \to D.$$

and the cost function defined in (1.6), this leads to a quadratic programming problem similar to the one for SVs. By computing Wolfe's dual (for details of the calculations see [Smola and Schölkopf, 1998a]), and using

$$D_{ij} := \langle (Pk)(x_i, .), (Pk)(x_j, .) \rangle \tag{3.14}$$

$$K_{ij} := k(x_i, x_j) \tag{3.15}$$

one gets $\alpha = D^{-1}K(\beta - \beta^*)$, with β_i, β_i^* being the solution of

$$\begin{array}{ll} \text{minimize} & \frac{1}{2}\sum\limits_{i,j=1}^{m}(\beta_i^* - \beta_i)(\beta_j^* - \beta_j)(KD^{-1}K)_{ij} + \\ & \sum\limits_{i=1}^{m}(\epsilon(\beta_i^* + \beta_i) - y_i(\beta_i^* - \beta_i)) \\ \text{subject to} & \sum\limits_{i=1}^{m}(\beta_i - \beta_i^*) = 0, \quad \beta_i, \beta_i^* \in [0, \frac{1}{m\lambda}] \end{array} \tag{3.16}$$

3.2.2 The Relation to SV Machines

Comparing (3.4) with (3.16) leads to the question if and under which condition the two methods might be equivalent and therefore also under which conditions regularization networks might lead to sparse decompositions (i.e. only a few of the expansion coefficients α_i in f would differ from zero). A sufficient condition is $D = K$ (thus $KD^{-1}K = K$), i.e.

$$\boxed{k(x_i, x_j) = \langle (Pk)(x_i, .), (Pk)(x_j, .) \rangle} \quad \text{(self consistency)} \tag{3.17}$$

In the case of K not having full rank D is only required to be the inverse on the image of K. The pseudoinverse for instance is such a matrix. Eq. (3.17) is the main equation of this chapter. The goal now is to solve the following two problems:

- Given a regularization operator P, find a kernel k such that a SV machine using k will not only enforce flatness in feature space, but also correspond to minimizing a regularized risk functional with P as regularization operator.
- Given an SV kernel k, find a regularization operator P such that a SV machine using this kernel can be viewed as a Regularization Network using P.

The two problems can be solved by employing the concept of Green's functions as described in [Girosi et al., 1993]. These functions were introduced in the context of solving differential equations. For the current purpose, it is sufficient to know that the Green's functions $G_{x_i}(x)$ of P^*P satisfy[4]

$$(P^*PG_{x_i})(x) = \delta_{x_i}(x). \tag{3.18}$$

The relationship between kernels and regularization operators is formalized in the following proposition:

4. $\delta_{x_i}(x)$ is the δ–distribution, which has the property that $\langle f, \delta_{x_i} \rangle = f(x_i)$.

Proposition 3.2 Green's functions and Mercer Kernels

Be P a regularization operator, and G be the Green's function of P^*P. Then G is a Mercer Kernel such that $D = K$, i.e. $G(x_i, x_j) = \langle (PG)(x_i, .), (PG)(x_j, .) \rangle$. SV machines using G minimize the regularized risk functional with $\|Pf\|^2$ as regularizer.[5]

Proof Substituting (3.18) into $G_{x_j}(x_i) = \left\langle G_{x_j}(.), \delta_{x_i}(.) \right\rangle$ yields

$$G_{x_j}(x_i) = \left\langle (PG_{x_i})(.), (PG_{x_j})(.) \right\rangle, \tag{3.19}$$

hence $G(x_i, x_j) := G_{x_i}(x_j)$ is symmetric and satisfies (3.17). Thus the SV optimization problem (3.4) is equivalent to the regularization network counterpart (3.16). Furthermore G is an admissible nonnegative kernel, as it can be written as a dot product in Hilbert Space, namely

$$G(x_i, x_j) = \langle \Phi(x_i), \Phi(x_j) \rangle \quad \text{with} \quad \Phi : x_i \longmapsto (PG_{x_i})(.). \tag{3.20}$$

which proves the proposition. ■

A similar result can be obtained by exploiting Mercer's theorem in a more straightforward manner, by using the expansion into a convergent series of its eigensystem (cf. (3.9) in theorem 3.1) directly.

This is particularly useful for the approximation of periodical functions and will come handy in section 3.3.4 as we will have to deal with a discrete eigensystem in this case.

Proposition 3.3 A Discrete Counterpart

Given a regularization operator P with an expansion of P^*P into a discrete eigensystem (λ_n, ψ_n) and a kernel k with

$$k(x_i, x_j) := \sum_n \frac{d_n}{\lambda_n} \psi_n(x_i) \psi_n(x_j), \tag{3.21}$$

where $d_n \in \{0, 1\}$ for all m, and $\sum_n \frac{d_n}{\lambda_n}$ convergent. Then k satisfies (3.17).

Proof Evaluating (3.17) and using orthonormality of the system $(\frac{d_n}{\lambda_n}, \psi_n)$, yields:

$$\begin{aligned} &\langle k(x_i, .), (P^*Pk)(x_j, .) \rangle \\ = &\left\langle \sum_n \frac{d_n}{\lambda_n} \psi_n(x_i) \psi_n(.), P^*P \left(\sum_{n'} \frac{d_{n'}}{\lambda_{n'}} \psi_{n'}(x_j) \psi_{n'}(.) \right) \right\rangle \end{aligned} \tag{3.22}$$

5. This condition is sufficient but not necessary for satisfying (3.17). Any projection of G onto an invariant subspace of P^*P would also satisfy this equation. Note that as $G(\,.\,,\,.\,)$ being a function on $\mathcal{X} \times \mathcal{X}$ the projection operator has to be applied to it as a function of both the first and the second argument.

$$= \sum_{n,n'} \frac{d_n}{\lambda_n} \frac{d_{n'}}{\lambda_{n'}} \psi_n(x_i)\psi_{n'}(x_j) \langle \psi_n(.), P^*P\psi_{n'}(.)\rangle$$
$$= \sum_n \frac{d_n}{\lambda_n} \psi_n(x_i)\psi_n(x_j) = k(x_i, x_j)$$

■

Rearranging of the summation coefficients is allowed, as the eigenfunctions are orthonormal and the series $\sum_n \frac{d_n}{\lambda_n}$ converges. Consequently a large class of kernels can be associated with a given regularization operator (and vice versa) thereby restricting oneself to some subspace of the eigensystem of P^*P.

The intuition of this reasoning is, that there exists a one to one correspondence between kernels and regularization operators only on the image of $\mathcal{H}$ under the integral operator $(T_k f)(x) := \int k(x,y)f(y)dy$, namely that T_k and P^*P are inverse to another. On the null space of T_k, however, the regularization operator P^*P may take on an arbitrary form. In this case k still will fulfill the self consistency condition.

Excluding eigenfunctions of P^*P from the kernel expansion effectively decreases the expressive power of the set of approximating functions, i.e. one limits the capacity of the system of functions. Removing low capacity (i.e. very flat) eigenfunctions from the expansion will have an adverse effect, though, as the data will have to be approximated by the higher capacity functions.

In the following this relationship will be exploited in both ways: to compute Green's functions for a given regularization operator P and to infer the regularization operator from a given kernel k.

3.3 Translation Invariant Kernels

Now consider more specifically regularization operators P that may be written as multiplications in Fourier space (i.e. P^*P is diagonalized in the Fourier basis)

$$\langle Pf, Pg\rangle = \frac{1}{(2\pi)^{n/2}} \int_\Omega \frac{\overline{\tilde{f}(\omega)}\tilde{g}(\omega)}{P(\omega)} d\omega \tag{3.23}$$

with $\tilde{f}(\omega)$ denoting the Fourier transform of $f(x)$, and $P(\omega) = P(-\omega)$ real valued, nonnegative and converging to 0 for $|\omega| \to \infty$ and $\Omega := \text{supp}[P(\omega)]$. *Small* values of $P(\omega)$ correspond to a *strong* attenuation of the corresponding frequencies. Hence small values of $P(\omega)$ for large ω are desirable, since high frequency components of $\tilde{f}$ correspond to rapid changes in f. Thus $P(\omega)$ describes the filter properties of P^*P — note that no attenuation takes place for $P(\omega) = 0$, as these frequencies have been excluded from the integration domain Ω.

For regularization operators defined in Fourier Space by (3.23) it can be shown by exploiting $P(\omega) = P(-\omega) = \overline{P(\omega)}$ that

$$G(x_i, x) = \frac{1}{(2\pi)^{n/2}} \int_{\mathbb{R}^d} e^{i\omega(x_i - x)} P(\omega) d\omega \tag{3.24}$$

is a corresponding Green's function satisfying translational invariance, i.e.

$$G(x_i, x_j) = G(x_i - x_j) \text{ and } \tilde{G}(\omega) = P(\omega). \tag{3.25}$$

For the proof, one only has to check that G satisfies (3.17). This yields an efficient tool for analyzing SV kernels and the types of capacity control they exhibit. In fact, the above is a special case of Bochner's theorem [Bochner, 1959] stating that the Fourier transform of a positive measure constitutes a positive Hilbert Schmidt kernel.

3.3.1 B_n-splines

Vapnik et al. [1997] proposed to use B_n-splines (see Fig. 3.1) as building blocks for kernels, i.e.

$$k(x) = \prod_{i=1}^{d} B_n(x_i) \tag{3.26}$$

with $x \in \mathbb{R}^d$. For the sake of simplicity, consider the case $d = 1$. Recalling the definition (up to scaling factors) of B_n splines as $n + 1$ times convolutions of the unit interval (cf. [Unser et al., 1991])

$$B_n = \bigotimes^{n+1} 1_{[-0.5, 0.5]}, \tag{3.27}$$

one can utilize the above result and the Fourier–Plancherel identity to construct the Fourier representation of the corresponding regularization operator. Up to a multiplicative constant, it equals

$$P(\omega) = \tilde{k}(\omega) = \prod_{i=1}^{d} \operatorname{sinc}^{(n+1)}\left(\frac{\omega_i}{2}\right). \tag{3.28}$$

This solves the question why only B–splines of odd order are admissible, although both even and odd order B–splines converge to a Gaussian for $n \to \infty$ due to the law of large numbers: the even ones have negative parts in the Fourier spectrum (which would result in an amplification of the corresponding frequency components). The zeros in $\tilde{k}$ stem from the fact that B_n has only compact support $[-(n+1)/2, (n+1)/2]$. By using this kernel one trades reduced computational complexity in calculating f (one only has to take points into account whose distance $\|x_i - x_j\|$ is smaller than c) for a possibly worse performance of the regularization operator as it completely removes frequencies ω_p with $\tilde{k}(\omega_p) = 0$.

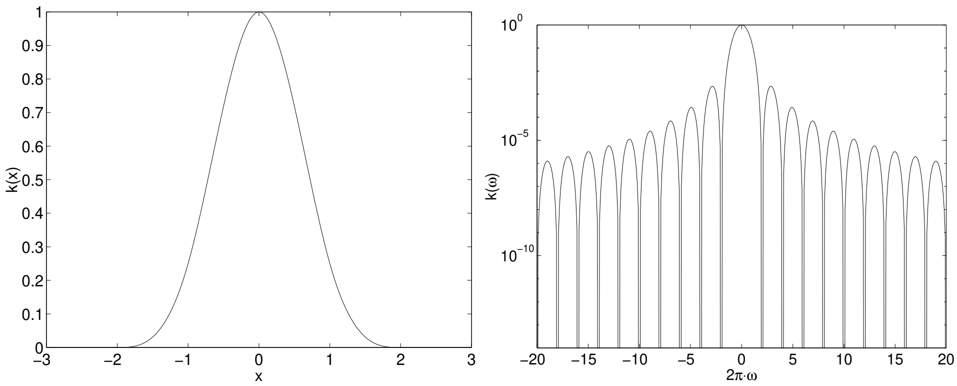

Figure 3.1 Left: B_3–spline kernel. Right: Fourier transform of k (in log scale).

3.3.2 Gaussian Kernels

Following the exposition of Yuille and Grzywacz [1988] as described in [Girosi et al., 1993], one can see that for the pseudodifferential operator[6]

$$\|Pf\|^2 = \int dx \sum_n \frac{\sigma^{2n}}{n!2^n}(O^n f(x))^2 \tag{3.29}$$

with $O^{2n} = \Delta^n$ and $O^{2n+1} = \nabla\Delta^n$, Δ being the Laplacian and ∇ the Gradient operator, we get Gaussians kernels (see Fig. 3.2)

$$k(x) = \exp\left(-\frac{\|x\|^2}{2\sigma^2}\right). \tag{3.30}$$

Moreover, we can provide an equivalent representation of P in terms of its Fourier properties, i.e. $P(\omega) = \exp\left(-\frac{\sigma^2\|\omega\|^2}{2}\right)$ up to a multiplicative constant. Training a SV machine with Gaussian RBF kernels [Schölkopf et al., 1997] corresponds to minimizing the specific cost function with a regularization operator of type (3.29). Recall that (3.29) means that all derivatives of f are penalized to obtain a very smooth estimate. This also explains the good performance of SV machines in this case, as it is by no means obvious that choosing a flat function in *some* high dimensional space will correspond to a simple function in low dimensional space (see section 3.3.3 for a counterexample).

Empirical findings show that Gaussian kernels tend to give good performance under general smoothness assumptions, and therefore should be considered especially if no additional knowledge of the data is available.

6. Roughly speaking a pseudodifferential operator differs from a differential operator insofar as it may contain an infinite number of differential operators. These correspond to a Taylor expansion of the operator in Fourier domain. Note the additional requirement that the arguments lie inside the radius of convergence.

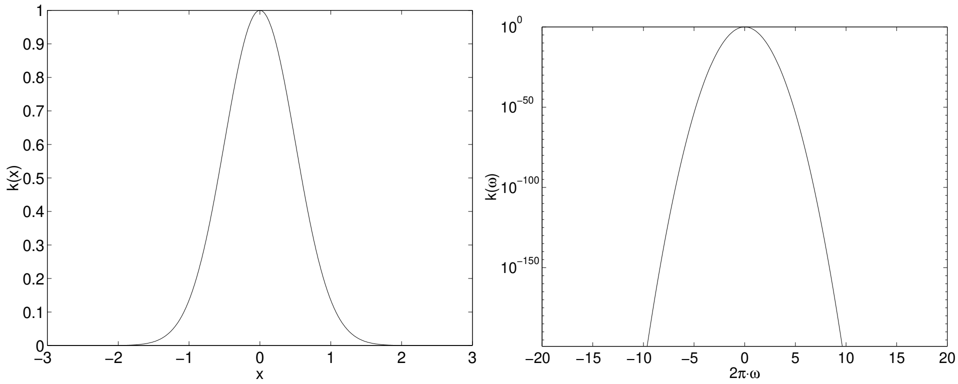

Figure 3.2 Left: Gaussian kernel with standard deviation 0.5. Right: Fourier transform of the kernel.

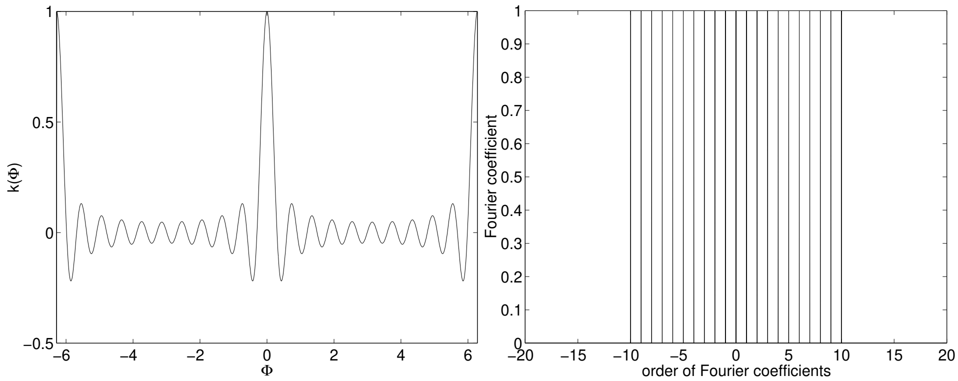

Figure 3.3 Left: Dirichlet kernel of order 10. Note that this kernel is periodical. Right: Fourier transform of the kernel.

3.3.3 Dirichlet Kernels

Vapnik et al. [1997] also introduced a class of kernels generating Fourier expansions to interpolate data on $\mathbb{R}^d$, namely

$$k(x) := 2\sum_{j=0}^{n} cosjx = \frac{\sin(2n+1)\frac{x}{2}}{\sin\frac{x}{2}}. \tag{3.31}$$

As in section 3.3.1 consider $x \in \mathbb{R}^1$ to avoid tedious notation. By construction, this kernel corresponds to $P(\omega) = \frac{1}{2}\sum_{i=-n}^{n} \delta_i(\omega)$. A regularization operator with these properties, however, may not be desirable, as it only damps a finite number of frequencies (cf. Fig. 3.3) and leaves all other frequencies unchanged which can lead to overfitting (Fig. 3.4).

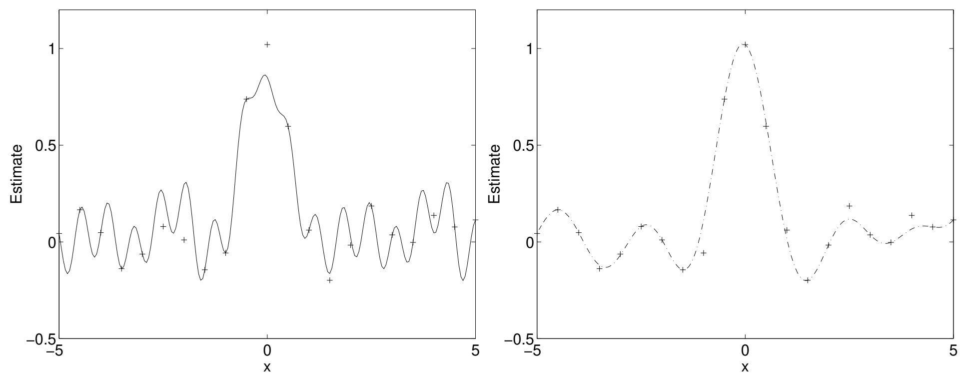

Figure 3.4 Left: Regression with a Dirichlet Kernel of order $N = 10$. One can clearly observe the overfitting (solid line: interpolation, '+': original data). Right: Regression of the same data with a Gaussian Kernel of width $\sigma^2 = 1$ (dash dotted line: interpolation, '+': original data).

In some cases it might be useful to approximate periodical functions, e.g. functions defined on a circle. This leads to the second possible type of translation invariant kernel functions, namely functions defined on factor spaces: defining translation invariant kernels on a bounded interval is not a reasonable concept as the data would hit the boundaries of the interval when translated by a large amount. Therefore only unbounded intervals and factor spaces are possible domains.

Without loss of generality assume the period to be 2π — thus consider translation invariance on $\mathbb{R}/2\pi$. The next section shows the consequences of this setting for the operator defined in section 3.3.2.

3.3.4 Periodical Gaussian Kernels

Analogously to (3.29), define a regularization operator on functions on $[0, 2\pi]^d$ by

$$\|Pf\|^2 = \pi^{-d} \int_{[0,2\pi]^d} dx \sum_n \frac{\sigma^{2n}}{n! 2^n} (O^n f(x))^2 \tag{3.32}$$

with O as in section 3.3.2. For the sake of simplicity assume $d = 1$. A generalization to multidimensional kernels is straightforward.

It is easy to check that the Fourier basis $\{\frac{1}{2}, \sin(nx), \cos(nx), n \in \mathbb{N}\}$ is an eigensystem of the operator defined above, with eigenvalues $\exp(\frac{n^2\sigma^2}{2})$. Now apply proposition 3.3, taking into account all eigenfunctions except $n = 0$. This yields the following kernel:

$$\begin{aligned} k(x, x') &= \sum_{n=1}^{\infty} e^{-\frac{n^2\sigma^2}{2}} (\sin(nx)\sin(nx') + \cos(nx)\cos(nx')) \\ &= \sum_{n=1}^{\infty} e^{-\frac{n^2\sigma^2}{2}} \cos(n(x - x')) \end{aligned} \tag{3.33}$$

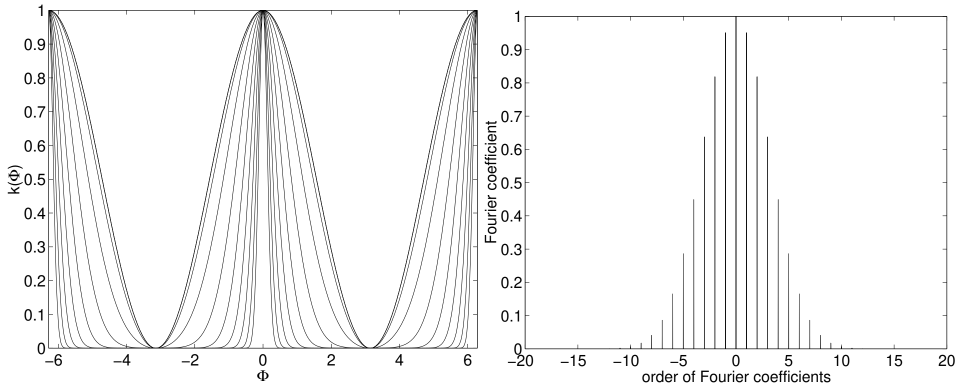

Figure 3.5 Left: Periodical Gaussian kernel for several values of σ (normalized to 1 as its maximum and 0 as its minimum value). Peaked functions correspond to small σ. Right: Fourier coefficients of the kernel for $\sigma^2 = 0.1$

For practical purposes one may truncate the expansion after a finite number of terms. Moreover k can be rescaled to have a range of exactly $[0, 1]$ by using the positive offset $\sum_{n=1}^{\infty} (-1)^{n-1} e^{-\frac{n^2\sigma^2}{2}}$ and the scaling factor $\frac{1}{2} \sum_{n=1}^{\infty} e^{-\frac{(2n-1)^2\sigma^2}{2}}$ (cf. Fig. 3.5). Another kernel k_p for P on a periodical domain can by found by letting

$$k_p(x, x') := \sum_{n \in \mathbb{Z}} k(x - x' + 2\pi n) \tag{3.34}$$

as can be checked easily.

In the context of periodical functions, the difference between this kernel and the Dirichlet kernel of section 3.3.3 is that the latter does not distinguish between the different frequency components in $\omega \in \{-n\pi, \ldots, n\pi\}$. However, it effectively limits the maximum capacity of the system to an approximation of the data with a Fourier expansion up to the order n.

3.3.5 Practical Implications

The question that arises now is which translation invariant kernel to choose. One can think of two extreme situations.

- Suppose that the shape of the power spectrum[7] $\mathrm{Pow}[f](\omega)$ of the function one would like to estimate is known beforehand. In this case one should choose k such that $\tilde{k}$ matches the expected value of the power spectrum of f. This can be seen as follows in a maximum likelihood setting:

The power spectrum of a function $f(x) = \sum_j \alpha_j k(x_j, x)$ (i.e. the Fourier transform

7. In the following $\mathrm{Pow}[f](\omega)$ denotes the power spectrum of f at frequency ω. Similarly $F[f](\omega)$ denotes the Fourier transform of f at ω.

of its autocorrelation) is defined as

$$\mathrm{Pow}[f](\omega) := |F[f](\omega)|^2. \tag{3.35}$$

One has to show that the expected value of the power spectrum of f has to be proportional to the autocorrelation of the kernel function itself, i.e.

$$E\left[\mathrm{Pow}[f](\omega)\right] \propto \mathrm{Pow}[k](\omega). \tag{3.36}$$

By the linearity of the Fourier transform, $F[f](\omega)$ can be written as

$$F[f](\omega) = F[k](\omega) \sum_j \alpha_j e^{-i\omega x_j}. \tag{3.37}$$

Hence

$$\begin{aligned} E\left[\mathrm{Pow}[f](\omega)\right] &= E\left[\mathrm{Pow}[k](\omega) e^{i\omega\tau} \sum_j \overline{\alpha_j} e^{i\omega x_j} \sum_{j'} \alpha_{j'} e^{-i\omega x_{j'}}\right] \\ &= \mathrm{Pow}[k](\omega) E\left[\sum_j \overline{\alpha_j} e^{i\omega x_j} \sum_{j'} \alpha_{j'} e^{-i\omega x_{j'}}\right] \\ &= \mathrm{Pow}[k](\omega) E\left[\sum\nolimits_j |\alpha_j|^2\right] \end{aligned} \tag{3.38}$$

as $E[\overline{\alpha_j}\alpha_{j'} e^{i\omega(x_j - x_{j'})}] = \delta_{jj'}$ under the assumption of translation invariance of the setting. The same result can be obtained directly by construction in a Gaussian process setting — there the functions f are estimated according to a covariance (= autocorrelation) function, namely $k(x, x')$.

▪ However, if one knows very little about the given data, a general smoothness assumption is a reasonable choice. Thus a Gaussian kernel like in section 3.3.2 or 3.3.4 is recommendable. If computing time is important one might moreover consider kernels with compact support, e.g. using the B_n–spline kernels of section 3.3.1. This choice will cause many matrix elements $k_{ij} = k(x_i - x_j)$ to vanish.

The usual scenario will be in between the two extreme cases and one will have some limited prior knowledge available which should be used in the choice of kernels as the goal of the present reasoning is to give a guide to selection of kernels through a deeper understanding of the regularization properties. For more information on using prior knowledge for choosing kernels, e.g. by explicit construction of kernels exhibiting only a limited amount of interactions, see [Schölkopf et al., 1998].

Prior knowledge can also be used to determine the free parameters of the kernel, e.g. its width (σ) in sections 3.3.2 and 3.3.4 . Besides that model selection principles like structural risk minimization [Vapnik, 1982], cross validation [Stone, 1974, Amari et al., 1997, Kearns, 1997, Guyon et al., 1998], MDL [Rissanen, 1985], Bayesian methods [MacKay, 1991, Bishop, 1995], etc. can be employed. See also chapters 7 and 8 on generalization bounds for kernels.

Choosing a small width of the kernels can lead to high generalization error as it effectively decouples the separate basis functions of the kernel expansion into very

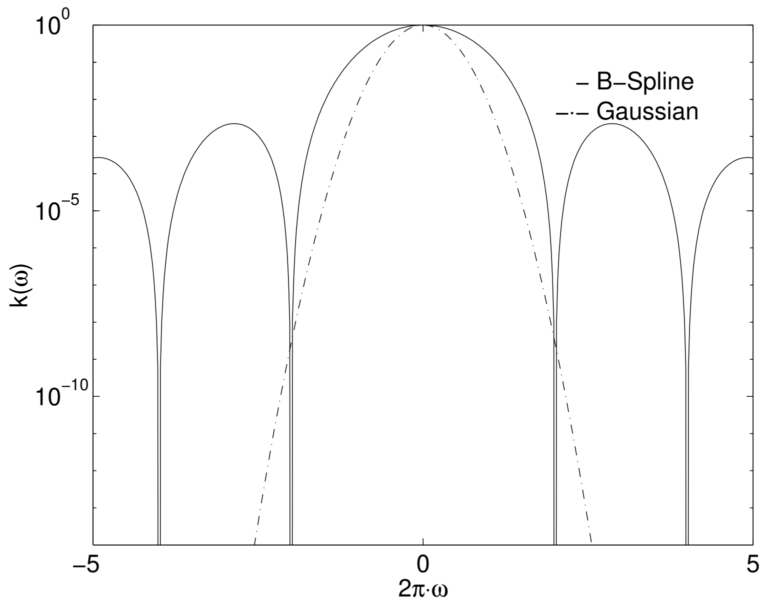

Figure 3.6 Comparison of regularization properties in the low frequency domain of B_3–spline kernel and Gaussian kernel ($\sigma^2 = 20$). Down to an attenuation factor of $5 \cdot 10^{-3}$, i.e. in the inteval $[-4\pi, 4\pi]$ both types of kernels exhibit similar filter characteristics.

localized functions which is equivalent to memorizing the data, whereas an overly wide kernel tends to oversmooth.

Finally, note that the choice of the width may be more important than the actual functional form of the kernel. There may be little difference in the relevant part of the filter properties between e.g. a B–spline and a Gaussian kernel (cf. Fig. 3.6). As will be shown in Sec. 8 this heuristic is quite true if one is interested only in uniform convergence results of a certain degree of precision — in that case only a small part of the power spectrum of k really matters.

The current section concludes with an analysis of Ridge Regression viewed under the aspect of regularization in Fourier domain.

3.3.6 Ridge Regression

A frequent choice of the regularization operator is $D = \mathbf{1}$ (see (3.14)), i.e. $D_{ij} = \delta_{ij}$. This approach often is called Ridge Regression, and is a very popular method in the context of shrinkage estimators. For instance example 1.3 leads to this setting when introducing nonlinear functions.

Now one may pose a similar question as in section 3.2.2, namely regarding the equivalence of Ridge Regression and Support Vectors. No answer is available for a direct equivalence, however, one can show that one may obtain models generated by the same type of regularization operators. The requirement for an equivalence of the latter would be

$$D_{ij} = D(x_i, x_j) = \langle (Pk)(x_i, .), (Pk)(x_j, .) \rangle = \delta_{ij} \tag{3.39}$$

for all possible choices of $x_i \in \mathbb{R}^d$. Unfortunately this requirement cannot be met for the case of the Kronecker δ, as (3.39) implies the function $D(x_0, \,.\,)$ to be nonzero only on a set with (Lebesgue) measure 0. The solution is to change the finite Kronecker δ into the more appropriate δ–distribution, i.e. $\delta(x_i - x_j)$.

By a similar reasoning as in Proposition 3.2, one can see that (3.39) holds for $k(x, y)$ being the Green's function of P. Note that as a regularization operator $(P^*P)^{\frac{1}{2}}$ is equivalent to P as one can always replace the latter by the former without any difference in the regularization properties. Therefore, without loss of generality, assume that P is a positive semidefinite endomorphism. Formally one requires

$$\langle (Pk)(x_i, .), (Pk)(x_j, .) \rangle = \left\langle \delta_{x_i}(.), \delta_{x_j}(.) \right\rangle = \delta_{x_i, x_j} \tag{3.40}$$

Again, this allows to connect regularization operators and Kernels (one has to find the Green's function of P to satisfy the equation above). For the special case of translation invariant operators denoted in Fourier space one can associate P with $P_{\text{ridge}}(\omega)$, leading to

$$\|Pf\|_2^2 = \int \left| \frac{\tilde{f}(\omega)}{P_{\text{ridge}}(\omega)} \right|^2 d\omega. \tag{3.41}$$

This expansion is possible as the Fourier transform "diagonalizes" the corresponding regularization operator, i.e. multiple applications of P become multiplications in the Fourier domain. Comparing (3.41) with (3.23) leads to the conclusion that the following relation between kernels for Support Vector Machines and Ridge Regression has to hold:

$$\tilde{P}_{\text{SV}}(\omega) = |P_{\text{ridge}}(\omega)|^2 \tag{3.42}$$

In other words, the in Ridge Regression it is the *squared* Fourier transform of the kernels that determines the regularization properties.

This connection also explains the performance of Ridge Regression Models in a smoothing regularizer context (the squared norm of the Fourier transform of kernel functions describes the regularization properties of the corresponding kernel) and allows one to 'transform' Support Vector Machines to Ridge Regression models and vice versa. Note, however, that the sparsity properties of Support Vectors are lost.

Also note (cf. section 9.2.3) that it is much more difficult to obtain good capacity bounds in this context, as very often the model complexity increases arbitrarily with the number of basis functions.

3.4 Translation Invariant Kernels in Higher Dimensions

Things get more complicated in higher dimensions. There are basically two ways for constructing kernels in $\mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ with $d > 1$, if no particular assumptions on the data are made. Firstly one could construct kernels $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ by

$$k(x - x') = k(x_1 - x'_1) \cdot \ldots \cdot k(x_d - x'_d). \tag{3.43}$$

This choice will usually lead to preferred directions in input space, as the kernels are not rotation invariant in general (the exception being Gaussian kernels).

The second approach consists in setting

$$k(x - x') = k(\|x - x'\|_{\ell_2}). \tag{3.44}$$

This approach leads to kernels which are both translation invariant and rotation invariant. It is quite straightforward, however, to generalize the exposition to the rotation asymmetric case. In order to proceed one has to define the basic ingredients needed for the further calculations.

3.4.1 Basic Tools

The d-dimensional Fourier transform is defined by

$$F : L_2(\mathbb{R}^d) \to L_2(\mathbb{R}^d) \text{ with } F[f](\omega) := \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} e^{-i\langle \omega, x\rangle} f(x) dx. \tag{3.45}$$

Then its inverse transform is given by

$$F^{-1} : L_2(\mathbb{R}^d) \to L_2(\mathbb{R}^d) \text{ with } F^{-1}[f](x) = \frac{1}{(2\pi)^{d/2}} \int_{\mathbb{R}^d} e^{i\langle \omega, x\rangle} f(\omega) d\omega. \tag{3.46}$$

Moreover for radially symmetric functions, i.e. $f(x) = f(\|x\|_2)$, one can explicitly carry out the integration on the sphere to obtain a Fourier transform which is also radially symmetric (cf. [Sneddon, 1972, Müller, 1997]):

$$F[f](\|\omega\|) = \omega^{-\nu} H_\nu[r^\nu f(r)](\|\omega\|), \tag{3.47}$$

where $\nu := \frac{1}{2}d - 1$ and H_ν is the Hankel transform over the positive real line. The latter is defined as

$$H_\nu[f](\omega) := \int_0^\infty r f(r) J_\nu(\omega r) dr. \tag{3.48}$$

Here J_ν is the Bessel function of the first kind defined by[8]

$$J_\nu(r) := r^\nu 2^{-\nu} \sum_{j=0}^{\infty} \frac{(-1)^j r^{2j}}{2^{2j} j! \Gamma(j + \nu + 1)}. \tag{3.49}$$

8. $\Gamma(x)$ is the gamma function, i.e. for $n \in \mathbb{N}$ we have $\Gamma(n+1) = n!$.

Note that $H_\nu = H_\nu^{-1}$, i.e. $f = H_\nu[H_\nu[f]]$ (in L_2) due to the Hankel inversion theorem [Sneddon, 1972]. This finally allows one to use (3.24) to compute the Green's functions directly from the regularization operators given in Fourier space in $\mathbb{R}^d$.

3.4.2 Regularization Properties of Kernels in $\mathbb{R}^d$

Now for some examples of kernels typically used in SV machines, this time in $\mathbb{R}^d$. First one has to compute the Fourier/Hankel transform of the kernels.

Example 3.2 Gaussian RBFs
For Gaussian rbfs in d dimensions one gets $k(r) = \sigma^{-d} e^{-\frac{r^2}{2\sigma^2}}$ and correspondingly

$$\begin{aligned} F[k](\omega) &= \omega^{-\nu}\sigma^{-d} H_\nu \left[r^\nu e^{-\frac{r^2}{2\sigma^2}} \right](\omega) \\ &= \omega^{-\nu}\sigma^{2(\nu+1)-d}\omega^\nu e^{-\frac{\omega^2\sigma^2}{2}} \\ &= e^{-\frac{\omega^2\sigma^2}{2}} \end{aligned} \tag{3.50}$$

or in other words — the Fourier transform of a Gaussian is a Gaussian, also in higher dimensions.

Example 3.3 Exponential RBFs
In the case of $k(r) = e^{-ar}$ one gets

$$\begin{aligned} F[k](\omega) &= \omega^{-\nu} H_\nu \left[r^\nu e^{-ar} \right](\omega) \\ &= \omega^{-\nu} 2^{\nu+1} \omega^\nu a \pi^{-\frac{1}{2}} \Gamma\left(\nu + \tfrac{3}{2}\right) \left(a^2 + \omega^2\right)^{-\nu-\frac{3}{2}} \\ &= 2^{\frac{d}{2}} a \pi^{-\frac{1}{2}} \Gamma\left(\tfrac{d}{2} + 1\right) \frac{1}{\left(a^2+\omega^2\right)^{\frac{d+1}{2}}} \end{aligned} \tag{3.51}$$

i.e. in the case of $d = 1$ one recovers the damped harmonic oscillator (in frequency domain). In general, a decay in the Fourier spectrum like $\omega^{-(d+1)}$ can be observed. Moreover the Fourier transform of k, viewed itself as a kernel, i.e. $k(r) = \left(1 + r^2\right)^{-\frac{d+1}{2}}$, yields the initial kernel as its corresponding power spectrum in Fourier domain.

Example 3.4 Damped Harmonic Oscillator
Another way to generalize the harmonic oscillator, this time in a way, that k does not depend on the dimensionality d is to set $k(r) = \frac{1}{a^2+r^2}$. Following [Watson, 1958, sec. 13.6] leads to

$$\begin{aligned} F[k](\omega) &= \omega^{-\nu} H_\nu \left[\frac{r^\nu}{a^2+r^2} \right](\omega) \\ &= \omega^{-\nu} a^\nu K_\nu(\omega a) \end{aligned} \tag{3.52}$$

where K_ν is the Bessel function of the second kind, defined by (see [Sneddon, 1972])

$$K_\nu(x) = \int_0^\infty e^{-x \cosh t} \cosh(\nu t) dt. \tag{3.53}$$

It is possible to upper bound $F[k]$ via

$$K_\nu(x) = \sqrt{\frac{\pi}{2x}} e^{-x} \left[\sum_{j=0}^{p-1} (2x)^{-j} \frac{\Gamma\left(\nu + j + \frac{1}{2}\right)}{j!\Gamma\left(\nu - j + \frac{1}{2}\right)} + \theta \cdot (2x)^{-p} \frac{\Gamma\left(\nu + p + \frac{1}{2}\right)}{j!\Gamma\left(\nu - p + \frac{1}{2}\right)} \right] \quad (3.54)$$

with $p > \nu - \frac{1}{2}$ and $\theta \in [0, 1]$ [Gradshteyn and Ryzhik, 1981, eq. (8.451.6)]). As one can see, the term in the brackets $[\cdot]$ converges to 1 for $x \to \infty$ and thus results in exponential decay of the Fourier spectrum.

Example 3.5 Generalized B_n Splines
Finally it remains to find a suitable generalization of B_n–splines to d dimensions. One method consists in defining

$$B_n^d := \bigotimes_{j=0}^{n} 1_{U_d}(.) \quad (3.55)$$

i.e. to define B_n^d to be the $n+1$–times convolution of the unit ball U_d in d dimensions. Employing the Fourier–Plancherel identity one can obtain its Fourier transform as the $(n+1)^{\text{th}}$ power of the Fourier transform of the unit ball, i.e.

$$F[B_0^d](\omega) = \omega^{-(\nu+1)} J_{\nu+1}(\omega) \quad (3.56)$$

and therefore

$$F[B_n^d](\omega) = \omega^{-(n+1)(\nu+1)} J_{\nu+1}^{n+1}(\omega). \quad (3.57)$$

One can observe that only odd n generate admissible Hilbert–Schmidt kernels, as only then the kernel has a nonnegative Fourier transform.

3.4.3 A Note on Other Invariances

The invariance of the kernels presented so far has been exploited only in the context of invariance with respect to the translation group in $\mathbb{R}^d$. Yet the methods could also be applied to other symmetry transformations corresponding to other canonical coordinate systems such as the rotation and scaling group as proposed by Segman et al. [1992] and Ferraro and Caelli [1994], e.g. to a logpolar parametrization of $\mathbb{R}^d$ (leading to the Fourier–Mellin transform), or the parametrization of manifolds. In particular any Abelian (Lie) group generates a map on $\mathcal{X}$ via its generators. This in turn can be exploited to define Fourier transforms on these domains and thereby build kernels which exhibit invariance wrt. other transformations.

3.5 Kernels of Dot–Product Type

A large class of Support Vector Kernels is not translation invariant. This includes kernels of the type

$$k(x, x') = t\left(\langle x, x' \rangle\right). \quad (3.58)$$

For instance, polynomial kernels $(\langle x, x' \rangle + c)^p$ of homogeneous ($c = 0$) or inhomogeneous type ($c \neq 0$) belong to this class. As already discussed in the beginning of this chapter it follows directly from [Poggio, 1975] that polynomial kernels satisfy Mercer's condition. Now the question arises, which regularization operator P these kernels might correspond to, and which functions t might be admissible ones. Obviously P cannot be translation invariant, as this is not the case for k either. Note that although lacking translation invariance, these kernels still exhibit (by construction) the property of rotation invariance — orthogonal transformations R are isometries of the Euclidean dot product: $\langle x, y \rangle = \langle Rx, Ry \rangle$.

Skipping tedious calculations, one can "guess" an operator satisfying (3.17) for homogeneous polynomials. The following proposition formalizes this finding

Proposition 3.4 Regularization for Polynomial Kernels
Denote by $\mathbf{n} = (n_1, \ldots, n_d) \in \mathbb{N}_0^d$ a multi index with $|\mathbf{n}| := \sum_{i=1}^d n_i$ and by

$$\binom{p}{\mathbf{n}} := \frac{p!}{(p - |\mathbf{n}|)! \prod_{i=1}^d n_i!} \tag{3.59}$$

the multinomial coefficient. Moreover let

$$D_0^{\mathbf{n}} f := \frac{1}{n_1!} \partial_{x_1}^{n_1} \ldots \frac{1}{n_d!} \partial_{x_d}^{n_d} f(x) \Big|_{x=0} \tag{3.60}$$

and $e_{\mathbf{n}}$ be an orthonormal basis, i.e. $\langle e_{\mathbf{n}}, e_{\mathbf{n}'} \rangle = \delta_{\mathbf{n}\mathbf{n}'}$.[9] Then the operator P_p

$$P_p := \sum_{|\mathbf{n}|=p} e_{\mathbf{n}} \binom{p}{\mathbf{n}}^{\frac{1}{2}} D_0^{\mathbf{n}}. \tag{3.61}$$

acts as a regularization operator and satisfies (3.17) with $k(x, x') = \langle x, x' \rangle^p$ as its Green's function.

Note that P_p is only well-defined on functions that are p times differentiable. Accordingly one has to restrict the space of functions under consideration to $\mathcal{C}^p$. This is not a major restriction as polynomial kernels are in $\mathcal{C}^\infty$ by construction.

Proof **(sketch only)** To prove the statement one has to expand k into

$$\langle x, x' \rangle^p = \sum_{|\mathbf{n}|=p} \binom{p}{\mathbf{n}} x^{|\mathbf{n}|} x'^{|\mathbf{n}|} \tag{3.62}$$

Here $x^{|\mathbf{n}|} = \prod_{i=1}^d x_i^{n_i}$. Substitution of (3.61) and (3.62) into (3.17) proves the proposition. ■

This result can be used to give an analogous expansion for the inhomogeneous case, and present a sufficient condition for $t(\langle x, x' \rangle)$ to be an admissible Mercer kernel.

9. Observe how for each $\mathbf{n}$ $D_0^{\mathbf{n}}$ extracts exactly one coefficient from the monomials of degree $\mathbf{n}$.

Going back to example 3.1 one may now construct an operator for $k(x, x') = \langle x, x' \rangle^2$. Denoting $\frac{1}{2}\partial^2_{x_1}, \partial_{x_1}\partial_{x_2}, \frac{1}{2}\partial^2_{x_2}$ the projectors onto the corresponding monomials one gets

$$P_2 = e_1 \frac{1}{2}\partial^2_{x_1} + e_2 \partial_{x_1}\partial_{x_2} + e_3 \frac{1}{2}\partial^2_{x_2}. \tag{3.63}$$

An intuitive description of P_2 would be that the data is mapped from $\mathbb{R}^2$ into a 3–dimensional feature space ($\mathfrak{S} = \mathbb{R}^3$) by computing monomials of degree 2. Subsequently one seeks to compute the *flattest* function in this new space.

It is interesting that the homogeneous polynomial kernel also satisfies the self consistency condition (3.17) for the following operator

$$P = \sum_{i=0}^{\infty} P_i. \tag{3.64}$$

Remark 3.5 Regularization for Inhomogeneous Polynomial Kernels
In order to construct an operator for inhomogeneous polynomials, one makes use of the expansion

$$(\langle x, y \rangle + c)^p = \sum_{i=1}^{p} \binom{p}{i} c^{p-|\mathbf{n}|} \langle x, y \rangle^i \tag{3.65}$$

(for convenience set $c = 1$). Hence one may decompose the inhomogeneous polynomial kernel into a series of homogeneous kernels and construct the corresponding operator by

$$P_{\text{inh}} = \sum_{i=0}^{p} \binom{p}{i}^{\frac{1}{2}} P_i = \sum_{|\mathbf{n}| \leq p} e_{\mathbf{n}} \binom{p}{\mathbf{n}}^{\frac{1}{2}} D_0^{\mathbf{n}}. \tag{3.66}$$

Exploiting this idea even further allows to state a sufficient condition for $t(\langle x, y \rangle)$ to be a Mercer kernel. As homogeneous polynomial kernels satisfy Mercer's condition so does any positive linear combination of them.

Corollary 3.6 Functions with Nonnegative Power–Series
For every function $t(x)$ that can be expanded into a uniformly convergent power series on $\mathbb{R}$ with nonnegative expansion coefficients, i.e.

$$t(x) = \sum_{i=0}^{\infty} a_i x^i \text{ with } a_i \geq 0 \tag{3.67}$$

the kernel $k(x, x') := t(\langle x, x' \rangle)$ is a Mercer kernel. The corresponding regularization operator is

$$P_t = \sum_{i=0}^{\infty} a_i^{1/2} P_p. \tag{3.68}$$

Consequently, functions like e^x, $\cosh(x)$, $\sinh(x)$, etc. could be used as possible Mercer kernels (their practical usability may be a different question, though). Moreover note that the same argument applies for $t(k(x, x'))$: if k is a Mercer kernel, and t satisfies the conditions of Corollary 3.6 then

$$t(k(x,x')) = t(\langle\Phi(x),\Phi(x')\rangle) \tag{3.69}$$

is a Mercer kernel. So (3.69) provides further means to construct more general kernels, e.g. $k(x,y) := \sinh(e^{\langle x,y\rangle})$.

3.6 Regularization for the Multi Output Case

One might conclude that a regularizer of the type $\|Pf\|^2$ is the most suitable one in all cases. In fact, it is one of the most widespread ones (cf. [Tikhonov and Arsenin, 1977, Morozov, 1984]). However, there is more to regularization than just to consider functions where $\mathcal{Y}$ is a scalar or $Q[f]$ is a quadratic functional.

Consider a straightforward extension of the functional $Q[f] = \frac{1}{2}\|Pf\|^2$ to $\frac{1}{2}\langle Pf, Pf\rangle$ where now $\mathcal{Y}$, the space of target values, is a dot product space of dimensionality greater than 1 and P remains a *scalar* operator (i.e. it acts identically and independently in each dimension / direction of $\mathcal{Y}$). It is quite difficult even to notice what has changed, and this extension definitely does not look very exciting. Yet, under some assumptions of invariance, this is the only valid extension of homogeneous second order regularization terms to vector valued functions.

Proposition 3.7 Homogeneous Invariant Regularization
Any regularization term $Q[f]$ that is both homogeneous quadratic and invariant under an irreducible orthogonal representation ρ of the group[10] $\mathcal{G}$ on $\mathcal{Y}$, i.e. satisfies

$$Q[f] \geq 0 \text{ for all } f \in \mathcal{F} \tag{3.70}$$

$$Q[af] = |a|^2 Q[f] \text{ for all scalars } a \tag{3.71}$$

$$Q[\rho(g)f] = Q[f] \text{ for all } g \in \mathcal{G} \tag{3.72}$$

is of the form

$$Q[f] = \langle Pf, Pf\rangle \text{ where } P \text{ is a scalar operator.} \tag{3.73}$$

The motivation for the requirements (3.70) to (3.72) is as follows. The necessity that a regularization term has to be positive (3.70) is self evident — at least it has to be bounded from below, which then, via a positive offset, can be transformed into a posititivity condition.

Homogeneity (3.71) is a useful condition for effective capacity control — it allows easy capacity control by noting that the entropy numbers (a quantity to be defined in chapter 7) scale in a linear, hence homogeneous, fashion when the hypothesis class is rescaled by a constant. The requirement of being quadratic is a mere algorithmic one as it allows to avoid the modulus operation in the linear or cubic case to ensure positivity.

10. $\mathcal{G}$ also may be directly defined on $\mathcal{Y}$, i.e. it might be a matrix group like $\mathrm{SU}(d)$.

Finally, the invariance has to be chosen beforehand. If it happens to be sufficiently strong, it can rule out all operators but scalar ones. Permutation symmetry is such a case — for instance in classification this would mean that all class labels are treated equally.

Proof It follows directly from (3.71) and Euler's homogeneity property, that $Q[f]$ has to be a quadratic form, thus $Q[f] = \langle f, Mf \rangle$ for some operator M. Moreover M can be written as P^*P as it has to be a positive operator (cf. (3.70)). Finally from

$$\langle Pf, Pf \rangle = \langle P\rho(g)f, P\rho(g)f \rangle \tag{3.74}$$

and the polarization equation it follows that $P^*P\rho(g) = \rho(g)P^*P$ has to hold for any $g \in \mathcal{G}$. Thus, by virtue of Schur's lemma (cf. e.g. [Hamermesh, 1962]), it follows that P^*P only may be a scalar operator. Then, without loss of generality, also P may be assumed to be scalar. ■

A consequence of the proposition above is that there exists no vector valued regularization operator satisfying the invariance conditions. Hence it is useless to look for other operators P in the presence of a sufficiently strong invariance.

Now for some practical applications of proposition 3.7, which will be stated in the form of corollaries.

Corollary 3.8 Permutation and Rotation Symmetries
Under the assumptions of proposition 3.7 both the canonical representation of the permutation group in a finite dimensional vector space $\mathcal{Y}$ and the group of orthogonal transformations on $\mathcal{Y}$ enforce scalar operators P.

This follows immediately from the fact that these groups (or representations on $\mathcal{Y}$) are unitary and irreducible on $\mathcal{Y}$ by construction.

Corollary 3.9 Kernel Expansions
Under the assumptions of proposition 3.7 the regularization functional $Q[f]$ for a kernel expansion

$$f(x) = \sum_i \alpha_i k(x_i, x) \quad \text{with} \quad \alpha_i \in \mathcal{Y}, \tag{3.75}$$

where $k(x_i, x)$ is a function mapping $\mathcal{X} \times \mathcal{X}$ to the space of scalars compatible with the dot product space $\mathcal{Y}$, can be stated as follows:

$$Q[f] = \sum_{i,j} \langle \alpha_i, \alpha_j \rangle \langle Pk(x_i, \cdot), Pk(x_j, \cdot) \rangle. \tag{3.76}$$

In particular, if k is the Green's function of P^*P, one gets

$$Q[f] = \sum_{i,j} \langle \alpha_i, \alpha_j \rangle k(x_i, x_j). \tag{3.77}$$

For possible application such as regularized principal manifolds see section 5.3.

3.7 A New Class of Support Vector Kernels

The strategy follows the lines of [Madych and Nelson, 1990] as pointed out by [Girosi et al., 1993]. The main statement is that conditionally positive definite (cpd) functions generate admissible SV kernels. This is very useful as the property of being cpd is often easier to verify than Mercer's condition, especially when combined with the results of Schoenberg and Micchelli on the connection between cpd and completely monotonic functions [Schoenberg, 1938a,b, Micchelli, 1986]. Moreover cpd functions lead to a class of SV kernels that do not necessarily satisfy Mercer's condition, yet still can be used in a (modified) SV algorithm.

3.7.1 Tools and Functions from Interpolation Theory

Definition 3.10 Conditionally Positive Definite Functions

A continuous function h, defined on $[0,\infty)$, is said to be conditionally positive definite (cpd) of order p on $\mathbb{R}^d$ if for any distinct points $x_1, \ldots, x_m \in \mathbb{R}^d$ the quadratic form

$$\sum_{i,j=1}^{m} c_i c_j h(\|x_i - x_j\|^2) \tag{3.78}$$

is nonnegative provided that the scalars $c_1, \ldots, c_m$ satisfy $\sum_{i=1}^{m} c_i p(x_i) = 0$ for all polynomials $p(\cdot)$ on $\mathbb{R}^d$ of degree lower than p.

Definition 3.11 Completely monotonic functions

A function $h(x)$ is called completely monotonic of order m if

$$(-1)^n \frac{d^n}{dx^n} h(x) \geq 0 \text{ for all } x \in \mathbb{R}_0^+ \text{ and } n \geq m. \tag{3.79}$$

It can be shown [Schoenberg, 1938a,b, Micchelli, 1986] that a function $h(x^2)$ is conditionally positive definite if and only if $h(x)$ is completely monotonic of the same order. This gives a (sometimes simpler) criterion for checking whether a function is cpd or not.

Proposition 3.12 CPD Functions and Admissible Kernels

Define Π_p^d to be the space of polynomials of degree lower than p on $\mathbb{R}^d$. Every cpd function h of order p generates an admissible Kernel for SV expansions on the space of functions f orthogonal to Π_p^d by setting $k(x_i, x_j) := h(\|x_i - x_j\|^2)$.

Proof Dyn [1991], Madych and Nelson [1990] show that cpd functions h of order p generate semi–norms $\|.\|_h$ by

$$\|f\|_h^2 := \int dx_i dx_j h(\|x_i - x_j\|^2) f(x_i) f(x_j), \tag{3.80}$$

provided that the projection of f onto Π_p^d is zero. For these functions, this, however, also defines a dot product in some feature space. Hence they can be used as SV kernels. ■

Consequently, one may use kernels like those proposed in the context of regularization networks by [Girosi et al., 1993] as SV kernels:[11]

$$k(x, x') = e^{-\beta\|x-x'\|^2} \quad \text{Gaussian}, (p = 0) \tag{3.81}$$

$$k(x, x') = -\sqrt{\|x - x'\|^2 + c^2} \quad \text{multiquadric}, (p = 1) \tag{3.82}$$

$$k(x, x') = \frac{1}{\sqrt{\|x - x'\|^2 + c^2}} \quad \text{inverse multiquadric}, (p = 0) \tag{3.83}$$

$$k(x, x') = \|x - x'\|^2 \ln \|x - x'\| \quad \text{thin plate splines}, (p = 2) \tag{3.84}$$

Here the corresponding regularization operator P is given implicitly by the seminorm (3.80) as

$$\|Pf\|_2^2 := \|f\|_h^2. \tag{3.85}$$

However one has to ensure the orthogonality of the estimate with respect to Π_p^d, i.e. ensure that $\sum_{i=1}^m c_i p(x_i) = 0$ for all polynomials $p(\cdot)$ on $\mathbb{R}^d$ of degree lower than p with c_i being the expansion coefficients of the estimate, i.e. α_i.

3.7.2 Algorithms

The next step is to state the algorithmic details how to actually compute the expansion. In order not to lose expressive power in the estimate f it is necessary to take the polynomials separately into account, i.e. modify expansion (3.13) to get

$$f(x) = \sum_{i=1}^m c_i k(x_i, x) + p(x) \text{ with } p(x) \in \Pi_p^d \tag{3.86}$$

Both of these issues can be addressed by splitting f into a term $f^\perp$ orthogonal to Π_p^d for which $\|f^\perp\|_h^2$ is well defined, and a polynomial term which will not be regularized at all. A logical extension of this concept leads to semiparametric regression schemes as discussed in section 3.8. Hence the regularized risk functional (3.12) takes on the

11. The necessary conditions derived by Burges [1998a] using differential geometrical methods on the first and second derivative of k are a subset of the necessary and sufficient requirement of positive definiteness.

following form

$$R_{\text{reg}}[f] = R_{\text{emp}}[f] + \lambda \|f^{\perp}\|_h^2 \tag{3.87}$$

with $f^{\perp} := (1 - \text{Proj}[\Pi_p^d])f$ and Proj[.] denoting the projection operator. Note the similarity to the expansions in [Kimeldorf and Wahba, 1971, Cox and O'Sullivan, 1990], which were stated to obtain expansions in the context of Reproducing Kernel Hilbert Spaces (cf. definition 3.13). Repeating the calculations that led to (3.4) yields a similar optimization problem with the difference being that the equality constraint

$$\sum_{i=1}^{m} (\beta_i - \beta_i^*) = 0 \tag{3.88}$$

has been replaced with

$$\sum_{i=1}^{m} (\beta_i - \beta_i^*) p(x_i) = 0 \text{ for all } p \in \Pi_p^d \tag{3.89}$$

Note that for $p = 1$ condition (3.89) reduces to (3.88) as Π_1^d contains only the constant function. The resulting optimization problem is positive semidefinite, however only in the feasible region given by the equality constraints. Some of the eigenvalues of the matrix K may be negative in the space of coefficients not satisfying (3.89). It can be seen very easily for the multiquadric case (3.82) — all entries K_{ij} are negative. This can lead to numerical instabilities for quadratic programming codes as they usually assume the quadratic matrix to be positive semidefinite not only in the feasible region of the parameters but on the whole space [More and Toraldo, 1991, Vanderbei, 1994]. A way to solve this problem is to remove the space $\mathcal{S}$ spanned by all polynomials Π_p^d on the data $\{x_1, \ldots, x_m\}$ from the image of K_{ij} while keeping it symmetric by substituting K_{ij} with $((\mathbf{1} - \text{Proj}[\mathcal{S}])^t K (\mathbf{1} - \text{Proj}[\mathcal{S}]))_{ij}$.

Example 3.6 Projecting out Π_1^d

The space Π_1^d consists of all polynomials on $\mathbb{R}^d$ of degree lower than 1, i.e. only of the constant function. Hence $\mathcal{S}$, the span of Π_1^d on any nonempty set $\{x_1, \ldots, x_m\} \subset \mathbb{R}^d$ is span$\{\vec{1}\}$. Consequently $\frac{1}{m}\vec{1}\vec{1}^t$ is a projector onto that space and one gets

$$k_{ij} \longmapsto \left(\left(\mathbf{1} - \tfrac{1}{m}\vec{1}\vec{1}^t\right) k \left(\mathbf{1} - \tfrac{1}{m}\vec{1}\vec{1}^t\right) \right)_{ij} \tag{3.90}$$

Curiously enough the matrix one obtains by this method is identical to the one being diagonalized in Kernel PCA [Schölkopf et al., 1998a] (see also section 5.1). This is clear, as projecting out the span of constant polynomials is equivalent to centering in feature space.

Note that the standard SV approach already can deal with $p = 1$, due to the constraint $\sum_{i=1}^{m} (\alpha_i - \alpha_i^*) = 0$.

Example 3.7 Projecting out Π_2^d

Π_2^d consists of all constant and linear functions on $\{x_1, \ldots, x_m\}$. Here $\mathcal{S} = \operatorname{span}(\{v_0, \ldots, v_n\})$ with

$$\begin{aligned} v_0 &:= (1, \ldots, 1) \\ v_i &:= (x_{i1}, \ldots, x_{im}) \text{ for } i \in \{1, \ldots, d\} \end{aligned} \tag{3.91}$$

In the case of $m \leq d+1$, already a linear model will suffice to reduce $R_{\text{reg}}[f]$ to 0. In this case the solution of the quadratic optimization problem is simply 0 as k_{ij} will have rank 0 after the projection.

For $m > d+1$ one has to transform $v_0 \ldots, v_n$ into an orthonormal basis $e_0, \ldots, e_d$ of $\mathcal{S}$, e.g. by applying the Gram–Schmidt procedure. This in turn allows one to construct an orthogonal projector onto $\mathcal{S}$ and the corresponding modified matrix from k_{ij}.

As one can observe, only cpd functions of order up to 2 are of practical interest for SV methods, as the number of additional constraints and projection operations increases in a combinatorial way, rendering the calculations computationally infeasible for $p > 2$. For $p = 3$ for instance, one would have to project out all quadratic monomials that can be computed in d dimensions ($d = \dim(\mathcal{X})$). This is clearly too expensive to compute, provided d is sufficiently large.

3.8 Semiparametric Estimation

The developments in the previous section lead to the question whether the action of projecting out subspaces of the regularization operator might not be useful by themselves. This is true, indeed, as it leads to semiparametric estimators.[12]

3.8.1 Why Semiparametric Models are useful

One of the strengths of Support Vector (SV) machines is that they are *nonparametric* techniques, where one does not have to e.g. specify the number of basis functions beforehand.

While this is advantageous in general, parametric models are useful techniques in their own right. Especially if one happens to have additional knowledge about the problem, it would be unwise not to take advantage of it. For instance, it might be the case that the major properties of the data are described by a combination of a small set of linear independent basis functions $\{\phi_1(\cdot), \ldots, \phi_n(\cdot)\}$. Or one may want to correct the data for some (e.g. linear) trends. Secondly it also may be the case that the user wants to have an *understandable* model, without sacrificing

12. Parts of this section have been published as [Smola et al., 1998c].

accuracy. Many people in life sciences tend to have a preference for linear models. This may be some motivation to construct *semiparametric* models, which are both easy to understand (for the parametric part) and perform well (often due to the nonparametric term). For more advocacy on semiparametric models see [Bickel et al., 1994].

A common approach is to fit the data with the parametric model and train the nonparametric add–on on the errors of the parametric part, i.e. fit the nonparametric part to the errors. It is shown in Sec. 3.8.4 that this is useful only in a very restricted situation. In general, it is impossible to find the best model amongst a given class for different cost functions by doing so. The better way is, to solve a convex optimization problem like in standard SV machines, however, with a different set of admissible functions

$$f(x) = \langle w, \psi(x) \rangle + \sum_{i=1}^{n} \beta_i \phi_i(x). \tag{3.92}$$

3.8.2 Theoretical Basis

In the following this setting will be treated more formally. For the sake of simplicity, the exposition is restricted to the case of SV regression with the ε–insensitive loss function (1.6). Extensions are quite straightforward and follow directly from the results of chapter 2.

Some basic notions on reproducing kernel Hilbert spaces are useful in this context. The following definition is adapted from [Aronszajn, 1950].

Definition 3.13 Reproducing Kernel Hilbert Space
Let F be a class of functions defined in H forming a Hilbert space (complex or real). The function $k(x, y)$ in H is called a *reproducing kernel* of F if

1. For every y, $k(x, y)$ as function of x belongs to F.
2. The *reproducing property*: for every $y \in E$ and every $f \in F$,

$$f(y) = \langle f(\cdot), k(\cdot, y) \rangle_H \tag{3.93}$$

The function k is unique for given F, H, in fact, it is the SV kernel introduced previously. As the regularization operator P generates a Hilbert space by $\langle P(\cdot), P(\cdot) \rangle$, one can associate a corresponding kernel k (i.e. the Green's function of P^*P) with it.

Note that the regularized risk approach can also be dealt with, by using Reproducing Kernel Hilbert Spaces (RKHS), which may lead to sometimes more elegant exposition of the subject (cf. [Aronszajn, 1944, 1950, Kimeldorf and Wahba, 1971, Micchelli, 1986, Saitoh, 1988, Wahba, 1990, Girosi, 1998, Schölkopf, 1997]).

The semiparametric approach now is motivated by the following theorem (which holds for arbitrary cost functions):

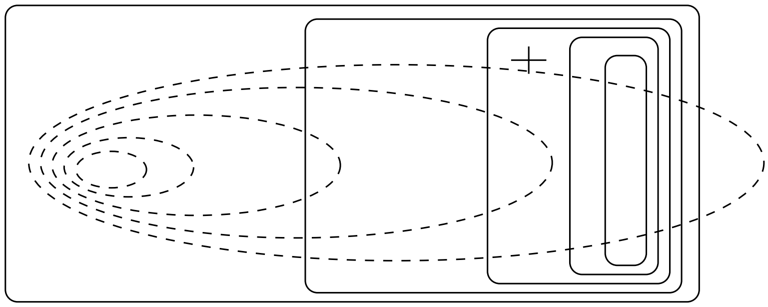

Figure 3.7 Two different nested subsets (solid and dotted lines) of hypotheses and the optimal model (+) in the realizable case.

Theorem 3.14 Kimeldorf and Wahba [1971], Cox and O'Sullivan [1990]

Let H be a reproducing kernel Hilbert space of real valued functions on $\mathcal{X}$ with reproducing kernel k. Denote by X the training set, and let $\Phi := \{\phi_1, \ldots, \phi_n\}, n \in \mathbb{N}$ be a set of functions on $\mathcal{X}$ such that the matrix $\Phi_{\nu j} := \phi_\nu(x_j)$ has maximal rank. Then for

$$\hat{f} = \operatorname*{argmin}_{f \in \mathrm{span}\,(\Phi)+h, h\in H} \left[\frac{1}{m}\sum_{i=1}^{m} c(x_i, y_i, f(x_i)) + \lambda\|h\|_H^2\right] \tag{3.94}$$

one has

$$\hat{f} \in \mathrm{span}\,(\Phi \cup \{k(x_1, .), \ldots, k(x_m, .)\}). \tag{3.95}$$

Thus an expansion of type (3.92) is equivalent to (3.95) and moreover optimal in the above setting. Effectively this result already contains the subsequent reasoning. Hence one should consider it mainly as an approach to include semiparametric regularization functionals in a convex programming approach. Moreover it is also an extended solution to the problem created in Sec. 3.7, where for conditionally positive definite kernels of order $p \geq 1$ the final estimate may not contain polynomial contributions up to degree p.

Keeping $Q[f] = \frac{1}{2}\|w\|^2$, i.e. the standard regularizer (1.23), means, that there exist functions $\phi_1(\cdot), \ldots, \phi_n(\cdot)$ whose contribution is not regularized at all. If n is sufficiently smaller than m this need not be a major concern, as the VC–dimension of this additional class of linear models is n, hence the overall capacity control will still work, provided the nonparametric part is restricted sufficiently.

Figure 3.7 explains the effect of choosing a different structure in detail. Observe that the optimal model is already contained in much a smaller (in the diagram, size corresponds to the capacity of a subset) subset of the structure with solid lines than in the structure denoted by the dotted lines. Hence prior knowledge in choosing the structure, i.e. making a "luckier" guess, can have a large effect on generalization bounds and performance. Chapter 8 contains considerations on how to control the capacity of such mixed models as semiparametric SV machines.

3.8.3 The Algorithm

Formulating the optimization equations for this particular expansion (3.92), the ε–insensitive loss function, and introducing kernels one arrives at the following primal optimization problem:

$$\begin{array}{ll} \text{maximize} & \frac{\lambda}{2}\|w\|^2 + \sum_{i=1}^{m} \xi_i + \xi_i^* \\ \text{subject to} & \left\{ \begin{array}{lcl} \langle w, \psi(x_i)\rangle + \sum_{j=1}^{n} \beta_j \phi_j(x_i) - y_i & \leq & \epsilon + \xi_i^* \\ y_i - \langle w, \psi(x_i)\rangle - \sum_{j=1}^{n} \beta_j \phi_j(x_i) & \leq & \epsilon + \xi_i \\ \xi_i, \xi_i^* & \geq & 0 \end{array} \right. \end{array} \tag{3.96}$$

Solving (3.96) for its Wolfe dual yields

$$\begin{array}{ll} \text{maximize} & \left\{ \begin{array}{l} -\frac{1}{2} \sum_{i,j=1}^{m} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) k(x_i, x_j) \\ -\varepsilon \sum_{i=1}^{m} (\alpha_i + \alpha_i^*) + \sum_{i=1}^{m} y_i (\alpha_i - \alpha_i^*) \end{array} \right. \\ \text{subject to} & \left\{ \begin{array}{lcl} \sum_{i=1}^{m} (\alpha_i - \alpha_i^*) \phi_j(x_i) & = & 0 \text{ for all } 1 \leq j \leq n \\ \alpha_i, \alpha_i^* & \in & [0, 1/\lambda] \end{array} \right. \end{array} \tag{3.97}$$

Note the similarity to the standard SV regression model. The objective function and the box constraints on the Lagrange multipliers α_i, α_i^* remain unchanged. The only modification comes from the additional unregularized basis functions. Instead of a single (constant) function $b \cdot 1$ in the standard SV case, one now has an expansion in the basis $\beta_i \phi_i(\cdot)$. This gives rise to n constraints instead of one. Finally f can be found as

$$f(x) = \sum_{i=1}^{m} (\alpha_i - \alpha_i^*) k(x_i, x) + \sum_{i=1}^{n} \beta_i \phi_i(x) \quad \text{since} \quad w = \sum_{i=1}^{m} (\alpha_i - \alpha_i^*) \psi(x_i). \tag{3.98}$$

The only difficulty remaining is how to determine β_i. This can be done by exploiting the Karush–Kuhn–Tucker optimality conditions (by a reasoning much similar to the one that led to (1.14)), or much more easily, by using an interior point optimization code [Vanderbei, 1994] (see also Chapter 4). In the latter case the variables β_i can be obtained as the dual variables of the dual (dual dual = primal) optimization problem (3.97) as a by–product of the optimization process. This is also, how these variables were obtained in the experiments described below.

3.8.4 Why Backfitting is not sufficient

One might think that the approach presented above is quite unnecessary and overly complicated for semiparametric modelling. In fact, one could try to fit the data to the parametric model first, and then fit the nonparametric part to the residuals, an approach called backfitting. In most cases, however, this does not lead to the minimum of the regularized risk functional. It can be shown at a simple example.

Take a SV machine with linear kernel (i.e. $k(x, x') = \langle x, x' \rangle$) in one dimension and a constant term as parametric part (i.e. $f(x) = wx + \beta$). This is one of the simplest semiparametric settings possible. Now suppose the data was generated by

$$y_i = x_i \quad \text{where} \quad x_i \geq 1 \tag{3.99}$$

without noise. Clearly then also $y_i \geq 1$ for all i. By construction the best overall fit of the pair (β, w) will be arbitrarily close to $(0, 1)$ if the regularization parameter λ is chosen sufficiently small.

For backfitting one first carries out the parametric fit to find a constant β minimizing the term $\sum_{i=1}^{m} c(y_i - \beta)$. Depending on the chosen cost function $c(\cdot)$, β will be the mean (L_2–error), the median (L_1–error), a trimmed mean (ε–insensitive loss), etc., of the set $\{y_1, \ldots, y_m\}$. As all $y_i \geq 1$ also $\beta \geq 1$, which is surely not the optimal solution of the overall problem, as in the latter case β would be close to 0, as seen above. Hence, not even in the simplest of all settings, backfitting minimizes the regularized risk functional, thus one cannot expect the latter to happen in the more complex case either. There exists only one case in which backfitting would suffice, namely if the function spaces spanned by the kernel expansion $\{k(x_i, \cdot)\}$ and $\{\phi_i(\cdot)\}$ were orthogonal. Consequently in general one has to jointly solve for both the parametric and the nonparametric part.

3.8.5 Experiments

What follows is a proof of concept and an analysis of the properties of the new algorithm. The function analyzed is a modification of the *Mexican hat* function, namely

$$f(x) = \sin x + \operatorname{sinc}(2(x - 5)). \tag{3.100}$$

Our semiparametric assumption consists in using the basis functions $1, \sin x$, and $\cos x$ (and for a sanity check also $1, \sin 2x, \cos 2x$). Data is generated by an additive noise process, i.e. $y_i = f(x_i) + \xi_i$, where ξ_i is additive noise. For the experiments we choose Gaussian rbf–kernels with width $\sigma = 1/4$, normalized to maximum output 1. The noise is uniform with standard deviation 0.2, the cost function $|\cdot|_\varepsilon$ with $\varepsilon = 0.05$. Unless stated otherwise averaging is done over 100 datasets with 50 samples each. The x_i are drawn uniformly from the interval $[0, 10]$. L_1 and L_2 errors are computed on the interval $[0, 10]$ with uniform measure.

Figure 3.8 shows the function and typical predictions in the nonparametric, semiparametric, and parametric setting. Note the different length scales of $\sin x$ and $\operatorname{sinc} 2x$. For convenience the functions are shifted by an offset of 2 and 4 respectively. The regularization constant for the estimate was set to $\lambda = 2$. Observe that the semiparametric model picks up the characteristic *wiggles* of the original function, thus is able to generalize better than the standard SV machine.

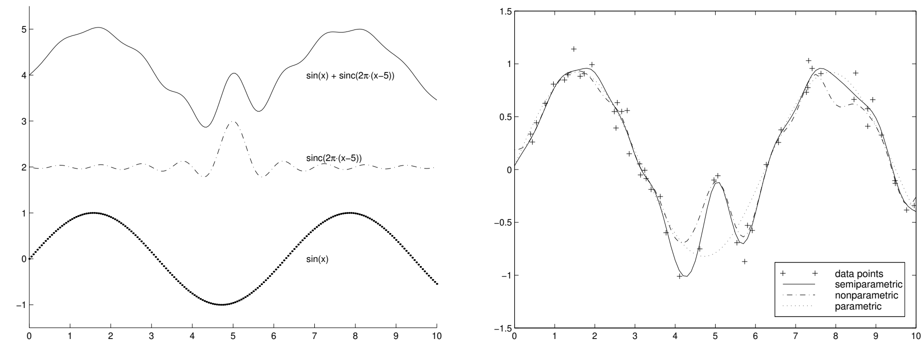

Figure 3.8 Left: Basis functions used in the toy example. Right: Training data denoted by '+', nonparametric (dash–dotted line), semiparametric (solid line), and parametric regression (dots).

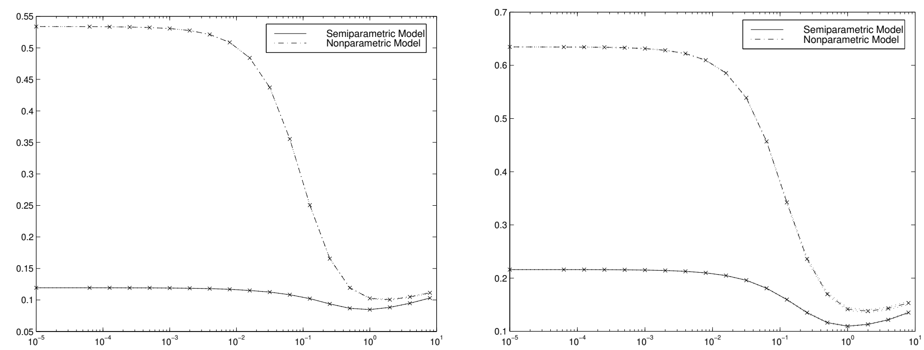

Figure 3.9 L_1 error (left) and L_2 error (right) of the nonparametric / semiparametric regression computed on the interval $[0, 10]$ vs. the regularization strength $1/\lambda$.

Figure 3.9 shows that the generalization performance is better in the semiparametric case. Note that in both error measures the semiparametric model consistently outperforms the nonparametric one.

The length of the weight vector of the kernel expansion $\|w\|$ is displayed in Fig. 3.10. Note that $\|w\|$, controlling the capacity of that part of the function, belonging to the kernel expansion, is smaller (for practical choices of the regularization term) in the semiparametric than in the nonparametric model. If this difference is sufficiently large, the overall capacity of the resulting model is smaller in the semiparametric approach.

Finally figure 3.11 shows the quality of the estimate of the nonparametric part (note that for decreasing regularization strength the contributions of the parametric model converge to their least mean squares values, i.e. $0.8702 \sin x + 0.0384 \cos x$). Training set size was $m = 50$. Note the small variation of the estimate. Also note that even in the parametric case ($1/\lambda = C = 0$) neither the coefficient for $\sin x$ converges to 1, nor does the corresponding term for $\cos x$ converge to 0. This is due to the additional frequency contributions of $\operatorname{sinc} 2x$.

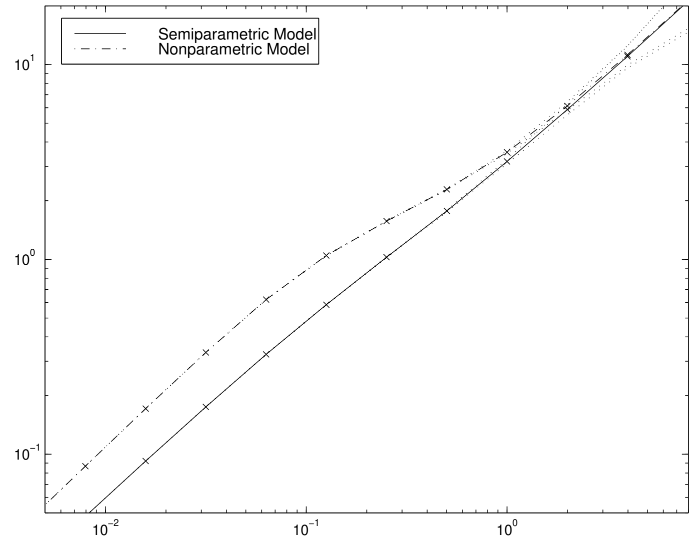

Figure 3.10 Length of the vector w in *feature space* $(\sum_{i,j}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(x_i, x_j))^{1/2}$ vs. regularization strength. As before dotted lines indicate the variance.

	Nonparam.	Semiparam. $\sin x, \cos x, 1$	Semiparam. $\sin 2x, \cos 2x, 1$
L_1 error	0.1263 ± 0.0064 (12)	0.0887 ± 0.0018 (82)	0.1267 ± 0.0064 (6)
L_2 error	0.1760 ± 0.0097 (12)	0.1197 ± 0.0046 (82)	0.1864 ± 0.0124 (6)

Table 3.1 L_1 and L_2 error for model selection by 10–fold crossvalidation. The number in parentheses denotes the number of trials in which the corresponding model was the best among the three models.

To make a more realistic comparison, model selection (in this case how to determine $1/\lambda$) was carried out by 10–fold cross validation for both algorithms independently for all 100 datasets. Table 3.1 shows generalization performance for both a nonparametric model, a correctly chosen and an incorrectly chosen semiparametric model. The *correct* semiparametric model $(\sin x, \cos x, 1)$ outperforms the nonparametric model by at least 30%, and has significantly smaller variance, whereas the wrongly chosen semiparametric model $(\sin 2x, \cos 2x, 1)$ gives performance comparable to the nonparametric one, in fact, no significant performance degradation was noticeable.

The experiments indicate that cases, in which prior knowledge exists on the type of functions to be used, will benefit from semiparametric modelling. Future experiments have to show how much can be gained in real world examples.

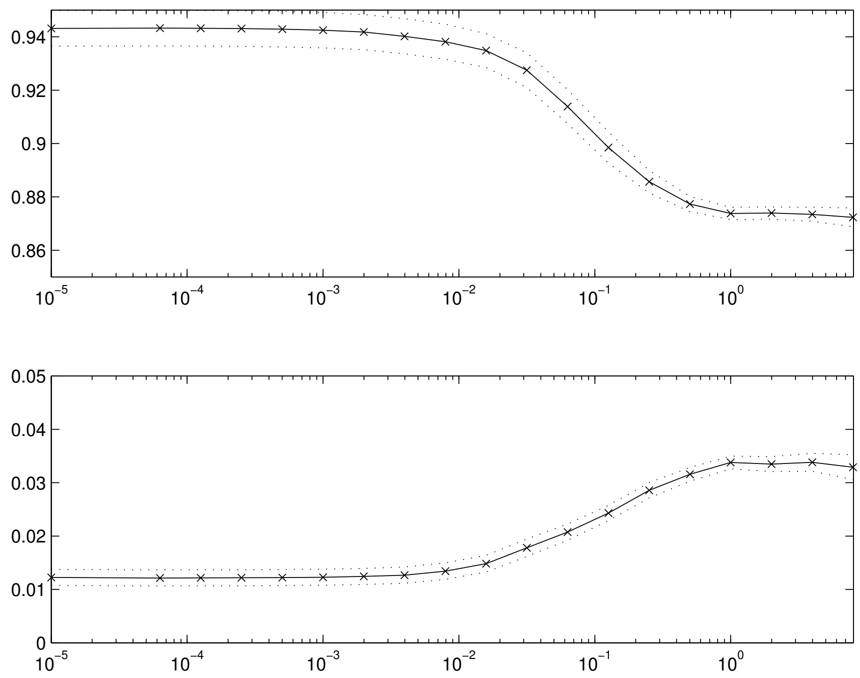

Figure 3.11 Estimate of the parameters for $\sin x$ (top picture) and $\cos x$ (bottom picture) in the semiparametric model vs. regularization strength $1/\lambda$. The dotted lines above and below show the variation of the estimate given by its variance.

3.9 ℓ_p^m Norms and Other Extensions

The last modification of regularization functionals are semiparametric regularizers based on functionals of type (1.24) and (1.26). These will be briefly reconsidered with the concept of kernel expansions in mind. It leads to slightly modified versions of the previously discussed algorithms (e.g. linear programming semiparametric machines).

3.9.1 Linear Programming Regularization (ℓ_1^m)

For $Q[f] = \sum_i |\alpha_i|$ the regularized risk minimization problem can be rewritten as

$$\begin{array}{ll} \text{minimize} & R_{\text{reg}}[f] = \lambda \sum_{i=1}^{m} |\alpha_i| + \sum_{i=1}^{m} (\tilde{c}(\xi_i) + \tilde{c}(\xi_i^*)) \\ \text{subject to} & \left\{ \begin{array}{lll} y_i - \sum_{j=1}^{m} \alpha_j k(x_j, x_i) - \sum_{j=1}^{n} \phi_j(x_i) - b & \leq & \varepsilon + \xi_i \\ \sum_{j=1}^{m} \alpha_i k(x_j, x_i) + \sum_{j=1}^{n} \phi_j(x_i) + b - y_i & \leq & \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* & \geq & 0 \end{array} \right. \end{array} \tag{3.101}$$

Besides replacing α_i with $\alpha_i - \alpha_i^*$, $|\alpha_i|$ with $\alpha_i + \alpha_i^*$, and requiring $\alpha_i, \alpha_i^* \geq 0$ there is hardly anything that can be done to render the problem computationally more feasible — the constraints are already linear. Moreover most optimization software can deal with problems of this type efficiently.

3.9.2 Weight Decay (ℓ_2^m)

What remains is to consider problems of the type

$$\begin{array}{ll} \text{minimize} & R_{\text{reg}}[f] = \frac{\lambda}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j D_{ij} + \sum_{i=1}^{m} (\tilde{c}(\xi_i) + \tilde{c}(\xi_i^*)) \\ \text{subject to} & \left\{ \begin{array}{lll} y_i - \sum_{j=1}^{m} \alpha_j k(x_j, x_i) - \sum_{j=1}^{n} \phi_j(x_i) - b & \leq & \varepsilon + \xi_i \\ \sum_{j=1}^{m} \alpha_i k(x_j, x_i) + \sum_{j=1}^{n} \phi_j(x_i) + b - y_i & \leq & \varepsilon + \xi_i^* \\ \xi_i, \xi_i^* & \geq & 0 \end{array} \right. \end{array} \tag{3.102}$$

resulting in weight decay for $D = \mathbf{1}$. This setting leads to optimization problems as stated in section 3.2.1 in (3.16). As already mentioned previously, sparsity is lost due to the unmatched regularization term D and kernel k.

3.9.3 Mixed Semiparametric Regularizers

Eq. (1.23) in combination with (3.92) gives rise to the question whether not also mixed regularization functionals would be possible. Indeed one can construct the following variants, which is a mixture between linear and quadratic regularizers, i.e.

$$Q[f] = \frac{1}{2}\|w\|^2 + \sum_{i=1}^{n} |\beta_i|. \tag{3.103}$$

The equation above is essentially the SV estimation model where an additional linear regularization term has been added for the parametric part. In this case the constraints of the optimization problem (3.97) change into

$$\begin{array}{rclcll} -1 & \leq & \sum_{i=1}^{m}(\alpha_i - \alpha_i^*)\phi_j(x_i) & \leq & 1 & \text{for all } 1 \leq j \leq n \\ & & \alpha_i, \alpha_i^* & \in & [0, 1/\lambda] & \end{array} \tag{3.104}$$

and the variables β_i are obtained as the dual variables of the constraints as already mentioned before for several similar cases. Finally one could reverse the setting to obtain a regularizer like

$$Q[f] = \sum_{i=1}^{m} |\alpha_i - \alpha_i^*| + \frac{1}{2} \sum_{i,j=1}^{n} \beta_i \beta_j M_{ij} \tag{3.105}$$

for some positive semidefinite matrix M. Note that (3.105) can be reduced to the case of (3.103) by renaming variables accordingly and a proper choice of M.

The proposed regularizers are a simple extension of existing methods like Basis Pursuit [Chen et al., 1995] or Linear Programming for classification (e.g. [Frieß and Harrison, 1998]). The common basic idea is to have two different sets of basis functions that are regularized differently, or where a subset may not be regularized at all. This is an efficient way of encoding prior knowledge or the preference of the user, as the emphasis will be on the functions with little or no regularization at all.

3.10 Summing Up

A connection between SV kernels and regularization operators has been established, which can provide one key to understanding why SV machines have been found to exhibit high generalization ability. In particular for the common choices of kernels, the mapping into feature space is not arbitrary but corresponds to good regularization operators (see sections 3.3.1, 3.3.2 and 3.3.4). For kernels, however, where this is not the case, SV machines may show poor performance (section 3.3.3). This will become more obvious in Sec. 8 where, building on the results of the current chapter, the eigenspectrum of integral operators is connected with generalization bounds of the corresponding SV machines.

Capacity control is one of the strengths of SV machines; however, this does not mean that the structure of the learning machine, i.e. the choice of a suitable kernel for a given task, should be disregarded. On the contrary, the rather general class of admissible SV kernels should be seen as another strength, provided that we have a means of choosing the right kernel. The link to regularization theory can thus be seen as a tool for constructing the structure consisting of sets of functions in which the SV machine (approximately) performs structural risk minimization (e.g. [Vapnik, 1995]), possibly in a data dependent manner. In other words it allows to choose an appropriate kernel given the data and the problem specific knowledge.

A simple consequence of the proposed link is a Bayesian interpretation of Support Vector machines. In this case the choice of a special kernel can be regarded as a prior on the hypothesis space with $P[f] \propto \exp(-\lambda \|Pf\|^2)$.

It should be clear by now that the setting of Tikhonov and Arsenin [1977] is a very powerful, but surely not the only one. However, a theorem on vector valued regularization operators showed that already under quite generic conditions on the isotropy of the space of target values only scalar operators are possible, thus an extended version of their approach is the only admissible one.

The regularization framework has made it possible to extend the class of admissible kernels to those defined by conditionally positive definite functions — a class of kernels that do not necessarily have to satisfy Mercer's condition.

Finally a closer consideration of the nullspace of regularization functionals $Q[f]$ led to the aforementioned semiparametric models. Its roots lie in the representer theorem (Th. 3.14) proposed and explored in the context of smoothing splines by Kimeldorf and Wahba [1971]. In fact, the SV expansion is a direct result of it.

Moreover the semiparametric setting solves a problem created by the use of *conditionally* positive definite kernels of order p (see sec. 3.7). There one had to exclude polynomials of order lower than p. Hence, to cope with this effect, one has to add polynomials back in "manually." The semiparametric approach presents a way of doing that. Another application of semiparametric models besides the conventional approach of treating the nonparametric part as *nuisance parameters* [Bickel et al., 1994] is the domain of hypothesis testing, e.g. to test whether a parametric model fits the data sufficiently well. This can be achieved in the framework of structural risk minimization [Vapnik, 1995] — given the different models (nonparametric vs. semiparametric vs. parametric) one can evaluate the bounds on the expected risk and then choose the model with the best bound.

3.11 Appendix: A worked through example

This section shows how to construct a Support Vector kernel for the operator

$$\|Pf\|_2^2 := \|f\|_2^2 + \sum_{i=1}^{n} \|\partial_{x_i} f\|_2^2. \tag{3.106}$$

This example is taken from [Girosi et al., 1993] and used to illustrate the reasoning of this chapter in detail. For ease of notation assume $f : \mathbb{R} \to \mathbb{R}$.

A corresponding representation of P^*P in Fourier space yields

$$\|Pf\|_2^2 = \int_{\mathbb{R}} d\omega |\tilde{f}(\omega)|^2 (1 + \omega^2) \tag{3.107}$$

or equivalently (cf. section 3.3, eq. (3.23)) $P(\omega) = \frac{1}{1+\omega^2}$. In order to satisfy the self consistency condition (3.17), one has to compute the inverse Fourier transform of $P(\omega)$ to obtain the Green's functions of P^*P (cf. (3.24)). This leads to a kernel of the form $k(x, x') = e^{-|x-x'|}$.

An expansion in terms of the *Laplacian* kernel (not to be confused with the Laplacian distribution of the same shape), however, may not always be desirable as it is by far less smooth than when using a Gaussian kernel (see Fig. 3.12).

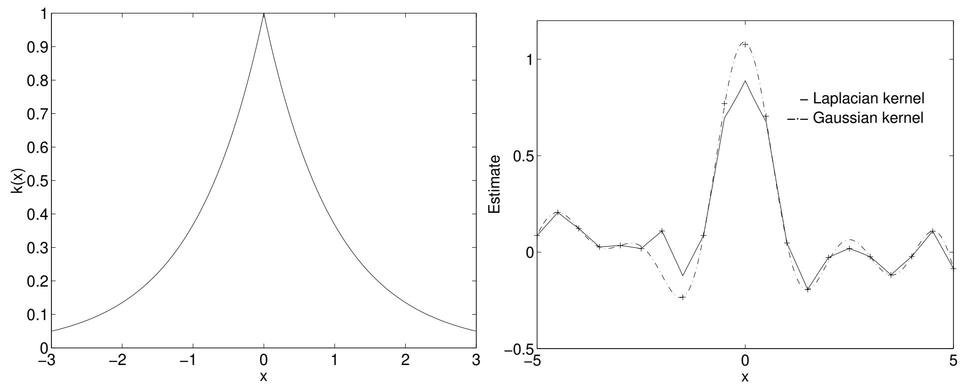

Figure 3.12 Left: *Laplacian* kernel. Right: Regression with a Gaussian ($\sigma = 1$) and a Laplacian kernel (kernel width 2) of the data shown in Fig. 3.4.

4 Optimization Algorithms

Due to the size of the optimization problems arising from the SV setting one has to pay special attention as to how these problems can be solved efficiently. The current chapter presents three algorithms for solving these convex programming problems and gives an outline on how these can be combined and under which conditions which algorithm should be preferred. This is quite an ambitious task and there is absolutely no claim that the conclusions are optimal. It should be considered as a starting point for the development of efficient algorithms. Finally the present chapter differs from the previous considerations insofar as it takes the setting of pattern classification[1] into account (as most of the conclusions drawn here apply to pattern recognition, too, and the latter is still the testbed of choice for new optimization algorithms).

Roadmap

Basic properties of convex optimization problems are reviewed in the first section. This lays the foundations of the subsequent development of algorithms. After a description of basic facts regarding the optimal solutions of such optimization problems, the primal–dual formulation of mathematical programming problems, a central aspect to constrained optimization problems, is introduced. Useful tricks which may be applied to any type of optimization algorithm conclude this first overview over optimization algorithms.

A direct consequence of the primal–dual reasoning are so called interior point algorithms. These proceed by finding a set of variables that is both primal and dual feasible — an attempt that is quite contrary to common intuition which would attempt to minimize the dual objective function directly without much further ado. Interior point algorithms are some of the most flexible ones, especially regarding the extensions of the basic SV algorithm to allow for optimal choices of ε, general convex cost functions, and semiparametric modelling. Special considerations concerning SV regression and numerical experiments on the computational (sample) complexity of such algorithms are given.

As interior point methods are flexible and precise but computationally expensive

1. For the optimization equations see [Boser et al., 1992, Cortes and Vapnik, 1995, Vapnik, 1995, Schölkopf, 1997].

methods for large datasets, working set methods are looked at in the subsequent section. After deriving the optimization equations for the restricted set of variables it is shown how subset selection rules can be seen from the point of primal and dual objective functions. Common selection rules are described with an analysis of their significance.

The last section is devoted to sequential minimal optimization (SMO) which could be seen as a special case of the previously described working set algorithms. Besides explicit optimization equations for a subset of two variables, detailed instruction for implementation (including pseudocode) are given. Minor extensions such as pattern dependent regularization complement the modifications of the algorithm that was devised initially for pattern recognition.

4.1 Basic Notions and Duality Theory

Most optimization algorithms rely, in a more or less direct manner, on results from duality theory in convex optimization. Although some of the basic ideas already have been mentioned in section 1.1.1, these issues, for the sake of convenience, will be briefly reviewed. These constitute the core results one needs in order to derive interior point and subset selection algorithms. Details and proofs can be found in textbooks on optimization, e.g. [Fletcher, 1989, Vanderbei, 1997].

4.1.1 Properties of the Optimal Solution

The purpose of the properties, stated in the following, is to find a characterization of the optimal solution in terms of some optimality conditions, which, in turn can be exploited to compute the optimum.

Uniqueness Every strictly convex constrained optimization problem has a unique or no solution.[2] This means that SVs are not plagued with the problem of *local minima* as Neural Networks are.[3] The uniqueness property can be seen as follows: assume that there exist two points, say x_1 and x_2 where the minimum of the (primal

2. The latter might occur e.g. if the feasible region is noncompact in situations like $\min_x e^{-x}$. Thanks to Olvi Mangasarian for pointing this out.
3. For large and noisy problems (e.g. 100.000 patterns and more) it is quite impossible to find the *exact* minimum of the optimization equations. This is due to the fact that one has to use subset selection algorithms, hence joint optimization over the training set is impossible, and the global optimum is only approached up to a certain precision, say 0.001. Neural Networks, however, have the additional problem that one can not even be sure that it is the *global* minimum one is approaching, as there are exponentially many *local* minima [Minsky and Papert, 1969]. Moreover, no statement can be made about the absolute quality of the solution, i.e. the maximum distance of the current set of variables to the *optimal* solution. However, all this reasoning is valid only in the case of *convex* cost functions.

objective, i.e. target) function f(x), is obtained. As the problem is strictly convex, all points $x_\lambda := \lambda x_1 + (1-\lambda)x_2$ are feasible, i.e. satisfy the constraints on the manifold of the solution. Moreover $f(x_\lambda) < \lambda f(x_1) + (1-\lambda)f(x_2)$ for $\lambda \in (0,1)$ due to the convexity. This is a contradiction to the initial assumption that $f(x_1) = f(x_2)$ are both minima of the constrained optimization problem. The same reasoning also shows that there exist no local minima.

Lagrange Function The Lagrange function is given by the primal objective function (the one that should be minimized) minus the sum of all products between constraints and corresponding Lagrange multipliers (cf. e.g. [Goldstein, 1986, Fletcher, 1989]). Optimization can be seen as minimzation of the Lagrange function wrt. the primal variables or maximization wrt. the Lagrange multipliers, i.e. dual variables. Thus it has a saddle point at the optimal solution in terms of the primal and dual variables. Usually the Lagrange function is only used as a theoretical device to derive the dual objective function (cf. chapter 1.1.1).

Dual Objective Function It is derived by minimizing the Lagrange function with respect to the primal variables and subsequent elimination of the latter. Hence it can be written solely in terms of the dual variables (i.e. Lagrange multipliers) and leads to the dual *maximization* problem.

Duality Gap For both feasible primal and dual variables the primal objective function (of a convex minimization problem) is always greater or equal than the dual objective function. Equality is obtained if and only if we are at the optimal solution. Thus the duality gap is a measure how close (in terms of the objective function) the current set of variables is already to the optimal solution. It follows directly from the saddlepoint property of the Lagrange function.

Karush–Kuhn–Tucker (KKT) conditions A set of primal and dual variables that is both feasible and satisfies the KKT conditions is the optimal solution (i.e. constraint · dual variable = 0). The sum of the violated KKT terms determines (by construction of the Lagrange function) exactly the size of the duality gap. This allows to compute the latter quite easily.
A simple intuition to see why "constraint · dual variable = 0" can be found in the fact that for violated constraints the dual variable could be increased arbitrarily, thus rendering the Lagrange function arbitrarily large. This, however, is in contradition to the saddlepoint property.
Consider a simple example: a box containing a ball subject to the forces of gravity. Only at the faces of the box (i.e. the constraints) where the ball (i.e. the variables) touches the box (the domain of feasible regions) forces may be applied (i.e. yield nonzero Lagrange multipliers). The amount is given by the projections of the gradient of the objective function (potential energy) onto the constraints (faces of the box).

The above mentioned results allow to find efficient solutions of the convex optimization problem. In particular, a primal–dual formulation is quite useful in this context.

4.1.2 Primal–Dual Formulation

In order to avoid tedious notation consider a slightly more general problem. The results will be specialized to the SV case in the end.

$$\begin{array}{rl} \text{minimize} & \frac{1}{2}q(\alpha) + \langle c, \alpha \rangle \\ \text{subject to} & A\alpha = b \\ & l \leq \alpha \leq u \end{array} \tag{4.1}$$

with $c, \alpha, l, u \in \mathbb{R}^m$, $A \in \mathbb{R}^{n \cdot m}$, $b \in \mathbb{R}^n$, the inequalities between vectors holding componentwise, and $q(\alpha)$ being a convex function of α (in the feasible region, i.e. in the region where the constraints are satisfied). In the case of SV pattern recognition (with soft margin) $q(\alpha) = \frac{1}{2}(\alpha - \alpha^*)^\top K(\alpha - \alpha^*)$ with $K_{ij} = y_i y_j k(x_i, x_j)$, $l = 0, u = C = \frac{1}{\lambda}, b = 0$, etc.

Now add slack variables to get rid of all inequalities but the positivity constraints. This yields:

$$\begin{array}{rl} \text{minimize} & \frac{1}{2}q(\alpha) + \langle c, \alpha \rangle \\ \text{subject to} & A\alpha = b,\ \alpha - g = l,\ \alpha + t = u \\ & g, t \geq 0,\ \alpha \text{ free} \end{array} \tag{4.2}$$

The Wolfe dual of (4.2) is (cf. section 4.6.1)

$$\begin{array}{rl} \text{maximize} & \frac{1}{2}\left(q(\alpha) - \langle \partial_\alpha q(\alpha), \alpha \rangle\right) + \langle b, y \rangle + \langle l, z \rangle - \langle u, s \rangle \\ \text{subject to} & \frac{1}{2}\partial_\alpha q(\alpha) + c - (Ay)^\top + s = z \\ & s, z \geq 0,\ y \text{ free} \end{array} \tag{4.3}$$

Moreover, one gets the Karush Kuhn Tucker (KKT) conditions, namely

$$\left.\begin{array}{rcl} g_i z_i & = & 0 \\ s_i t_i & = & 0 \end{array}\right\} \text{ for all } i \in \{1, \ldots, m\}. \tag{4.4}$$

As stated before, a necessary and sufficient condition for the optimal solution to be found is that the primal / dual variables satisfy both the feasibility conditions of (4.2) and (4.3) and the KKT conditions (4.4). Interior Point path following algorithms work by solving the system of equations iteratively.

4.1.3 Useful Tricks

The following tricks apply to all algorithms described subsequently and (despite their simplicity) can be used to speed up training significantly or to monitor convergence in a reliable fashion.

Training with Different Regularization Parameters For several reasons (model selection, controlling the number of support vectors, etc.) it may happen that one has to train a SV machine with different regularization parameters C, but otherwise rather identical settings. If the parameters C_i are not too different, it is advantageous to use the *rescaled* values of the Lagrange multipliers (i.e. α_i, α_i^*) as a starting point for the new optimization problem. Rescaling is necessary to satisfy the modified constraints. Thus one gets

$$\alpha_{\text{new}} = \frac{C_{\text{new}}}{C_{\text{old}}}\alpha_{\text{old}} \quad \text{and analogously} \quad b_{\text{new}} = \frac{C_{\text{new}}}{C_{\text{old}}} b_{\text{old}}. \tag{4.5}$$

Assuming that the (dominant) convex part $q(\alpha)$ of the primal objective is quadratic, the latter scales with $\frac{C^2_{\text{new}}}{C^2_{\text{old}}}$, which is faster than the linear part. However, as the linear term dominates the objective function (one obtains negative values in practice, the convex term, however can only be nonnegative), the rescaled values are still a better starting point than $\alpha = 0$. In practice a speedup of approximately 95% of the overall training time can be observed when using the sequential minimization algorithm (cf. fig. 6.1).
A similar reasoning can be applied when retraining with the same regularization parameter but different (yet similar) width parameters of the kernel function. See [Cristianini et al., 1998] for details thereon in a different context.

Monitoring Convergence via the Feasibility Gap In the case of both primal and dual feasible variables the following connection between primal and dual objective function holds:

$$\text{Dual Objective} = \text{Primal Objective} - \sum_i (g_i z_i + s_i t_i) \tag{4.6}$$

This can be seen immediately by the construction of the Lagrange function. In the case of SV pattern recognition this translates into

$$\sum_i (g_i z_i + s_i t_i) = \sum_i \left[\max(0, f(x_i)y_i - 1)\alpha_i - \min(0, f(x_i)y_i - 1)(C - \alpha_i)\right]. \tag{4.7}$$

In regression estimation (with the ε–insensitive loss function) one has

$$\sum_i g_i z_i + s_i t_i = \sum_i \begin{bmatrix} +\max(0, f(x_i) - (y_i + \epsilon_i))(C - \alpha_i^*) \\ -\min(0, f(x_i) - (y_i + \epsilon_i))\alpha_i^* \\ +\min(0, (y_i - \epsilon_i^*) - f(x_i))(C - \alpha_i) \\ -\max(0, (y_i - \epsilon_i^*) - f(x_i))\alpha_i \end{bmatrix}. \tag{4.8}$$

Thus convergence with respect to the optimal solution can be expressed in terms of the duality gap. An effective stopping rule is to require

$$\frac{\sum_i g_i z_i + s_i t_i}{|\text{Primal Objective}| + 1} \leq \epsilon_{\text{tol}} \tag{4.9}$$

for some precision ϵ_{tol}. This condition is very much in the spirit of primal dual interior point path following algorithms, where convergence is measured in terms of the number of significant figures (which would be the decimal logarithm of (4.9)), a convention that will also be adopted in the subsequent parts of this exposition.

4.2 Interior Point Algorithms

In a nutshell the idea of those algorithms is to solve (4.2) and (4.3) simultaneously. This is achieved by seeking a set of variables that is both primal and dual feasible and satisfies the KKT conditions. The latter are only gradually enforced, while iteratively converging to a feasible solution. The duality gap between primal and dual objective function is used to determine the quality of the current set of variables. The special flavour of algorithm adopted is a primal–dual path–following one as described by Vanderbei [1994] (the present chapter largely preserves his notation).

4.2.1 Solving the Equations

Hence one tries not to solve (4.4) as it is, but a modified version instead for some $\mu > 0$ in the first place, and decrease μ while iterating.

$$\left.\begin{array}{rcl} g_i z_i & = & \mu \\ s_i t_i & = & \mu \end{array}\right\} \text{ for all } i \in \{1, \dots m\}. \tag{4.10}$$

Simultaneously the feasibility constraints are kept satisfied (or at least improved while iterating). Still it is rather difficult to solve the nonlinear system of equations (4.2), (4.3), and (4.10) exactly. However one is not interested in obtaining the exact solution — instead the aim is to find a somewhat more feasible solution for a given μ, then decrease μ, and keep on iterating. This can be done by linearizing the above system (i.e. expanding variables x into $x + \Delta x$) and solving the resulting equations by a predictor–corrector approach until the duality gap is small enough. This means that one will solve the linearized system for the variables in Δ once — this is the *predictor* step — then substitute these variables back into the quadratic terms in Δ and solve the linearized system again (*corrector*). The advantage is that one will get approximately equal performance as by solving the quadratic system directly, provided that the terms in Δ^2 are small enough. The linearized system reads as follows:

$$\begin{array}{rcl} A(\alpha + \Delta\alpha) & = & b \\ \alpha + \Delta\alpha - g - \Delta g & = & l \\ \alpha + \Delta\alpha + t + \Delta t & = & u \\ c + \frac{1}{2}\partial_\alpha q(\alpha) + \frac{1}{2}\partial_\alpha^2 q(\alpha)\Delta\alpha - (A(y + \Delta y))^\top + s + \Delta s & = & z + \Delta z \\ (g_i + \Delta g_i)(z_i + \Delta z_i) & = & \mu \\ (s_i + \Delta s_i)(t_i + \Delta t_i) & = & \mu \end{array} \tag{4.11}$$

for the variables in Δ. This method is described in great detail in [Vanderbei, 1994] for quadratic programming. One gets

$$\begin{array}{rcll} A\Delta\alpha & = & b - A\alpha & =: \ \rho \\ \Delta\alpha - \Delta g & = & l - \alpha + g & =: \ \nu \\ \Delta\alpha + \Delta t & = & u - \alpha - t & =: \ \tau \\ (A\Delta y)^\top + \Delta z - \Delta s - \frac{1}{2}\partial_\alpha^2 q(\alpha)\Delta\alpha & = & c - (Ay)^\top + \frac{1}{2}\partial_\alpha q(\alpha) + s - z & =: \ \sigma \\ g^{-1}z\Delta g + \Delta z & = & \mu g^{-1} - z - g^{-1}\Delta g \Delta z & =: \ \gamma_z \\ t^{-1}s\Delta t + \Delta s & = & \mu t^{-1} - s - t^{-1}\Delta t \Delta s & =: \ \gamma_s \end{array} \tag{4.12}$$

where g^{-1} denotes the vector $(1/g_1, \ldots, 1/g_n)$, and t^{-1} analogously. Moreover denote $g^{-1}z$ and $t^{-1}s$ the vector generated by the componentwise product of the two vectors. Solving for $\Delta g, \Delta t, \Delta z, \Delta s$ yields

$$\begin{array}{lrcl} & \Delta g & = & z^{-1}g(\gamma_z - \Delta z) \\ & \Delta t & = & s^{-1}t(\gamma_s - \Delta s) \\ \text{define} & \hat{\nu} & := & \nu - z^{-1}g\gamma_z \\ & \hat{\tau} & := & \tau - s^{-1}t\gamma_s \\ \text{hence} & \Delta z & = & g^{-1}z(\hat{\nu} - \Delta\alpha) \\ & \Delta s & = & t^{-1}s(\Delta\alpha - \hat{\tau}). \end{array} \tag{4.13}$$

Now one can formulate the *reduced KKT–system* (cf. [Vanderbei, 1994] for the quadratic case):

$$\begin{bmatrix} -\left(\frac{1}{2}\partial_\alpha^2 q(\alpha) + g^{-1}z + t^{-1}s\right) & A^\top \\ A & 0 \end{bmatrix} \begin{bmatrix} \Delta\alpha \\ \Delta y \end{bmatrix} = \begin{bmatrix} \sigma - g^{-1}z\hat{\nu} - t^{-1}s\hat{\tau} \\ \rho \end{bmatrix} \tag{4.14}$$

This equation is best solved by a standard Cholesky decomposition.[4] No special considerations about sparse matrices are necessary, as in general the matrix given by the kernel function k will be dense.

4.2.2 Iteration Strategies

For the *predictor–corrector* method one proceeds as follows. In the predictor step solve the system of (4.13) and (4.14) with $\mu = 0$ and all Δ–terms on the rhs set to 0, i.e. $\gamma_z = z, \gamma_s = s$. The values in Δ are substituted back into the definitions for γ_z and γ_s and (4.13) and (4.14) are solved again in the corrector step. As the quadratic part in (4.14) is not affected by the predictor–corrector steps, one only has to invert the quadratic matrix once. This is done best by solving (thus “manually” pivoting)

4. See [Press et al., 1992] for details. If the matrix should happen to be ill conditioned, as may occur in rare cases during the iterations, it is recommendable to use the pseudoinverse or the Bunch–Kaufman decomposition [Bunch and Kaufman, 1977] as a fall–back option.

for the $\frac{1}{2}\partial_\alpha^2 q(\alpha) + g^{-1}z + t^{-1}s$ part, as it is positive definite.

Next the values in Δ obtained by such an iteration step are used to update the corresponding values in $\alpha, s, t, z, \ldots$. To ensure that the variables meet the positivity constraints, the steplength ξ is chosen such that the variables move at most $1-\epsilon$ of their initial distance to the boundaries of the positive orthant. Usually one sets $\epsilon = 0.05$ [Vanderbei, 1994].

Another heuristic is used for computing μ, the parameter determining how much the KKT–conditions should be enforced. Obviously the aim is only to reduce μ as fast as possible, however if one happens to choose μ too small, the condition of the equations will worsen drastically. A setting that has proven to work robustly [Vanderbei, 1994] is

$$\mu = \frac{\langle g, z\rangle + \langle s, t\rangle}{2n}\left(\frac{\xi - 1}{\xi + 10}\right)^2. \tag{4.15}$$

The rationale behind (4.15) is to use the average of the satisfaction of the KKT–conditions (4.10), i.e. the size of the feasibility gap, as point of reference and then decrease μ rapidly if we are far enough away from the boundaries of the positive orthant, to which all variables (except y) are constrained to.

Finally one has to come up with good initial values. Analogously to [Vanderbei, 1994] choose a regularized version of (4.14) with auxiliary variables reset to 0, in order to determine the starting point. One solves

$$\begin{bmatrix} -\left(\frac{1}{2}\partial_\alpha^2 q(\alpha) + \mathbf{1}\right) & A^\top \\ A & \mathbf{1} \end{bmatrix} \begin{bmatrix} \alpha \\ y \end{bmatrix} = \begin{bmatrix} c \\ b \end{bmatrix} \tag{4.16}$$

Moreover one has to ensure positivity of the variables, thus

$$\begin{aligned} x &= \max\left(x, u/100\right) \\ g &= \min\left(\alpha - l, u\right) \\ t &= \min\left(u - \alpha, u\right) \\ z &= \min\left(\max\left(\tfrac{1}{2}\partial_\alpha q\left(\alpha\right) + c - \left(Ay\right)^\top, 0\right) + u/100, u\right) \\ s &= \min\left(\max\left(-\tfrac{1}{2}\partial_\alpha q\left(\alpha\right) - c + \left(Ay\right)^\top, 0\right) + u/100, u\right) \end{aligned} \tag{4.17}$$

4.2.3 Special Considerations for SV Regression

The algorithm described so far can be applied to both SV pattern recognition and regression estimation. For the standard setting in classification

$$q(\alpha) = \sum_{i,j=0}^{m} \alpha_i \alpha_j y_i y_j k(x_i, x_j), \tag{4.18}$$

and consequently

$$\begin{aligned} \partial_{\alpha_i} q(\alpha) &= y_i \textstyle\sum_{j=1}^m \alpha_j y_j k(x_i, x_j) \\ \partial^2_{\alpha_i \alpha_j} q(\alpha) &= y_i y_j k(x_i, x_j), \end{aligned} \tag{4.19}$$

i.e. the Hessian is dense, and as mentioned before the only thing one can do is compute its Cholesky factorization to solve (4.14). In the case of SV regression, however, one gets (with $\alpha = (\alpha_1, \ldots, \alpha_m, \alpha_1^*, \ldots, \alpha_m^*)$)

$$q(\alpha) = \sum_{i,j=1}^{m} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(x_i, x_j) + 2C \sum_{i=1}^{m} T(\alpha_i) + T(\alpha_i^*) \tag{4.20}$$

and therefore

$$\begin{aligned} \partial_{\alpha_i} q(\alpha) &= y_i \textstyle\sum_{j=1}^m \alpha_j y_j k(x_i, x_j) + \frac{d}{d\alpha_i} T(\alpha_i) \\ \partial^2_{\alpha_i \alpha_j} q(\alpha) &= k(x_i, x_j) + 2C\delta_{ij} \frac{d^2}{d\alpha_i^2} T(\alpha_i) \\ \partial^2_{\alpha_i \alpha_j^*} q(\alpha) &= -k(x_i, x_j) \end{aligned} \tag{4.21}$$

$\partial^2_{\alpha_i^* \alpha_j^*} q(\alpha)$ and $\partial^2_{\alpha_i^* \alpha_j} q(\alpha)$ can be computed analogously. Hence one has to deal with a Hessian $M = \begin{bmatrix} K + D & -K \\ -K & K + D' \end{bmatrix}$ where D, D' are diagonal. Via an orthogonal transform $O := \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{1} & \mathbf{1} \\ -\mathbf{1} & \mathbf{1} \end{bmatrix}$ one obtains $O^\top M O = \begin{bmatrix} 2K + \frac{D+D'}{2} & \frac{D-D'}{2} \\ \frac{D-D'}{2} & \frac{D+D'}{2} \end{bmatrix}$, which can be inverted essentially by inverting an $m \times m$ matrix instead of a $2m \times 2m$ system: "essentially," as in addition to the inversion of $2K + \frac{D+D'}{2}$ one has to solve for the diagonal matrices $D \pm D'$, which is, however, only an operation of computational cost $O(m)$. This is the additional advantage one can gain from implementing the optimization algorithm directly, instead of using an off the shelf optimizer.

Finally note that by solving the primal and dual optimization problem simultaneously one also computes parameters corresponding to the initial SV optimization problem. This observation is useful, as it allows to obtain the constant term b directly, namely by setting $b = y$ (where y denotes the corresponding dual–dual = primal variable). See appendix 4.6.2 for details.

4.2.4 Experiments

The purpose of the little benchmark described below is to show that by using interior point techniques for general convex cost functions, the optimization problems resulting from the latter can be solved just as efficiently as the commonplace ε–insensitive loss functions.

As interior point primal–dual path following methods are computationally very efficient in comparison to classical techniques [Vanderbei, 1994], an implementation of this algorithm for quadratic programming (i.e. standard ε–insensitive loss) is used as reference model. Figure 4.1 shows the number of floating point operations which was measured with MATLAB's `flops` command. The modified SV algorithm is as fast (in some cases it needs even less iterations) as the standard one, even though it uses more general cost functions, and thus results in a non quadratic programming problem. For both algorithms the same dataset was used. The cost functions used

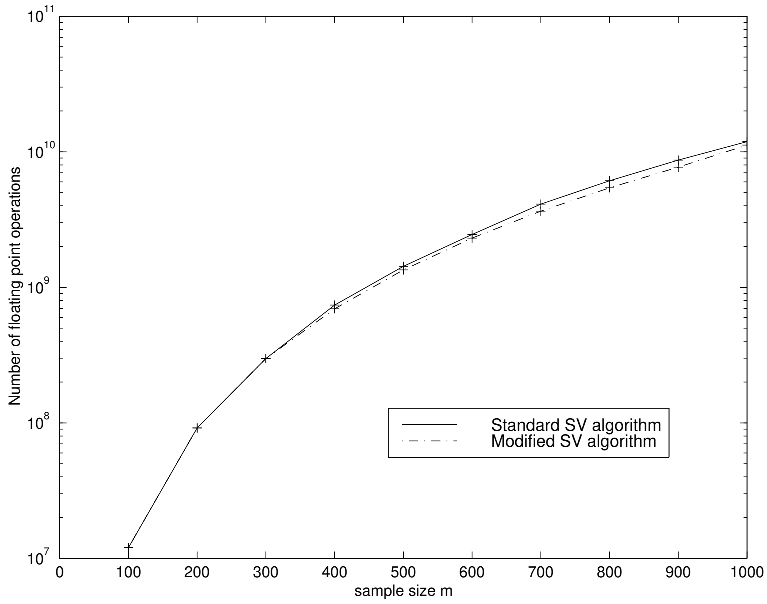

Figure 4.1 Floating point operations for regression problems of different size.

were those of tables 2.1 and 2.2, namely ε–insensitive and piecewise polynomial loss with $\varepsilon = 0.05$, $\lambda = 1$ and $p = 1.5$, the width of the Gaussian RBF–kernels was $\sigma = 1/4$, the dataset was m samples of x drawn uniformly from the interval $[0, 10]$ and $y_i = 2\text{sinc}\,(2(x-5)) + \xi$ with ξ normal noise with standard deviation 0.5.

4.3 Subset Selection Algorithms

The convex programming algorithms described so far can be used directly on moderately sized (up to 3000 samples) datasets without any further modifications to obtain a very precise solution (i.e. the duality gap is of the relative size of 10^{-7}), especially if one happens to have a fast implementation of BLAS [Lawson et al., 1979, Dongarra et al., 1988, 1990] and LAPACK [Anderson et al., 1995] at hand. On large datasets, however, it is difficult, due to memory and cpu limitations, to compute the dot product matrix $k(x_i, x_j)$ and *keep* it in memory. A simple calculation shows that e.g. storing the dot product matrix of the NIST OCR database (60.000 samples) at single precision (4 bytes) would consume 6.9 GBytes, already using the fact that $k(x_i, x_j) = k(x_j, x_i)$. A Cholesky decomposition thereof, which would additionally require roughly the same amount of memory, and 64 TeraFlops (counting multiplies and adds separately), seems unrealistic, at least at current processor speeds. Even worse, interior point algorithms typically need 15 Cholesky iterations to converge, i.e. nearly 10^{15} floating point operations. Hence one has to find more efficient ways for large datasets, even at the expense of obtaining a worse solution than what would be possible by using an interior point approach.

4.3.1 Chunking

A first solution, introduced by Vapnik [1982], relies on the observation that only the SVs are relevant for the final form of the hypothesis. In other words — if one was given only the SVs, one would obtain *exactly* the same final hypothesis as if one had the full training set at disposition. Hence, knowing the SV set beforehand, and moreover being able to fit it into memory, one could directly solve the reduced problem and thereby deal with significantly larger datasets. The catch is, that one does *not* know the SV set before solving the problem. The heuristic is to start with an arbitrary subset, a first *chunk* that fits into memory, train the SV algorithm on it, keep the SVs, and fill the chunk up with data, on which the current estimator would make errors (i.e. data lying outside the ε–tube of the current regression). Then retrain the system and keep on iterating until after training the KKT–conditions are satisfied for all samples.

4.3.2 Working Set Algorithms

The basic chunking algorithm just postponed the basic problem of dealing with datasets whose dot–product matrix cannot be suitably kept in memory. A possible solution to this dilemma was given by Osuna et al. [1997]. They propose to use only a subset of the variables as a working set, and optimize the problem with respect to them while *freezing* the other variables. This method is described in detail in [Osuna et al., 1997, Joachims, 1999] for the case of pattern recognition. Further information can be found in Saunders et al. [1998].

The following is an adaptation thereof to the case of regression with convex cost functions. This is straightforward, as the only non–quadratic part will appear in the term $\sum_i T(\alpha_i) + T(\alpha_i^*)$. Without loss of generality assume $\varepsilon \neq 0$ and $\alpha \in [0, C]$ (for $\varepsilon = 0$ the corresponding terms vanish and the case the $y_i - f(x_i) \in (-\varepsilon, \varepsilon)$ obviously never occurs). First, one has to extract a reduced optimization problem for the working set when all other variables are kept fixed. Denote $S_w \subset \{1, \ldots, m\}$ the working set and $S_f \subset \{1, \ldots, m\}$ the fixed set with $S_w \cup S_f = \{1, \ldots, m\}$ and $S_w \cap S_f = \emptyset$. Writing (2.12) as an optimization problem only in terms of S_w yields

$$\text{maximize} \quad \begin{cases} -\frac{1}{2} \sum\limits_{i,j \in S_w} (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) k(x_i, x_j) \\ + \sum\limits_{i \in S_w} (\alpha_i - \alpha_i^*) \left(y_i - \sum\limits_{j \in S_f} (\alpha_j - \alpha_j^*) k(x_i, x_j) \right) \\ + \sum\limits_{i \in S_w} \left(-\varepsilon (\alpha_i + \alpha_i^*) + C \left(T(\alpha_i) + T(\alpha_i^*) \right) \right) \end{cases} \tag{4.22}$$

$$\text{subject to} \quad \begin{cases} \sum\limits_{i \in S_w} (\alpha_i - \alpha_i^*) & = & - \sum\limits_{i \in S_f} (\alpha_i - \alpha_i^*) \\ \alpha_i & \in & [0, C] \end{cases}$$

Hence one only has to update the linear term by the coupling with the fixed set

```
(1) Initialize αi, αi* = 0
(2) Choose arbitrary working set Sw
(3) Repeat
    (3.1) Compute coupling terms (linear and constant) for Sw
    (3.2) Solve reduced optimization problem
    (3.3) Choose new Sw from variables αi, αi* not satisfying the
          KKT conditions
(4) Until working set Sw = ∅
```

Table 4.1 Basic structure of a working set algorithm.

$-\sum_{i \in S_w} (\alpha_i - \alpha_i^*) \sum_{j \in S_f} (\alpha_j - \alpha_j^*) k(x_i, x_j)$ and the equality constraint[5] by $-\sum_{i \in S_f} (\alpha_i - \alpha_i^*)$. It is easy to see that minimizing (4.22) also decreases (2.12) by exactly the same amount. By choosing variables for which the KKT–conditions are not satisfied, one is guaranteed to strictly decrease the overall objective function, whilst still keeping all variables feasible. Moreover, the objective function is bounded from below by 0. Table 4.1 describes the basic algorithm.

Even though the algorithm presented in table 4.1 cannot be shown to converge in a finite number of steps[6], in many cases this algorithm proves useful in practice. It is one of the few methods (besides [Kaufmann, 1999, Platt, 1999]) that *can* deal with problems whose quadratic part does not completely fit into memory. Still in practice one has to take special precautions to avoid stalling of convergence. The crucial part is step (3.3) of the algorithm in table 4.1, namely which working set S_w to select.

4.3.3 A Note on Optimality

For convenience the KKT conditions are repeated in a slightly modified form. Denote φ_i the error made by the current hypothesis at sample x_i, i.e.

$$\varphi_i := y_i - f(x_i) = y_i - \left[\sum_{j=1}^{m} k(x_i, x_j)(\alpha_i - \alpha_i^*) + b \right]. \tag{4.23}$$

Rewriting the feasibility condition of (4.3) in terms of α_i, α_i^* yields

$$\begin{aligned} 2\partial_{\alpha_i} T(\alpha_i) + \varepsilon - \varphi_i + s_i - z_i &= 0 \\ 2\partial_{\alpha_i^*} T(\alpha_i^*) + \varepsilon + \varphi_i + s_i^* - z_i^* &= 0 \end{aligned} \tag{4.24}$$

5. Also note that it is convenient to store the vector $K\alpha$ and update it after each optimization step. This is significantly cheaper than computing the linear term of the restricted optimization problem from the coefficients α_i, α_i^* over and over again.
6. The reasoning in Osuna et al. [1997] is slightly incorrect, as they argue that a decreasing sequence always reaches a corresponding lower bound in a finite number of steps, which is clearly not correct.

for all $i \in \{1, \ldots, m\}$ with $z_i, z_i^*, s_i, s_i^* \geq 0$. Now one has to find a set of dual feasible variables z, s. This is done by letting

$$\begin{aligned} z_i &= \max\left(2\partial_{\alpha_i} T(\alpha_i) + \varepsilon - \varphi_i, 0\right) \\ s_i &= -\min\left(2\partial_{\alpha_i} T(\alpha_i) + \varepsilon - \varphi_i, 0\right) \\ z_i^* &= \max\left(2\partial_{\alpha_i^*} T(\alpha_i^*) + \varepsilon + \varphi_i, 0\right) \\ s_i^* &= -\min\left(2\partial_{\alpha_i^*} T(\alpha_i^*) + \varepsilon + \varphi_i, 0\right) \end{aligned} \tag{4.25}$$

Consequently the KKT conditions (4.4) can be translated into

$$\begin{aligned} \alpha_i z_i = 0 &\quad \text{and} \quad (C - \alpha_i) s_i = 0 \\ \alpha_i^* z_i^* = 0 &\quad \text{and} \quad (C - \alpha_i^*) s_i^* = 0 \end{aligned} \tag{4.26}$$

All variables α_i, α_i^* violating some of the conditions of (4.26) may be selected for further optimization. In most cases, especially in the initial stage of the optimization algorithm, this set of patterns is much larger than any practical size of S_w. Unfortunately [Osuna et al., 1997] contains little information on how to select S_w. The heuristics presented here are an adaptation of [Joachims, 1999] to regression.

4.3.4 Selection Rules

Similarly to a merit function approach [El-Bakry et al., 1996] the idea is to select those variables that violate (4.24) and (4.26) most, thus contribute most to the feasibility gap. Hence one defines a score variable ζ_i by

$$\begin{aligned} \zeta_i &:= g_i z_i + s_i t_i \\ &= \alpha_i z_i + \alpha_i^* z_i^* + (C - \alpha_i) s_i + (C - \alpha_i^*) s_i^* \end{aligned} \tag{4.27}$$

By construction, $\sum_i \zeta_i$ is the size of the feasibility gap (cf. (4.8) for the case of ε–insensitive loss). By decreasing this gap, one approaches the optimal solution (upper bounded by the primal objective and lower bounded by the dual objective function). Hence, the selection rule is to choose those patterns for which ζ_i is largest.[7]

Finally, heuristics like assigning *sticky*–flags (cf. [Burges, 1998b]) to variables at the boundaries, thus effectively solving smaller subproblems, or completely removing the corresponding patterns from the training set while accounting for their couplings [Joachims, 1999] can significantly decrease the size of the problem and thus result in a noticeable speedup.

---

7. Some algorithms replace ζ_i by

$$\zeta_i' := \alpha_i H(z_i) + \alpha_i^* H(z_i^*) + (C - \alpha_i) H(s_i) + (C - \alpha_i^*) H(s_i) \quad \text{or} \tag{4.28}$$

$$\zeta_i'' := H(\alpha_i) z_i + H(\alpha_i^*) z_i^* + H(C - \alpha_i) s_i + H(C - \alpha_i^*) s_i \tag{4.29}$$

where $H(.)$ denotes the Heavyside function which is 1 if its argument is positive, and zero otherwise. One can see that $\zeta_i = 0$, $\zeta_i' = 0$, and $\zeta_i'' = 0$ mutually imply each other. But only ζ_i measures the contribution of the variable i to the size of the feasibility gap.

4.4 Sequential Minimal Optimization

Recently an algorithm — Sequential Minimal Optimization (SMO)— was proposed [Platt, 1999] that puts chunking to the extreme by iteratively selecting subsets only of size 2 and optimizing the target function with respect to them. It has been reported to be several orders of magnitude faster (up to a factor of 1000) and exhibit better scaling properties (typically up to one order better) than classical chunking (sec. 4.3.1). The key point is that for a working set of 2 the optimization subproblem can be solved analytically without explicitly invoking a quadratic optimizer.

While readily derived for pattern recognition by Platt [1999], one simply has to mimick the original reasoning to obtain an extension to regression estimation. This is what will be done in the following — for the sake of convenience, the complete algorithm is described (including pseudocode, cf. appendix 4.6.3). The modifications consist of a pattern dependent regularization, convergence control via the number of significant figures, and a modified system of equations to solve the optimization problem in two variables for regression analytically.

Note that the reasoning only applies to SV regression with the ε insensitive loss function — for most other convex cost functions an explicit solution of the restricted quadratic programming problem is impossible.

The exposition proceeds as follows: first one has to derive the (modified) boundary conditions for the constrained 2 indices (i, j) subproblem in regression (section 4.4.1), next one can proceed to solve the optimization problem analytically (cf. section 4.4.2), and finally one has to check, which part of the selection rules have to be modified to make the approach work for regression (section 4.4.3).

4.4.1 Pattern Dependent Regularization

Consider the constrained optimization problem (4.22) for two indices, say (i, j). Pattern dependent regularization means that C_i may be different for every pattern [Schmidt and Gish, 1996] (possibly even different for α_i and α_i^*) (for convenience, also results for the classification case are given — these are a direct drop in replacement of the corresponding equations in [Platt, 1999]). Define an auxiliary variable $s := y_i y_j$ for classification (here $y_i \in \{1, -1\}$). For regression one has to distinguish four different cases: $(\alpha_i, \alpha_j), (\alpha_i, \alpha_j^*), (\alpha_i^*, \alpha_j), (\alpha_i^*, \alpha_j^*)$. Here, set $s = 1$ for the first and last case, and $s = -1$ otherwise. Thus one obtains from the summation constraint

$$s\alpha_i + \alpha_j = s\alpha_i^{\text{old}} + \alpha_j^{\text{old}} =: \gamma \tag{4.30}$$

for classification, and

$$(\alpha_i - \alpha_i^*) + (\alpha_j - \alpha_j^*) = (\alpha_i^{\text{old}} - \alpha_i^{*\text{old}}) + (\alpha_j^{\text{old}} - \alpha_j^{*\text{old}}) := \gamma \tag{4.31}$$

for regression. Exploiting $\alpha_j^{(*)} \in [0, C_j^{(*)}]$ yields $\alpha_i^{(*)} \in [L, H]$ where L, H are defined as in Tables 4.2 and 4.3.

	$y_i = y_j$	$y_i \neq y_j$
α_i	$L = \max(0, \gamma - C_j)$ $H = \min(C_i, \gamma)$	$L = \max(0, \gamma)$ $H = \min(C_i, \gamma + C_j)$

Table 4.2 Boundary of feasible regions for classification.

	α_j	α_j^*
α_i	$L = \max(0, \gamma - C_j)$ $H = \min(C_i, \gamma)$	$L = \max(0, \gamma)$ $H = \min(C_i, C_j^* + \gamma)$
α_i^*	$L = \max(0, -\gamma)$ $H = \min(C_i^*, -\gamma + C_j)$	$L = \max(0, -\gamma - C_j^*)$ $H = \min(C_i^*, -\gamma)$

Table 4.3 Boundary of feasible regions for regression.

4.4.2 Analytic Solution for Regression

Next one has to solve the optimization problem analytically for two variables (actually one has to consider four variables — $\alpha_i, \alpha_i^*, \alpha_j, \alpha_j^*$ in the regression case). In analogy to [Platt, 1999], using (4.23) define

$$\begin{aligned} v_i &:= y_i - \sum_{a \neq i,j} (\alpha_a - \alpha_a^*) K_{ia} + b \\ &= \varphi_i + (\alpha_i^{\text{old}} - \alpha_i^{*\text{old}}) K_{ii} + (\alpha_j^{\text{old}} - \alpha_j^{*\text{old}}) K_{ij} \end{aligned} \tag{4.32}$$

and therefore

$$v_i - v_j - \gamma(K_{ij} - K_{jj}) = \varphi_i - \varphi_j + (\alpha_i^{\text{old}} - \alpha_i^{*\text{old}})(K_{ii} + K_{jj} - 2K_{ij}) \tag{4.33}$$

Now (4.22) restricted to (i, j) can be rewritten as follows:

$$\begin{aligned} &\text{maximize} \quad \left\{ \begin{array}{l} -\frac{1}{2} \begin{pmatrix} \alpha_i - \alpha_i^* \\ \alpha_j - \alpha_j^* \end{pmatrix}^\top \begin{pmatrix} K_{ii} & K_{ij} \\ K_{ji} & K_{jj} \end{pmatrix} \begin{pmatrix} \alpha_i - \alpha_i^* \\ \alpha_j - \alpha_j^* \end{pmatrix} \\ +v_i(\alpha_i - \alpha_i^*) + v_j(\alpha_j - \alpha_j^*) - \varepsilon(\alpha_i + \alpha_i^* + \alpha_j + \alpha_j^*) \end{array} \right. \\ &\text{subject to} \quad \left\{ \begin{array}{lll} (\alpha_i - \alpha_i^*) + (\alpha_j - \alpha_j^*) & = & \gamma \\ \alpha_i, \alpha_i^*, \alpha_j, \alpha_j^* & \in & [0, C] \end{array} \right. \end{aligned} \tag{4.34}$$

Next one has to eliminate α_j, α_j^* by exploiting the summation constraint. Ignoring terms independent of $\alpha_i^{(*)}$ one obtains[8]

$$\begin{array}{ll} \text{maximize} & \left\{ \begin{array}{l} -\frac{1}{2}(\alpha_i - \alpha_i^*)^2(K_{ii} + K_{jj} - 2K_{ij}) - \varepsilon(\alpha_i + \alpha_i^*)(1 - s) \\ +(\alpha_i - \alpha_i^*)(v_i - v_j - \gamma(K_{ij} - K_{jj})) \end{array} \right. \\ \text{subject to} & \alpha_i^{(*)} \in [L^{(*)}, H^{(*)}]. \end{array} \tag{4.35}$$

The unconstrained maximum of (4.35) with respect to α_i or α_i^* can be found in Table 4.4. Here the shorthand $\eta := K_{ii} + K_{jj} - 2K_{ij}$ is used. It may happen that for

α_i, α_j	$\frac{v_i - v_j - \gamma(K_{ij} - K_{jj})}{\eta}$	$=$	$\alpha_i^{\text{old}} + \frac{\varphi_i - \varphi_j}{\eta}$
α_i, α_j^*	$\frac{v_i - v_j - \gamma(K_{ij} - K_{jj}) - 2\varepsilon}{\eta}$	$=$	$\alpha_i^{\text{old}} + \frac{\varphi_i - \varphi_j - 2\varepsilon}{\eta}$
α_i^*, α_j	$\frac{v_j - v_i + \gamma(K_{ij} - K_{jj}) - 2\varepsilon}{\eta}$	$=$	$\alpha_i^{*\,\text{old}} - \frac{\varphi_i - \varphi_j + 2\varepsilon}{\eta}$
α_i^*, α_j^*	$\frac{v_j - v_i + \gamma(K_{ij} - K_{jj})}{\eta}$	$=$	$\alpha_i^{*\,\text{old}} - \frac{\varphi_i - \varphi_j}{\eta}$

Table 4.4 Unconstrained maximum of the quadratic programming problem.

a fixed pair of indices (i, j) the initially chosen quadrant, say e.g. (α_i, α_j^*) is the one with the optimal solution. In this case one has to check the other quadrants, too. This occurs if one of the two variables hits the 0 boundary — here computation of the corresponding values for the variable with(out) asterisk according to table 4.4, is required. This has to be repeated at most twice, if the overall optimum lies in the opposite quadrant. Fortunately, the additional computational cost is negligible in comparison to the overall update cost for the gradient/error vector, which is $O(m)$ per successful optimization step. All one has to recompute is $\varphi_i^{\text{new}} - \varphi_j^{\text{new}}$, which can be found as

$$\begin{aligned} \varphi_i^{\text{new}} - \varphi_j^{\text{new}} &= \varphi_i^{\text{old}} - \left((\alpha_i^{\text{new}} - \alpha_i^{*\text{new}}) - \left(\alpha_i^{\text{old}} - \alpha_i^{*\text{old}}\right)\right)(K_{ii} - K_{ij}) \\ &\quad -\varphi_j^{\text{old}} - \left(\left(\alpha_j^{\text{new}} - \alpha_j^{*\text{new}}\right) - \left(\alpha_j^{\text{old}} - \alpha_j^{*\text{old}}\right)\right)(K_{ij} - K_{jj}) \\ &= \varphi_i^{\text{old}} - \varphi_j^{\text{old}} - \eta\left((\alpha_i^{\text{new}} - \alpha_i^{*\text{new}}) - \left(\alpha_i^{\text{old}} - \alpha_i^{*\text{old}}\right)\right) \end{aligned} \tag{4.36}$$

The last equality was derived using (4.31).

Due to numerical instabilities, it may happen that $\eta < 0$. In that case η should be set to 0. Negative values of η are not allowed, as $k(\cdot, \cdot)$ has to satisfy Mercer's condition. In that case set $\eta = 0$. The optimal value of α_i lies on the boundaries H or L. One can find out by looking at the gradient, or simply by computing the value of the objective function at the endpoints, which one of the endpoints to take.

---

8. Note that (4.35) only holds for $\alpha_i \alpha_i^* = \alpha_j \alpha_j^* = 0$.

4.4.3 Selection Rule for Regression

Finally, one has to pick indices (i, j) such that the objective function is maximized. Again, the reasoning of SMO [Platt, 1999, sec. 12.2.2] for classification will be mimicked. This means that a two loop approach is chosen to maximize the objective function. The outer loop iterates over all patterns violating the KKT conditions, first only over those with Lagrange multipliers neither on the upper nor lower boundary, and once all of them are satisfied, over all patterns violating the KKT conditions, to ensure self consistency on the complete dataset.[9] This solves the problem of choosing the index i.

Now for j: To make a large step towards the minimum, one looks for large steps in α_i. As it is computationally expensive to compute η for all possible pairs (i, j) one chooses the heuristic to maximize the absolute value of the numerator in the expressions of table 4.4 (i.e. $|\varphi_i - \varphi_j|$ and $|\varphi_i - \varphi_j \pm 2\varepsilon|$, depending on the presence/absence of asterisks). The index j corresponding to the maximum absolute value is chosen for this purpose.

If this heuristic happens to fail, in other words if little progress is made by this choice, all other indices j are looked at (this is what is called "second choice hierarcy" in [Platt, 1999]) in the following way.

1. All indices j corresponding to non–bound examples are looked at, searching for an example to make progress on.
2. In the case that the first heuristic was unsuccessful, all other samples are analyzed until an example is found where progress can be made.
3. If both previous steps fail, SMO proceeds to the next index i.

For a more detailed discussion of these heuristics see [Platt, 1999].

Unlike interior point algorithms SMO does not automatically provide a value for b. However this can be chosen like in section 1.1.4 by having a close look at the Lagrange multipliers $\alpha_i^{(*)}$ obtained. If at least one of the variables $\alpha_i^{(*)}$ and $\alpha_j^{(*)}$ is inside the boundaries, one can exploit (1.14). In the rare case that this does not happen, there exists a whole interval (say $[b_i, b_j]$) of admissible thresholds. Hence one simply takes the average of both: $b = \frac{b_i + b_j}{2}$.

9. It is sometimes useful, especially when dealing with noisy data, to iterate over the complete KKT violating dataset already before complete self consistency on the subset has been achieved. Otherwise much computational resources are spent on making subsets self consistent that are not globally self consistent. This is the reason why in the pseudo code a global loop is initiated already when only less than 10% of the non bound variables changed.

4.4.4 Number of Significant Figures and Feasibility Gap

By essentially minimizing a constrained *primal* optimization problem one cannot ensure that the dual objective function increases with every iteration step.[10] Nevertheless one knows that the minimum value of the objective function lies in the inteval $[\text{dual objective}_i, \text{primal objective}_i]$ for all iteration steps i, hence also in the interval $\left[(\max_{j\leq i} \text{dual objective}_j), \text{primal objective}_i\right]$ for all i. One uses the latter to determine the quality of the current solution.

The calculation of the primal objective function from the prediction errors is straightforward. One uses

$$\sum_{i,j}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k_{ij} = -\sum_i(\alpha_i - \alpha_i^*)(\varphi_i + y_i - b), \tag{4.37}$$

i.e. the definition of φ_i to avoid the matrix–vector multiplication with the dot product matrix. The dual objective function can be computed via the KKT conditions (cf. (4.6)). The number of significant figures, finally, is computed as the decimal logarithm of (4.9), i.e.

$$\text{SigFig} = \log_{10}\left(\frac{\sum_i g_i z_i + s_i t_i}{|\text{Primal Objective}| + 1}\right) \tag{4.38}$$

The constant 1 is added to avoid division by zero. To save computational cost, primal and dual objective function are computed only every, say, 100 steps of the algorithm. Appendix 4.6.3 contains the pseudocode for SMO regression.

4.5 Summing Up

Several algorithms can be used to solve the quadratic programming problem arising in SV regression. Most of them can be shown to share some common strategy that can be understood well in the view of duality theory, as shown in secion 4.1. In particular, monitoring of convergence is done most reliably in this formulation, as the variables α_i are less interesting quantities than the value of the objective function itself.

A class of algorithms to exploit these properties explicitly are interior point primal-dual path following algorithms (see sec. 4.2). They are relatively fast and achieve a high precision of the solution in the case of moderately sized problems (up to approximately 3000 samples). Moreover, these algorithms can be modified easily to suit for general convex cost functions without additional computational cost. However, they require computation and inversion of the kernel matrix K_{ij},

10. It is still an open question how a subset selection optimization algorithm could be devised that decreases *both* primal and dual objective function at the same time. The problem is that this usually involves a number of dual variables of the order of the sample size, which makes this attempt unpractical.

and are thus overly expensive for large problems.

Chunking in its different variants is a first modification to make large scale problems solvable by classical optimization methods. It requires to break up the initial problem into subproblems which are then in turn solved separately. This is guaranteed to decrease the objective function, thus approaching the global optimum. Selection rules in view of duality theory are given in section 4.3.1.

Finally, an adaptation of SMO to the case of regression estimation is derived. Most of the considerations for pattern recognition carried over analogously, however with the additional twist of having to deal with four instead of two separate cases, which rendered some of the equations less elegant than in the classification setting. It is a very robust algorithm, easy to implement, and thus might be the method of choice for a first attempt.

4.6 Appendix

4.6.1 Derivation of Equation (4.3)

The Lagrange function corresponding to (4.2) can be found as

$$\begin{aligned} L &= \tfrac{1}{2}q(\alpha) + \langle c, \alpha\rangle - \langle y, A\alpha - b\rangle \\ &\quad -\langle \alpha - g - l, z\rangle - \langle u - t - \alpha, s\rangle - \langle \gamma, g\rangle - \langle \tau, t\rangle \end{aligned} \tag{4.39}$$

Here s, z, y are free Lagrange multipliers and γ, τ are positively constrained. The derivative of L with respect to the primal variables α, g, t has to vanish. Hence one obtains the following dual constraints.

$$\partial_\alpha L = \frac{1}{2}\partial_\alpha q(\alpha) + c - (Ay)^\top - z + s = 0 \tag{4.40}$$

$$\partial_g L = z - \gamma = 0 \tag{4.41}$$

$$\partial_t L = t - \tau = 0 \tag{4.42}$$

The last two constraints can be rewritten as $z, t \geq 0$, i.e. one obtains positively constrained dual variables. Moreover, substituting (4.41) and (4.42) into (4.39) yields

$$L = \frac{1}{2}q(\alpha) + \langle c, \alpha\rangle - \langle y, A\alpha - b\rangle - \langle \alpha - l, z\rangle - \langle u - \alpha, s\rangle \tag{4.43}$$

together with the KKT conditions $g_i z_i = 0, s_i t_i = 0$ for all $i \in \{1, \ldots, m\}$. Solving (4.40) yields

$$\frac{1}{2}\partial_\alpha q(\alpha) + c - (Ay)^\top + s = z, \tag{4.44}$$

which is the equality constraint of (4.3), moreover substitution into (4.43) yields the corresponding dual objective function:

$$D = \frac{1}{2}q(\alpha) - \frac{1}{2}\langle \partial_\alpha q(\alpha), \alpha\rangle + \langle b, y\rangle + \langle l, z\rangle - \langle u, s\rangle \tag{4.45}$$

Thus one obtains the dual objective function of (4.3).

4.6.2 The Dual–Dual Argument

The subsequent reasoning illustrates why the dual variables of the constraints in the SV optimization problem, thus the dual–dual variables, may be used as solutions of the corresponding primal variables in the case of *linear* dependencies. Assume one wants to solve the following linear programming problem

$$\begin{array}{ll} \text{minimize} & \langle c, \alpha\rangle \\ \text{subject to} & K\alpha + B\beta \leq Z \text{ and } \alpha, \beta \text{ free.} \end{array} \tag{4.46}$$

Here the constraint is meant to hold coefficientwise. The corresponding Lagrange function is

$$L = \langle c, \alpha \rangle - \langle y, Z - K\alpha - B\beta \rangle. \tag{4.47}$$

Requiring $\partial_\alpha L = 0$ and $\partial_\beta L = 0$ yields the dual constraints $c + K^\top y = 0$ and $B^\top y = 0$, and consequently, after backsubstitution, the dual optimization problem.

$$\begin{array}{ll} \text{maximize} & -\langle Z, y \rangle \\ \text{subject to} & c + K^\top y = 0,\ B^\top y = 0, \text{ and } y \geq 0. \end{array} \tag{4.48}$$

Dualizing again yields $\tilde{L} = \langle Z, y \rangle - \langle \tilde{\alpha}, c + K^\top y \rangle - \langle \tilde{\beta}, B^\top y \rangle$. Hence the dual–dual problem yields

$$\begin{array}{ll} \text{maximize} & -\langle c, \tilde{\alpha} \rangle \\ \text{subject to} & K\tilde{\alpha} + B\tilde{\beta} \leq Z \text{ and } \tilde{\alpha}, \tilde{\beta} \text{ free,} \end{array} \tag{4.49}$$

which is again the primal optimization problem, thus also the dual–dual variables can be used instead of the primal ones.

4.6.3 Pseudocode for SMO Regression

```
target = desired output vector
point = training point matrix

procedure takeStep(i1,i2)
   if (i1 == i2) return 0
   alpha1, alpha1* = Lagrange multipliers for i1
   y1 = target[i1]
   phi1 = SVM output on point[i1] - y1 (in error cache)

   k11 = kernel(point[i1],point[i1])
   k12 = kernel(point[i1],point[i2])
   k22 = kernel(point[i2],point[i2])
   eta = 2*k12-k11-k22
   gamma = alpha1 - alpha1* + alpha2 - alpha2*

   % we assume eta > 0. otherwise one has to repeat the complete
   % reasoning similarly (compute objective function for L and H
   % and decide which one is largest

   case1 = case2 = case3 = case4 = finished = 0
   alpha1old = alpha1, alpha1old* = alpha1*
   alpha2old = alpha2, alpha2old* = alpha2*
   delta_phi = phi1 - phi2
```

```
while !finished
   % this loop is passed at most three times
   % case variables needed to avoid attempting small changes twice
   if (case1 == 0) &&
      (alpha1 > 0 || (alpha1* == 0 && deltaphi > 0)) &&
      (alpha2 > 0 || (alpha2* == 0 && deltaphi < 0))
      compute L, H (wrt. alpha1, alpha2)
      if L < H
         a2 = alpha2 - deltaphi/eta
         a2 = min(a2, H)
         a2 = max(L, a2)
         a1 = alpha1 - (a2 - alpha2)
         update alpha1, alpha2 if change is larger than some eps
      else
         finished = 1
      endif
      case1 = 1;
   elseif (case2 == 0) &&
      (alpha1 > 0 || (alpha1* == 0 && deltaphi > 2 epsilon)) &&
      (alpha2* > 0 || (alpha2 == 0 && deltaphi > 2 epsilon))
      compute L, H (wrt. alpha1, alpha2*)
      if L < H
         a2 = alpha2* + (deltaphi - 2 epsilon)/eta
         a2 = min(a2, H)
         a2 = max(L, a2)
         a1 = alpha1 + (a2 - alpha2*)
         update alpha1, alpha2* if change is larger than some eps
      else
         finished = 1
      endif
      case2 = 1;
   elseif (case3 == 0) &&
      (alpha1* > 0 || (alpha1 == 0 && deltaphi < 2 epsilon)) &&
      (alpha2 > 0 || (alpha2* == 0 && deltaphi < 2 epsilon))
      compute L, H (wrt. alpha1*, alpha2)
      if L < H
         a2 = alpha2 - (deltaphi - 2 epsilon)/eta
         a2 = min(a2, H)
         a2 = max(L, a2)
         a1 = alpha1* + (a2 - alpha2)
         update alpha1*, alpha2 if change is larger than some eps
      else
         finished = 1
      endif
```

```
        case3 = 1;
      elseif (case4 == 0) &&
        (alpha1* > 0 || (alpha1 == 0 && deltaphi < 0)) &&
        (alpha2* > 0 || (alpha2 == 0 && deltaphi > 0))
        compute L, H (wrt. alpha1*, alpha2*)
        if L < H
          a2 = alpha2* + deltaphi/eta
          a2 = min(a2, H)
          a2 = max(L, a2)
          a1 = alpha1* - (a2 - alpha2*)
          update alpha1*, alpha2* if change is larger than some eps
        else
          finished = 1
        endif
        case4 = 1;
      else
        finished = 1
      endif
      update deltaphi
    endwhile
    Update threshold to reflect change in Lagrange multipliers
    Update error cache using new Lagrange multipliers
    if changes in alpha1(*), alpha2(*) are larger than some eps
      return 1
    else
      return 0
    endif
endprocedure

procedure examineExample(i2)
    y2 = target[i2]
    alpha2, alpha2* = Lagrange multipliers for i2
    C2, C2* = Constraints for i2
    phi2 = SVM output on point[i2] - y2 (in error cache)

    if ((phi2 > epsilon && alpha2* < C2*) ||
      (phi2 < epsilon && alpha2* > 0  ) ||
      (-phi2 > epsilon && alpha2  < C2 ) ||
      (-phi2 > epsilon && alpha2  > 0  ))
      if (number of non-zero & non-C alpha > 1)
        i1 = result of second choice heuristic
        if takeStep(i1,i2) return 1
      endif
      loop over all non-zero and non-C alpha, random start
```

```
         i1 = identity of current alpha
         if takeStep(i1,i2) return 1
      endloop
      loop over all possible i1, with random start
         i1 = loop variable
         if takeStep(i1,i2) return 1
      endloop
   endif
   return 0
endprocedure

main routine:
   initialize alpha and alpha* array to all zero
   initialize threshold to zero
   numChanged = 0
   examineAll = 1
   SigFig = -100
   LoopCounter = 0

   while ((numChanged > 0 | examineAll) | (SigFig < 3))
      LoopCounter++
      numChanged = 0;
      if (examineAll)
         loop I over all training examples
            numChanged += examineExample(I)
      else
         loop I over examples where alpha is not 0 & not C
            numChanged += examineExample(I)
      endif
      if (mod(LoopCounter, 2) == 0)
           MinimumNumChanged = max(1, 0.1*NumSamples)
      else
           MinimumNumChanged = 1
      endif
      if (examineAll == 1)
         examineAll = 0
      elseif (numChanged < MinimumNumChanged)
         examineAll = 1
      endif
   endwhile
endmain
```

5 Unsupervised Learning

The problems in unsupervised learning are by far less precisely defined than in the supervised counterpart. Usually no explicit cost function exists with desired outputs or anything alike. Instead, one has to make certain assumptions on the data, with respect to which several questions may be asked.

- A possible problem is: "Which properties of the data can be extracted with high confidence?" Or in other words, which feature extracting functions can be found among a given class with, say unit variance and zero mean, and moreover whose properties will not change too much on unseen data. This leads to a *feature extracting* approach of unsupervised learning.
- Another question is: "Which properties describe the data best?" This means that one is looking for a *descriptive* model of the data, thus also a (possibly quite crude) model of the underlying probability distribution. Generative models like Principal Curves [Hastie and Stuetzle, 1989], the Generative Topological Map [Bishop et al., 1998], several linear Gaussian models [Roweis and Ghahramani, 1998], or also simple vector quantizers [Bartlett et al., 1998] are examples thereof.

Roadmap

In the feature extracting approach to extend PCA data is mapped into some feature space $\mathfrak{S}$ where PCA is performed. As the latter can also be seen as connected to regularization, i.e. to extracting the simplest function with given variance, the initial algorithm is extended in two ways. First the contrast function (i.e. the variance in kernel PCA) is replaced, choosing functions common in projection pursuit. Secondly, the regularization is replaced by constraints from sparse coding and linear programming.

The second, data descriptive, approach to generalize PCA follows the reasoning of principal surfaces and generative topographic maps. After the introduction of a regularized quantization functional, a setting that is able to describe a number of unsupervised learning algorithms in a common framework, one may observe that the resulting kernel based algorithm to minimize this functional is closely connected to principal curves with a length constraint. Finally an algorithm for finding such manifolds is given.

5.1 Kernel Principal Component Analysis

Principal Component Analysis (PCA) is widely used as a tool to extract reliable features from given data. Possible applications are object recognition (cf. [Kirby and Sirovich, 1990, Swets and Weng, 1996]) or image processing, and compression. For more details on the matter see e.g. [Pearson, 1901, Hotelling, 1933, Karhunen, 1946, Jolliffe, 1986, Diamantaras and Kung, 1996]. However there may arise situations where PCA is simply not enough. Not enough in the sense that one might have a large amount of low dimensional data at hand, i.e. the case where one could extract more features reliably, or the case where *linear* features extracted from the data are simply not the (most) interesting ones.

One possible solution is to nonlinearly preprocess the data, i.e. map the data into some *feature space* and perform principal component analysis there. This is exactly what is done in Kernel PCA [Schölkopf et al., 1998a]. The following two sections briefly review the basic algorithm.

5.1.1 The Basic Algorithm

As in PCA, one tries to find directions of maximum variance of the data, i.e. one tries to diagonalize the covariance matrix of the data — however this time not of x_i but of $\Phi(x_i)$. Thus the goal is to find the eigensystem of the matrix

$$C := \frac{1}{m} \sum_{i=1}^{m} \Phi(x_i) \Phi^\top(x_i). \tag{5.1}$$

As in chapter 3, the map into feature space may lead to very high dimensional spaces, thus the explicit computation of C is a difficult (and computationally expensive) task. For instance, one might want to compute PCA in the space of all monomials of degree p, in other words compute most important p-th order correlations of the data.

Again, as in chapter 3, the solution is to rewrite the problem, to diagonalize (5.1) in terms of dot products between the mapped images, i.e. $\langle \Phi(x_i), \Phi(x_j) \rangle =: k(x_i, x_j)$. To achieve this goal note that the requirement for eigenvectors/eigenvalues (v, λ) in feature space can be written as

$$Cv = \lambda v \tag{5.2}$$

By construction, the image of C lies in span $\{\Phi(x_1), \ldots \Phi(x_m)\}$, thus for $\lambda \neq 0$ v can be written as a linear combination of the mapped images ($v = \sum_{i=1}^m \alpha_i \Phi(x_i)$). This is all one needs to reformulate the problem. Eq. (5.2) is equivalent to the following equation which has to hold for all $i \in \{1, \ldots m\}$:

$$\langle \Phi(x_i), Cv \rangle = \lambda \langle \Phi(x_i), v \rangle \tag{5.3}$$

$$\frac{1}{m} \sum_{j,j'}^{m} k(x_i, x_j) k(x_j, x_{j'}) \alpha_{j'} = \lambda \sum_j k(x_i, x_j) \alpha_j \tag{5.4}$$

This problem can be solved essentially[1] by computing the eigensystem of the matrix $K_{ij} := k(x_i, x_j)$. This gives an eigensystem (λ_i, α^i) where the eigenvectors are normalized to 1 in coefficient space. Thus one has to rescale α^i to obtain the eigensystem (λ_i, v_i).

$$\left\| \sum_{j=1}^{m} \alpha_j^i \Phi(x_j) \right\|^2 = \sum_{j,j'}^{m} \alpha_j^i \alpha_{j'}^i k(x_j, x_{j'}) = \lambda_i \sum_{j=1}^{m} (\alpha_j^i)^2 = \lambda_i \tag{5.5}$$

Thus the eigenvectors v_i can be written as

$$v_i = (\lambda_i)^{-\frac{1}{2}} \sum_{j=1}^{m} \alpha_j^i \Phi(x_j). \tag{5.6}$$

with eigenvalues λ_i/m due to the normalization of C. Moreover the projection of $\Phi(x_j)$ onto v_i can be written as follows

$$\langle \Phi(x_j), v_i \rangle = (\lambda_i)^{-\frac{1}{2}} \sum_{j'=1}^{m} k(x_j, x_{j'}) \alpha_{j'}^i = (\lambda_i)^{\frac{1}{2}} \alpha_j^i \tag{5.7}$$

and therefore

$$|\langle \Phi(x_j), v_i \rangle| \leq (\lambda_i)^{\frac{1}{2}}. \tag{5.8}$$

This equation will become quite useful in chapter 8 as it means that the data in feature space is contained inside a box with sidelengths at most $2\sqrt{\lambda_i}$.

Also note that the above reasoning works just as well for bilinear forms $\langle \cdot, \cdot \rangle$ with non positive signature (i.e. there exist vectors v with $\langle v, v \rangle < 0$) — most of the considerations regarding Mercer kernels can be thrown overboard in the latter case and one may use quite general symmetric kernels $k(x, y)$.

5.1.2 Centering in Feature Space and Higher Order Subspaces

The definition of the covariance matrix C according to (5.1) relies on the silent assumption that the data be centered in feature space. In general, however, this is not true. One has to compute C based on

$$\tilde{\Phi}(x_i) := \Phi(x_i) - \frac{1}{m} \sum_{i=1}^{m} \Phi(x_i). \tag{5.9}$$

After some algebra (see e.g. [Schölkopf et al., 1998a]) this results in computing the eigensystem of

$$\tilde{K} := (\mathbf{1} - 1_m) K (\mathbf{1} - 1_m), \tag{5.10}$$

1. The problem comes from the nullspace of the matrix K_{ij}, however the corresponding eigenvectors are not interesting anyway for the present considerations. For details see [Schölkopf et al., 1998a]

where the matrix 1_m is defined as the matrix with all entries set to $1/m$, thus the projector onto constant features. This setting is identical to the one derived in section 3.7.2 for the case of conditionally positive definite kernels. There it was used to project out polynomial components of order 0 to ensure positive semidefiniteness of the resulting optimization problem.

The above fact leads to the question whether not a similar approach also might be possible for the case of higher order features. In fact, one might, similar to example 3.6, project out all linear features from K and use matrices as defined in (3.90). This is useful when one wants to ensure that the extracted features are orthogonal to those found by conventional PCA.

Carrying the idea even further, one could construct feature extractors that extract complete hierarchies of features, say the most prominent features of first, second, and third order, plus possibly additional features connected with rbf-kernels.

5.1.3 Optimality of Polynomial Kernel PCA

As already mentioned in the introduction, it is sometimes important to compute the p-th order correlations of some data, or more precisely the most important p-th order cumulants. One can easily see that this goal cannot be achieved by simply computing all correlations and performing PCA on such preprocessed data, simply due to the requirements in terms of memory and computational resources. On the other hand, homogeneous polynomial kernels like $k(x, y) = \langle x, y\rangle^p$ allow to compute these features by using kernel PCA. The theorem stated below shows that Kernel PCA is not just a method which is "second best," in lack of alternatives, but that in fact, it is the only possible method to compute p-th order moments under some invariance conditions.

Theorem 5.1 Invariance of Polynomial Kernels
Up to a scaling factor, kernel PCA with $k(x, y) = \langle x, y\rangle^p$ is the only PCA in a space of all monomials of degree d which is invariant under orthogonal transformations in input space.

This means that even if one *could* compute all monomials of degree p for the data at hand and perform PCA on the monomials, with the additional requirement of not implying any preferred directions, one would obtain multiples of the results generated by kernel PCA. The proof is given in the appendix.

5.2 Kernel Feature Analysis

A closer look at the properties of PCA (e.g. [Jolliffe, 1986]) shows that finding the first eigenvector with respect to some (centered) data $X = \{x_1, \ldots x_m\}$ can also be formulated as finding the direction which exhibits most variance wrt. X. More formally, the first eigenvector v_1 can be obtained as

$$v_1 = \operatorname*{argmax}_{\|v\|^2=1} \frac{1}{m} \sum_{i=1}^{m} |\langle v, x_i \rangle|^2, \tag{5.11}$$

and the next eigenvectors $v_2, \ldots v_d$ as those which form an orthonormal basis where each eigenvector v_i satisfies a similar propertiy to (5.11) with respect to the remaining $(d-i+1)$-dimensional subspace. A similar reasoning can be applied in the case of Kernel PCA. Define the set of admissible weight vectors as

$$V_{\mathrm{SV}} := \left\{ w \,\middle|\, w = \sum_{i=1}^{m} \alpha_i \Phi(x_i) \text{ with } \|w\|^2 = \sum_{i,j}^{m} \alpha_i \alpha_j k(x_i, x_j) = 1 \right\}. \tag{5.12}$$

Now (5.11) may be transformed into

$$v_1 = \operatorname*{argmax}_{v \in V_{\mathrm{SV}}} \frac{1}{m} \sum_{i=1}^{m} |\langle v, \Phi(x_i) \rangle|^2. \tag{5.13}$$

This immediately leads to the question whether not other sets V might lead to useful feature extractors. In particular one could choose

$$V_{\mathrm{LP}} := \left\{ w \,\middle|\, w = \sum_{i=1}^{m} \alpha_i \Phi(x_i) \text{ with } \sum_{i}^{m} |\alpha_i| = 1 \right\}. \tag{5.14}$$

This is equivalent to a regularizer of the type of (1.24) and leads to the following definition of the first "principal vector" in the ℓ_1 context, hence a new way of kernel *feature* analysis.

$$v_1 = \operatorname*{argmax}_{v \in V_{\mathrm{LP}}} \frac{1}{m} \sum_{i=1}^{m} |\langle v, \Phi(x_i) \rangle|^2 \tag{5.15}$$

Again, subsequent "principal vectors" can be defined by enforcing optimality with respect to the remaining subspaces. The solution of (5.15) has the nice property of being sparse in terms of the coefficients α_i, due to the ℓ_1 constraint (the coefficients may be chosen from the "hyperdiamond-shaped" ℓ_1 ball).[2]

The second modification regarding (5.13) is to choose another *contrast* function instead of the variance, that should be optimized. Hence one obtains solutions of the following type of problems

$$v_1 = \operatorname*{argmax}_{v \in V} \frac{1}{m} \sum_{i=1}^{m} q(\langle v, \Phi(x_i) \rangle) \tag{5.16}$$

or more generally

$$v_1 = \operatorname*{argmax}_{v \in V} Q\left(\{\langle v, \Phi(x_1) \rangle, \ldots, \langle v, \Phi(x_m) \rangle\}\right) \tag{5.17}$$

2. Note that the requirement of $\|v\|^2 = 1$ or the corresponding ℓ_1 constraint are necessary — the value of the target function could increase without bound otherwise.

where $q(\cdot)$ and $Q(\cdot)$ are functions which are maximized for a property of the resulting function $\langle v_1, \Phi(x)\rangle$ that might be of interest. This leads to methods which are quite similar to projection pursuit, however with the novelty that they act in feature space rather than in input space.[3]

Common contrast functions define directions as interesting, if they extract features which are least Gaussian, have several modalities, maximize the Fisher Information, the negative Shannon entropy, or other quantities of interest. For a detailed account on these issues see the work of Friedman and Tukey [1974], Friedman and Stuetzle [1981], Huber [1985], Jones and Sibson [1987], Friedman [1987], Härdle [1991].

The price one has to pay for these modifications with respect to standard Kernel PCA is quite high — the optimization problems may get trapped in local minima and the computation of the projections can be computationally expensive.

5.3 Regularized Principal Manifolds

Principal Curves represent the other option given in unsupervised learning.[4] Instead of trying to extract reliable features from the data, one wants to describe properties of the distribution itself.

5.3.1 Quantization Errors

In the following consider (compact) index sets $\mathcal{B}$, maps $\gamma : \mathcal{B} \to \mathcal{X}$, and classes of such maps $\mathcal{F}$ (with $\gamma \in \mathcal{F}$). Here the map $\gamma(\cdot)$ is supposed to describe some basic properties of the probability distribution underlying the sample X. In particular, it minimizes the so–called quantization error

$$R_{\mathrm{q}}[\gamma] := E\left[\min_{b\in\mathcal{B}} \|x - \gamma(b)\|^2\right]. \tag{5.18}$$

Unfortunately, the problem of finding the map γ that minimizes R_{q} is unsolvable, due to the same problems as in section 1.2.1. Hence instead of (5.18) one analyzes the empirical counterpart defined by

$$R_{\mathrm{q,emp}}[\gamma] := \frac{1}{m}\sum_{i=1}^{m} \min_{b\in\mathcal{B}} \|x_i - \gamma(b)\|^2. \tag{5.19}$$

This definition is more useful than it may seem. Many problems of unsupervised learning can be cast in the form of finding a minimizer of (5.18) or (5.19). Consider some practical examples.

3. Both methods coincide when setting $\Phi(x) = x$.
4. For an extended version of this chapter see [Smola et al., 1998d].

Example 5.1 Sample Mean
Define $\mathcal{B} := \{1\}$, $\gamma : 1 \to \gamma_1$ with $\gamma_1 \in \mathcal{X}$, and $\mathcal{F}$ to be the set of all such functions. Then

$$R_{\mathrm{q}}[\gamma] := E\left[\|x - \gamma_1\|^2\right] \tag{5.20}$$

denotes the variance of the data and

$$\operatorname*{argmin}_{\gamma \in \mathcal{F}} R_{\mathrm{q}}[\gamma] = E[x] \tag{5.21}$$

$$\operatorname*{argmin}_{\gamma \in \mathcal{F}} R_{\mathrm{q,emp}}[\gamma] = \frac{1}{m}\sum_{i=1}^{m} x_i. \tag{5.22}$$

Hence, one obtains the (empirical) sample mean as minimizer of the quantization functional. It follows from the law of large numbers that the sample mean and the (estimated) variance converge to the actual values of the distribution. The same holds for $R_{\mathrm{q,emp}}$ and its convergence to R_{q}.

Example 5.2 k–Means Clustering
Define $\mathcal{B} := \{1, \ldots, k\}$, $\gamma : i \to \gamma_i$ with $\gamma_i \in \mathcal{X}$, and $\mathcal{F}$ to be the set of all such functions. Then

$$R_{\mathrm{q}}[\gamma] := E\left[\min_{i \in \{1,\ldots,k\}} \|x - \gamma_i\|^2\right] \tag{5.23}$$

denotes the canonical distortion error of a vector quantizer. In practice one uses the k–means algorithm to find a set of vectors $\{\gamma_1, \ldots, \gamma_k\}$ finding a (local) minimum of the empirical quantization error. Also in this case, one can prove convergence properties of $R_{\mathrm{q,emp}}[\gamma]$ to $R_{\mathrm{q}}[\gamma]$ (cf. [Bartlett et al., 1998]).

Instead of discrete quantization one can also consider a quantizer that maps the data onto a manifold of lower dimensionality than the input space and tries to achieve optimality for this mapping. PCA can also be viewed in this way. In particular the line along the first principal component passing through the empirical sample mean is the line with minimal quantization error [Hastie and Stuetzle, 1989]. This is formalized in the following example:

Example 5.3 Principal Components
Define $\mathcal{B} := \mathbb{R}$, $\gamma : b \to \gamma_0 + b \cdot \gamma_1$ with $\gamma_0, \gamma_1 \in \mathcal{X}$, $\|\gamma_1\| = 1$, and $\mathcal{F}$ to be the set of all such line segments. Then the minimizer of

$$R_{\mathrm{q}}[\gamma] := E\left[\min_{b \in \mathbb{R}} \|x - \gamma_0 - b \cdot \gamma_1\|^2\right] \tag{5.24}$$

yields a line segment that is parallel to the first principal component vector of the distribution underlying X.

Based on the properties of the current example, Hastie and Stuetzle [1989] carried this idea further by also allowing other functions $\gamma(b)$ than linear ones. This leads to the next example.

Example 5.4 Principal Curves
Define $\mathcal{B} := [0,1]$, $\gamma : b \to \gamma(b)$ with $\gamma \in \mathcal{F} = \mathcal{C}^0([0,1])$, i.e. the class of continuous curves, possibly with a further restriction of $\mathcal{F}$. Unfortunately the minimizer of

$$R_{\mathrm{q}}[\gamma] := E\left[\min_{b\in[0,1]} \|x - \gamma(b)\|^2\right] \tag{5.25}$$

is not well defined, unless $\mathcal{F}$ is a compact set. Moreover, even the minimizer of the empirical quantization functional $R_{\mathrm{q,emp}}$ is generally not well defined, either. In fact it is an ill posed problem in the sense of Tikhonov and Arsenin [1977]. Finally, until recently [Kégl et al., 1999] no convergence properties of $R_{\mathrm{q,emp}}[\gamma]$ to $R_{\mathrm{q}}[\gamma]$ could be stated.

Despite the problems mentioned above, an algorithm to minimize $R_{\mathrm{q,emp}}[\gamma]$, was devised by Hastie and Stuetzle [1989]. It proceeds as follows: after initialization to the principal components, the projections of the datapoints onto the curve are estimated, the curve based on that is reestimated, and the latter is smoothed by kernel smoothers or similar techniques.

Kégl et al. [1999] modified the original "principal–curves" algorithm slightly, to be able to prove uniform convergence type results. In particular the changes imply a restriction of $\mathcal{F}$ to polygonal lines with a fixed number of knots and, most important, *fixed* length L.[5] Under these assumptions they are able to prove upper bounds on the expected quantization error with respect to the empirical error.

5.3.2 A Regularized Version

In the following, yet another modification will be proposed, which will lead to an algorithm that is more amenable to implementation. Moreover uniform convergence bounds can be obtained for smooth curves, independently of the number of nodes/gridpoints. For this purpose, a regularized version of the empirical quantization functional is needed.

$$R_{\mathrm{q,reg}}[\gamma] := R_{\mathrm{q,emp}}[\gamma] + \lambda Q[\gamma]. \tag{5.26}$$

In particular, homogeneous quadratic regularizers will be considered.

$$R_{\mathrm{q,reg}}[\gamma] = \sum_{i=1}^{m} \min_{b\in\mathcal{B}} \|x_i - \gamma(b)\|^2 + \frac{\lambda}{2}\|P\gamma(\cdot)\|^2. \tag{5.27}$$

Here P is a regularization operator penalizing unsmooth functons $\gamma(\cdot)$ as defined in section 3.6. In the present case this is a useful assumption, since all curves, which can be transformed into each other by rotations, should be penalized equally.

5. In practice Kegl et al. use a constraint on the angles of a polygonal curve rather than the actual length constraint to achieve sample complexity rates on the training time. For the uniform convergence part, however, the length constraint is used.

Using the results of chapter 3 regarding the connection between regularization operators and kernels it appears suitable to choose a kernel expansion of $\gamma(\cdot)$ matching the regularization operator P. Hence one gets

$$\gamma(b) = \gamma_0 + \sum_{i=1}^{M} \alpha_i k(b_i, b) \text{ with } b_i \in \mathcal{B} \text{ and } \alpha_i \in \mathcal{X}. \tag{5.28}$$

for some previously chosen nodes $b_1, \ldots, b_M$ (one takes as many as one may afford in terms of computational cost). Thus the regularization term can be written as

$$\|P\gamma(\cdot)\|^2 = \sum_{i,j=1}^{M} \langle \alpha_i, \alpha_j \rangle k(b_i, b_j). \tag{5.29}$$

A number of things will be shown in the following:

(1) An EM type algorithm for efficiently minimizing (5.26) is presented.

(2) A special choice of a regularization operator, minimizing $R_{\text{q,reg}}[\gamma(\cdot)]$ is equivalent to minimizing the optimization problem posed in [Kégl et al., 1999].

(3) The connection to the GTM algorithm by Bishop et al. [1998] is made explicit.

An Algorithm for minimizing $R_{\text{q,reg}}[\gamma(\cdot)]$ No re–interpretation of the regularized quantization error as some likelihood (with a suitable prior) of a class of generative models is done. Instead, the techniques of EM algorithms [Dempster et al., 1977] are adapted to solve

$$\min_{\substack{\{\alpha_1,\ldots,\alpha_M\}\subset\mathcal{X} \\ \{\beta_1,\ldots,\beta_m\}\subset\mathcal{B}}} \left[\sum_{i=1}^{m} \left\| x_i - \sum_{j=1}^{M} \alpha_j k(\beta_i, b_j) \right\|^2 + \frac{\lambda}{2} \sum_{i,j=1}^{M} \langle \alpha_i, \alpha_j \rangle k(b_i, b_j) \right] \tag{5.30}$$

likewise in an iterative fashion. For this purpose one iterates over minimizing (5.30) with respect to $\{\beta_1, \ldots, \beta_m\}$, equivalent to the projection step, and $\{\alpha_1, \ldots, \alpha_M\}$, which corresponds to the expectation step. This is repeated until convergence, in practice until the regularized quantization functional does not decrease significantly any further. One obtains:

Projection For each $i \in \{1, \ldots, m\}$ choose β_i such that

$$\beta_i := \operatorname*{argmin}_{\beta \in \mathcal{B}} \|x_i - \gamma(\beta)\|^2. \tag{5.31}$$

Clearly, for fixed α_i, the so chosen β_i minimize the term in (5.30), which in turn is equal to $R_{\text{q,reg}}[\gamma]$ for given α_i and X.

Adaptation Now the parameters β_i are fixed and α_i is adapted such that $R_{\text{q,reg}}[\gamma]$ decreases further. For fixed β_i differentiation of (5.30) with respect to α_i yields

$$\left(\frac{\lambda}{2} K_b + K_\beta^\top K_\beta \right) \alpha = K_\beta^\top X \tag{5.32}$$

where $(K_b)_{ij} := k(b_i, b_j)$ is an $M \times M$ matrix and $(K_\beta)_{ij} := k(\beta_i, b_j)$ is $m \times M$. Moreover, with slight abuse of notation, α, and X denote the *matrix* of all parameters, and samples, respectively. The term in (5.30) keeps on decreasing until the algorithm converges to a (local) minimum. What remains is to find good starting values.

Initialization Unless dealing, as assumed, with centered data, set γ_0 to the sample mean, i.e. $\gamma_0 = \frac{1}{m}\sum_{i=1}^{m} x_i$. Moreover, choose the coefficients α_i such that γ approximately points into the directions of the first D principal components given by the matrix $E := (e_1, \ldots, e_D)$. This is done as follows, analogously to the initialization in the generative topographic map [Bishop et al., 1998, eq. (2.20)].

$$\min_{\{\alpha_1,\ldots,\alpha_M\}\subset \mathcal{X}} \left[\sum_{i=1}^{M} \left\| E(b_i - b_0) - \sum_{j=1}^{M} \alpha_j k(b_i, b_j) \right\|^2 + \frac{\lambda}{2} \sum_{i,j=1}^{M} \langle \alpha_i, \alpha_j \rangle k(b_i, b_j) \right].$$

Thus α is determined as the solution of $\left(\frac{\lambda}{2}\mathbf{1} + K_b\right)\alpha = E(B - B_0)$ where B denoted the matrix of b_i, b_0 the mean of b_i and B_0 the corresponding matrix.

The derivation of this algorithm was quite "ad hoc", however, one can show that there exist similar precursors in the literature. First it is shown that minimizing (5.26) is equivalent to minimizing the quantization error subject to a length constraint on the estimated curve.

Regularizers for Length Constraints By choosing $P := \partial_b$, i.e. the differentiation operator in the one–dimensional case, $\|Pf\|^2$ becomes an integral over the squared "speed" of the curve.[6]

Reparametrizing $\gamma(\cdot)$ to constant speed leaves the empirical quantization error unchanged, whereas the regularization term is minimized. This can be seen as follows — by construction $\int_{[0,1]} \|\partial_t \gamma(b)\| dt$ does not depend on the (re)parametrization. The integral over $\|\partial_t \gamma(b)\|^2$, however, is minimal for a constant function. Hence $\|\partial_t \gamma(b)\|$ has to be constant over interval $[0,1]$. Thus $\|P\gamma(\cdot)\|^2$ equals the squared length L^2 of the curve at the optimal solution.

Due to the reasoning in section 1.2.4 one can see that minimizing the empirical quantization error plus a regularizer is equivalent to minimizing the empirical quantization error for a fixed value of the regularization term (for λ adjusted suitably). Hence the proposed algorithm is equivalent to finding the optimal curve subject to a length constraint, i.e. it is equivalent to the approach theoretically postulated (not the implementation, though) by Kégl et al. [1999].[7] However, one only finds the length a posteriori (which is not a major restriction, cf. the reasoning in section 1.2.4).

The Connection to the GTM The basic aim of the Generative Topographic Map was to provide a principled probabilistic replacement of more ad hoc methods

---

6. Usually $\gamma(\cdot)$ is parametrized to have "unit-speed" and $\mathcal{B}$ is adapted instead of fixing it to $[0,1]$. However this is not computationally convenient in the present case.
7. The reasoning only holds for an infinite number of nodes, as otherwise $\gamma(\cdot)$ *cannot* be completely reparametrized to constant speed, being an expansion in terms of a *finite* number of nodes. However the basic properties still hold.

such as the Self Organizing Map (cf. e.g. [Kohonen, 1990]). In particular it attempts to describe the data in terms of a generative, lower dimensional model plus additive Gaussian noise. The prior over the space of manifolds is Gaussian, too, in each basis function.

A closer look at the GTM (ignoring the Bayesian framework), reveals that it minimizes a rather similar quantity to $R_{\mathrm{q,reg}}[\gamma]$. It differs in its choice of $\mathcal{B}$, which is chosen to be a grid, identical with the points b_i in our setting, and the different regularizer (called Gaussian prior in that case) which is of ℓ_2 type. In other words instead of using $\|P\gamma\|^2 = \sum_{i,j} \alpha_i \alpha_j k(x_i, x_j)$ Bishop et al. [1998] choose $\sum_i \|\alpha_i\|^2$ as a regularizer. Finally in the GTM several β_i may take on "responsibility" for having generated a data-point x_i (this follows naturally from the generative model setting in the latter case).

Note that unlike in the GTM (cf. [Bishop et al., 1998, sec. 2.3]) the number of nodes (for the kernel expansion) is not a critical parameter. This is due to the fact that there is a *coupling* between the single centers of the basis functions $k(b_i, b_j)$ via the regularization operator. If needed, one could also see the proposed algorithm in a Gaussian Process context (see Williams [1998]) — the data X then should be interpreted as created by a homogeneous process mapping from $\mathcal{B}$ to $\mathcal{X}$.

Finally the use of periodical kernels (cf. sec. 3) allows one to model circular structures in $\mathcal{X}$. After solving the algorithmic issue one has to come up with good uniform convergence bounds, which will be done in section 9.3.

In analogy to section 3.8 one could also modify (5.28) to allow for a gradual transition from PCA to regularized principal curves. For this purpose let

$$\gamma(b) = \gamma_0 + \alpha_0 b + \sum_{i=1}^{M} \alpha_i k(b_i, b) \text{ with } b_i \in \mathcal{B},\ \gamma_0, \alpha_i \in \mathcal{X}, \text{ and } \alpha_0 \in \mathcal{X}^{\dim \mathcal{B}}. \tag{5.33}$$

For large λ, the nonparametric term essentially vanishes (due to the corresponding regularization), and one recovers standard PCA; for decreasing λ the curve adapts more and more to the nonlinearity that might be inherent in the data. Thus in hindsight, also the initialization of the parameters α_i, as adopted from GTM can be seen to be reasonable.

It is worth while noticing that the equations derived above do not make any implicit assumption on the dimensionality of $\mathcal{B}$. In fact, instead of regularized principal curves one could also construct other possible manifolds.

Finally the use of periodical kernels as derived in chapter 3 allows one to model circular structures (or more generally toroidal manifolds) in $\mathcal{X}$. For tools to bound the quantization error in terms of the empirical one and the model complexity (via $\|P\gamma(\cdot)\|^2$) see section 9.3.

5.4 Summing Up

This chapter briefly reviewed the problem of unsupervised learning from two different viewpoints. Seeking reliable feature extractors, which can be done by using suitable contrast functions in conjunction with regularization functionals. This approach led to algorithms like Kernel PCA and more generally Kernel Feature Analysis. Moreover one could see that Kernel PCA is optimal for extraction of polynomial features, thus making it possible to work with spaces of polynomials otherwise untractable to computation.

The second (data descriptive) approach led to an algorithm whose roots can be found both in the GTM and Principal Curves. Due to the formulation of the problem as minimizing the expected quantization error, an expression quite similar to the classical risk functional of supervised learning, a number of methods (such as regularization and kernels) could be applied to the case of unsupervised learning. Moreover the paper of Kégl et al. [1999] shows a way to compute uniform convergence bounds in the latter case. The issue of preparing the tools to derive practical bounds, however, is relegated to section 9.3 as it requires some concepts from functional analysis.

5.5 Appendix: Proof of Theorem 5.1

The proof requires basic notions from group theory. Denote $\mathrm{O}(d)$ the orthogonal group on $\mathbb{R}^d$, i.e. the group of $d \times d$ matrices with $O^\top O = 1$; with the additional requirement $\det O = 1$, one obtains the special orthogonal group $\mathrm{SO}(d)$. A representation ρ of $\mathrm{SO}(d)$ is a map that preserves the group structure, i.e. $\rho(O_1 O_2) = \rho(O_1)\rho(O_2)$ for all $O_1, O_2 \in \mathrm{SO}(d)$.

Proof **($\Longrightarrow$)** Due to the definition of $k(x,y) := \langle x, y\rangle^p$, it follows immediately that $k(Ox, Oy) = \langle x, O^\top O y\rangle^p = \langle x, y\rangle^p$ for any $O \in \mathrm{O}(d)$.

($\Longleftarrow$) Denote $P(p,d)$ to be the (feature) space given by the evaluation of all possible monomials of order p on $\mathbb{R}^d$, furnished with a Euclidean dot product. The map into feature space, $\Phi : \mathbb{R}^d \to P(p,d), x \to \Phi(x)$, induces a representation ρ of $\mathrm{SO}(d)$ on $P(p,d)$ via $\Phi(Ox) = \rho(O)\Phi(x)$. This follows from [Vilenkin, 1968, ch. IX.2]. Hence

$$\langle \Phi(x), \Phi(y)\rangle = \langle x, y\rangle^p = \langle Ox, Oy\rangle^p = \langle \rho(O)\Phi(x), \rho(O)\Phi(y)\rangle. \tag{5.34}$$

Moreover ρ is an orthogonal representation, i.e. $\rho(O)^\top \rho(O) = 1$ for all $O \in \mathrm{SO}(d)$. This follows from $\langle \Phi(x), \Phi(y)\rangle = \langle \rho(O)\Phi(x), \rho(O)\Phi(y)\rangle$ and $\operatorname{span}\Phi(\mathbb{R}^d) = P(p,d)$. Next one has to prove that any positive diagonal matrix D acting on $P(d,N)$, satisfying the invariance condition

$$\langle D^{\frac{1}{2}}\Phi(x), D^{\frac{1}{2}}\Phi(y)\rangle = \langle D^{\frac{1}{2}}\rho(O)\Phi(x), D^{\frac{1}{2}}\rho(O)\Phi(y)\rangle \tag{5.35}$$

for all $O \in \mathrm{SO}(d)$, is necessarily a multiple of the unit matrix. If that were not true, then $k_D(x,y) := \langle D^{\frac{1}{2}}\Phi(x), D^{\frac{1}{2}}\Phi(y)\rangle = k_D(Ox, Oy)$ would be a different kernel invariant under $\mathrm{SO}(d)$. Again, as $\operatorname{span}\Phi(\mathbb{R}^d) = P(p,d)$, one may rewrite (5.35) as

$$D = \rho(O)^\top D \rho(O), \text{ i.e. } D\rho(O) = \rho(O) D. \tag{5.36}$$

In componentwise notation (in $P(p,d)$), this reads

$$D_i \rho(O)_{ij} = D_j \rho(O)_{ij}. \tag{5.37}$$

Therefore, one can show that $D_i = D_j$ for all i, j by showing that there exist sufficiently many nonzero $\rho(O)_{ij}$. To this end, consider a rotation $\tilde{O}$ mapping $x_1 := (1, 0, \ldots, 0)$ into $x_2 := \frac{1}{\sqrt{N}}(1, \ldots, 1)$. Clearly, $\Phi(x_1) = (1, 0, \ldots, 0) \in P(p,d)$, whereas $\Phi(x_2) = \Phi(\tilde{O}x_1) = \rho(\tilde{O})\Phi(x_1)$ contains only nonzero entries. Hence also the first row of $\rho(\tilde{O})$ contains only nonzero entries. By (5.37), one concludes that $D_1 = D_i$ for all i, and therefore $D = \lambda 1$.

This completes the argument concerning the invariance of the polynomial kernel. The transfer to the invariance of Kernel PCA, i.e. to the invariance for all test and training sets, is straightforward.[8] ■

8. The above statement does not hold if D may be an arbitrary matrix of full rank. Via Schur's lemma [Hamermesh, 1962] one can show that the number of different subspaces that can be scaled separately equals the number of irreducible representations in ρ.

6 Applications

Besides the experimental demonstrations that the extensions to the basic SV algorithm are feasible, as done in the previous chapters, the present chapter is devoted to two examples where SV methods yield state of the art performance. This is done to give evidence that the performance of SV machines is not limited to Optical Character Recognition (OCR) where world class results has been achieved (cf. Schölkopf [1997]).

Roadmap

The first section contains an application of SVs to classification of elementary particle events. This is a large and very noisy dataset. Hence it is challenging in terms of finding a suitable optimization algorithm and also to see whether SV machines are still competitive in the high noise regime.

Prediction of time series is the other extreme where relatively few samples were available, especially when the data was segmented first by an adaptive clustering algorithm in function space [Müller et al., 1995, Pawelzik et al., 1996]. Also in this case, world class results were obtained using SV regression.

6.1 Classification of Elementary Particle Events

One of the reasons to deal with this classification task (quite different from the rest of this thesis which is mainly concerned with regression) is that classification problems provide a simplified testbed for optimization algorithms. Secondly, the dataset is rather large (75.000 samples for training, 25.000 samples validation, and 50.000 samples test) and very noisy (typically more than 60% of the samples will become SVs) which is a real challenge for optimization algorithms. Moreover it is important to check that SV machines, which, until recently, have been mainly tested on low noise data, also perform well on noisy data. A theoretical reason for that can be found in the robustness result of Schölkopf et al. [1998b], where SV machines and the trimmed mean estimator are connected.

6.1.1 Algorithmic Results

The algorithm chosen to solve the SV optimization problem was SMO (cf. section 4.4), mainly due to the reason that interior point codes are by far computationally to expensive to use in the present case. They would need a quadratic amount of memory and a cubic amount of computational time to converge, and that subset selection codes (cf. section 4.3) are not always stable and stalled in our experiments instead of converging to the optimal solution. SMO proved to be the most robust one.

Figure 6.1 shows the convergence properties of SMO. One may observe that the feasibility gap decreases rapidly. The small plateau like structures in the primal objective function are most likely due to the fact that SMO attempted to optimize on the non–bound variables only (which is a good strategy under the assumption that the final SV set has been found), or that major changes in the configuration of the SV set were necessary before convergence could proceed.

The convergence properties can also be observed in figure 6.2. It shows the number of significant figures, i.e. the logarithm of the maximum deviation from the optimal solution, depending on the number of dot product computations needed. The first thing to notice is the approximately exponential convergence of the algorithm (at least up to a precision of 0.01). Thus the number of significant figures appears to be a good parametrization of the algorithm.

The small "dents" are due to the fact that after convergence, i.e. after that a precision of 3 significant figures was reached, the regularization parameter was decreased by 0.1, the Lagrange multipliers were rescaled accordingly, and the optimization was restarted for the new value of C (cf. sec. 4.1.3). One can see that convergence to the new optimal solution occurs approximately 20 times faster (only $5 \cdot 10^4$ dot product evaluations) than when training from scratch ($9 \cdot 10^5$ dot product evaluations). The kernels used were Gaussian rbfs with σ set to 15, $C_{\text{start}} = 2$, and sample size 75.000.

6.1.2 Classification

As the dataset is very large and data is rather cheap, model selection can be done by cross validation. Moreover, the OPAL collaboration that generated this dataset by a Monte Carlo simulation, has been using a validation set of 25.000 samples, which is sufficiently large.

Being synthetical data, the relative frequencies of occurence of the several classes are different from their counterparts in real data. Hence the classification results have to be corrected by the relative frequencies of occurence. Moreover, punting (rejection of samples) is allowed. This means that only samples with high confidence need to be classified. Overall one is interested in obtaining predictions for one class (charm) with high "purity" (i.e. the fraction of *correctly* classified "charm" events in the class of events classified as "charm"), while having acceptable "efficiency" (i.e. the fraction of correctly classified "charm" events anong all "charm" events).

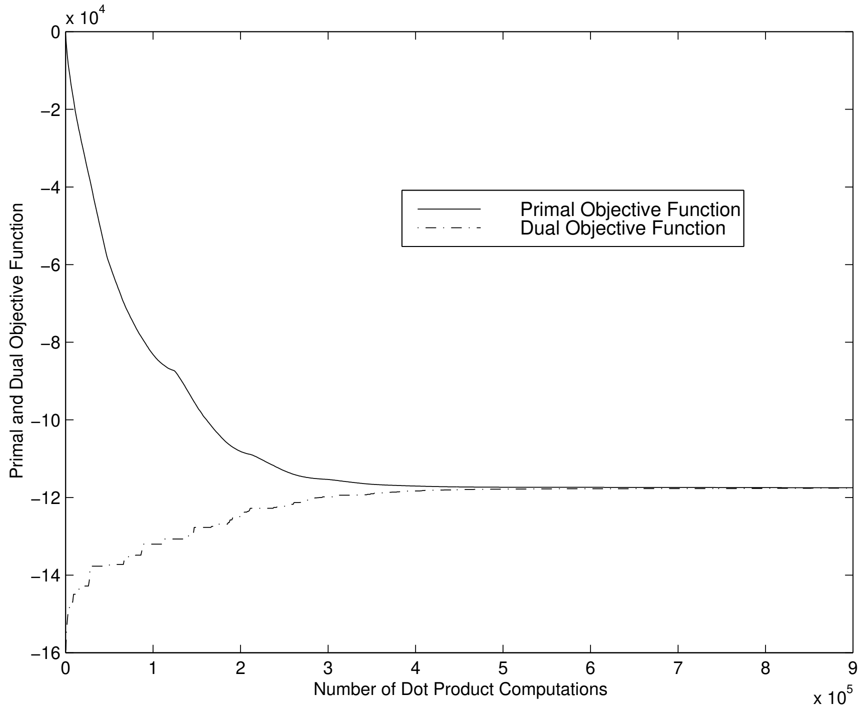

Figure 6.1 Value of the Primal and Dual Objective Function vs. number of dot product computations for the SMO algorithm.

More formally this can be written as follows:

$$\text{efficiency}_{ij} = \text{preselect}_i \frac{\text{number of events "i" classified as "j"}}{\text{total number of events "i"}} \tag{6.1}$$

where preselect_i is a preselection coefficient which depends on the physical setting of the particle detector (i.e. a certain type of events cannot be detected well). One has $\text{preselect}_\text{c} = 0.676199$, $\text{preselect}_\text{b} = 0.726494$, and $\text{preselect}_\text{uds} = 0.646795$. Moreover the purity of such a classification is computed as

$$\text{purity}_i = \frac{p_i \cdot \text{efficiency}_{ii}}{\sum_j p_j \cdot \text{efficiency}_{ji}} \tag{6.2}$$

with $p_\text{b} = 0.614$, $p_\text{c} = 0.171$, and $p_\text{uds} = 0.215$. In practice, one tries to achieve highest purity for an efficiency of approximately 15%. This is done by setting a threshold value such that only patterns with classifier output above this threshold will be accepted as "charm" events. The threshold parameter is determined by a line search (as the efficiency is a monotonically decreasing function of the threshold).

As one can observe, the setting is rather robust to the choice of parameters and achieves roughly 39.4% purity on the test set (cf. figure 6.3). This may seem low, but chance level would be $p_i \text{preselect}_i / (\sum_j p_j \text{preselect}_j)$, i.e. 16.5% for "charm" events.

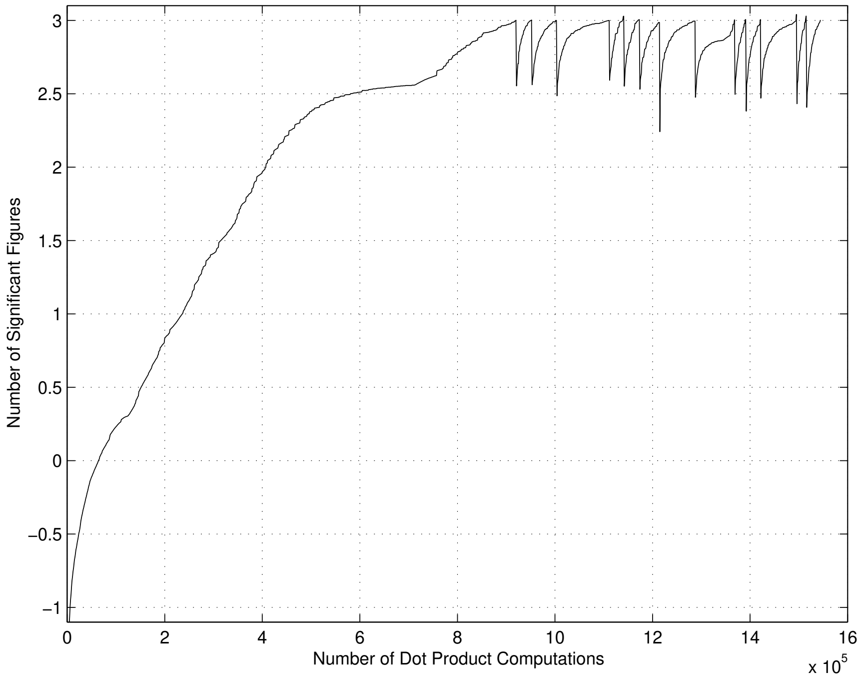

Figure 6.2 Number of significant figures vs. number of dot product computations for the SMO algorithm.

6.1.3 Reference Results

The results were compared to basic techniques, such as k nearest neighbour methods, as those constitute the base line performance. As one is allowed to reject patterns the standard k-nn algorithm is modified in the way that only patterns with a (to be determined) minimum fraction of "charm" events as neighbours will be accepted. In other words, a pattern will be accepted if

$$\frac{n_{\mathrm{c}}}{n_{\mathrm{c}} + n_{\mathrm{b}} + n_{\mathrm{uds}}} - t > 0. \tag{6.3}$$

Here t denotes the threshold and $n_{\mathrm{c}}, n_{\mathrm{b}}, n_{\mathrm{uds}}$ denote the number of "charm", "bottom", or "up/down/strange" events among the k nearest neighbours. As already mentioned before, a problem arises from the fact that the relative frequencies of occurence of the training set, and real data are not identical. In particular, one has to reweight them by the probabilities given in the previous section. Hence a natural modification of (6.3) appears to be the following criterion:

$$\frac{p_{\mathrm{c}} n_{\mathrm{c}}}{p_{\mathrm{c}} n_{\mathrm{c}} + p_{\mathrm{b}} n_{\mathrm{b}} + p_{\mathrm{uds}} n_{\mathrm{uds}}} - t > 0 \tag{6.4}$$

Both variants of the k–nn rule were tested on the dataset. In each case the "efficiency" of the classifier was required to achieve at least 15%. This was done by adjusting the threshold on the validation set. Figure 6.4 shows the performance of

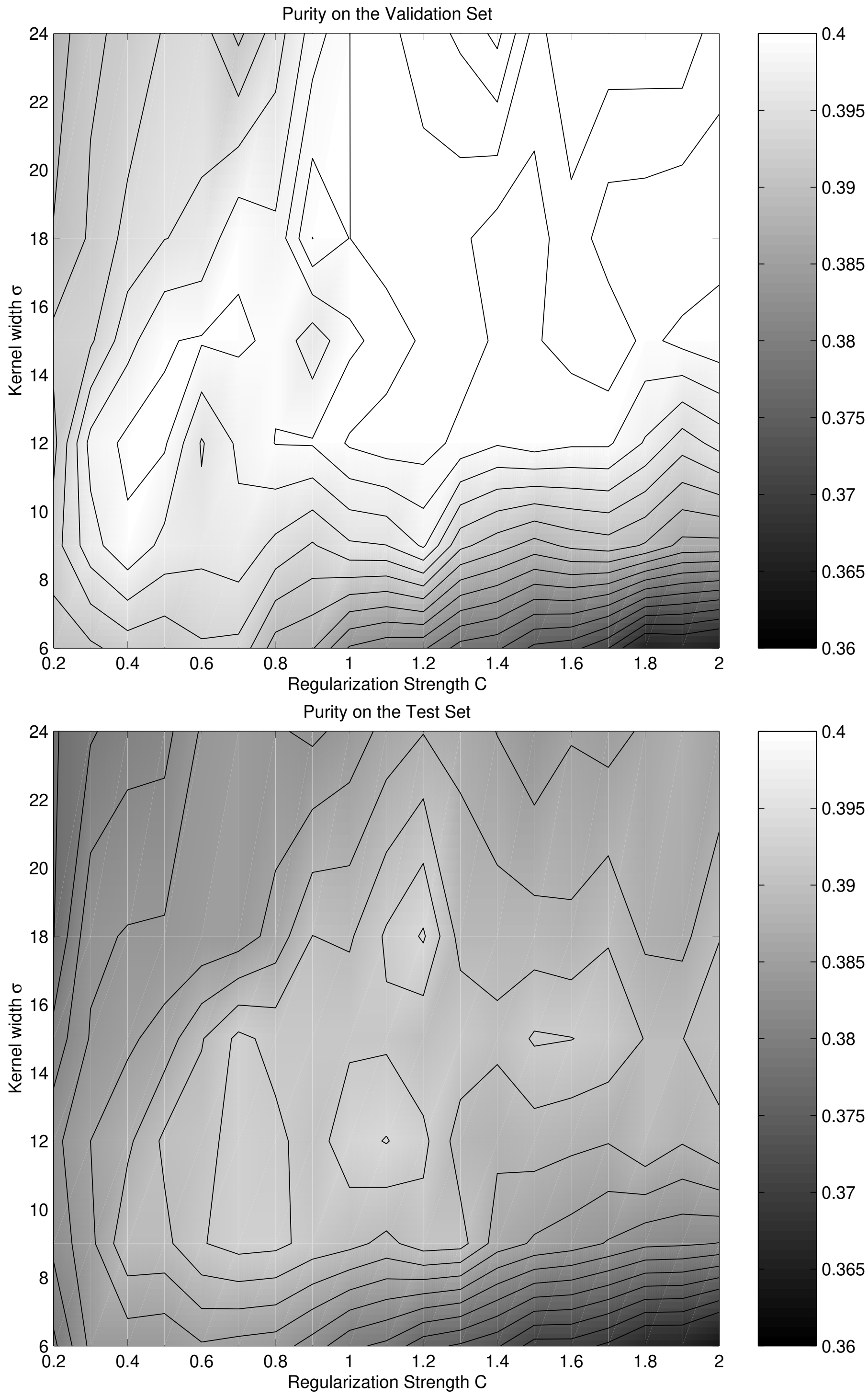

Figure 6.3 Purity of "charm" events for rbf–kernels at 15% efficiency (top: validation set, bottom: test set). Contour lines indicate level changes of 0.2%.

the classifier for both (6.3) and (6.4). Note that the quality of the predictions is best for large k, i.e. for averaging over large parts of the data space. Also note that the different probability weights did not seriously influence the performance. Both may be considered as an indication that the data is extremely noisy.

What is even worse, performance is close to that achieved by SV classification (off by at most 2% in terms of purity). This could cast some serious doubt about the performance of SV machines. However, comparing these results to those of Gaussian rbf networks (with adaptively tuned centers and kernel widths, cf. [Moody and Darken, 1989, Bishop, 1995, Müller et al., 1999, Rätsch, 1998]) on the same dataset show that this is due to the difficulty of the problem. The latter also achieve 40.1% purity on the dataset. Hence not much can be gained from an advanced technique if the dataset is very noisy.

Also a reweighting of the data according to its relative probability weights did not change the overall classification performance.

6.2 Time Series Prediction

The goal in time series prediction[1] is to learn the dynamic of a series of measurements

$$\mathcal{T} := \{t_1, t_2, t_3, \ldots, t_i, \ldots\} \subset \mathcal{Y}, \tag{6.5}$$

i.e. to estimate t_{i+1} (or t_{i+n}) given the sequence $(t_1, \ldots, t_i)$. One way to solve this problem is to use an autoregressive model, i.e. to assume

$$t_{i+1} = f((t_{i-\eta+1}, \ldots, t_i)) \tag{6.6}$$

where $\eta \in \mathbb{N}$ is the embedding. Under quite mild assumptions it can be shown [Takens, 1981] that a chaotic attractor can be represented as a smooth mapping of type (6.6) if the embedding η is chosen to be larger than twice the (fractal) box counting dimension of the attractor.

Given η, the problem reduces to a regression setting, i.e. to find a mapping for the pairs $(x_i, y_i) := ((x_{i-\eta}, \ldots, x_i), x_{i+1})$. In practice, not much further worry is spent on the fact that this data is actually not iid (*independently* identically distributed), i.e. that a permutation of the patterns *would* make a difference. Thus the standard statistical techniques, which rely on the iid assumption, are not applicable. Only recently a method was pointed out how to perform structural risk minimization (which requires uniform convergence bounds) on time series data [Weyer, 1992, Weyer et al., 1996, 1995, Meir, 1998].

However, in the case discussed above, assumptions about certain mixing properties, or about the impulse response properties of the system have to be made,

1. This section follows largely [Müller et al., 1997].

hence the result is not completely distribution free.[2] Despite the aforementioned problems and concerns, the data will be treated as if it had been generated by an iid source and standard regression techniques (SV machines and RBF networks) applied to it.

6.2.1 Techniques

The RBF nets[3] used in the experiments (to compare with) are based on the methods of Moody and Darken [1989] and Müller et al. [1999]. However, not only the output weights are adjusted by backpropagation (on squared loss with regularization), but also the RBF centers and variances. In this way, the networks fine-tune themselves to the data after the clustering step, yet of course overfitting has to be avoided (cf. Bishop [1995], Müller et al. [1999], Rätsch [1998]).

The following experimental setup is fixed for the comparison: (a) RBF nets and (b) SV machines are trained using a simple cross validation technique. Training of the RBF networks is stopped at the minimum of the one step prediction error, measured on a validation set. For SV machines the parameters (λ, ϵ) are also determined at the minimum of the one step prediction error on the validation set. Other methods, e.g. bootstrap could also be used to assess λ and ϵ. Note, for SV machines, a distinction is made between Huber's loss function and the ε–insensitive loss (cf. chapter 2). Gaussian RBF kernels are used with $\sigma^2 = 0.75$.

In the following two experiments will be considered: (i) a toy problem (Mackey Glass time series) to understand and control the experimental set-up and (ii) a benchmark problem from the Santa Fe Competition [Weigend and Gershenfeld, 1994, dataset D].

6.2.2 Mackey Glass Timeseries

The first application is a high-dimensional chaotic system generated by the Mackey-Glass delay differential equation

$$\frac{dx(t)}{dt} = -0.1x(t) + \frac{0.2x(t-t_d)}{1+x(t-t_d)^{10}}, \tag{6.7}$$

with delay $t_d = 17$. Eq. (6.7) was originally introduced as a model of blood cell regulation [Mackey and Glass, 1977] and became quite common as artificial forecasting benchmark. After integrating (6.7), noise was added to the time series. From the time series training (1000 patterns) and validation (194 patterns) sets

2. The uniform convergence proofs of Vapnik [1982] for regression are not completely distribution free, either. Here assumptions on the finiteness of some moments of the distribution are made (which is also the case for the central limit theorem).
3. RBF results courtesy of Gunnar Rätsch.

were obtained, using an embedding dimension $\eta = 6$ and a step size $\tau = 6$, i.e.

$$x_i = (x_i, x_{i-\tau}, \ldots, x_{i-(\eta-1)\tau}). \tag{6.8}$$

The test set (1000 patterns) is noiseless to measure the true prediction error. Experiments were conducted for different signal to noise ratios (SNR), using Gaussian and uniform noise (Table 6.2). RBF networks and SVR achieve similar results for normal noise. It is to be expected that the method using the proper loss function (squared loss) wins for gaussian noise, so one would actually expect the RBF nets to perform best followed by SVR trained with Huber loss, which is (depending on the width parameter) close to the squared loss and finally followed by SVR using an ϵ-insensitive loss. Table 6.2 confirms this intuition largely. For uniform noise the whole scenario should be reversed, since ϵ-insensitive loss is the more appropriate noise model. This is again confirmed in the experiment.

The use of a validation set to assess the proper parameters λ and ϵ, however, is suboptimal and so the low resolution with which the (λ, ϵ) space is scanned is partly responsible for table entries that do not match the above intuition.

noise	normal				uniform					
SNR	22.15%		44.3%		6.2%		12.4%		18.6%	
R_{test}	1S	100S	1S	100S	1S	100S	1S	100S	1S	100S
ϵ iter	0.018	0.079	0.039	0.161	0.018	0.153	0.015	0.111	0.021	0.152
ϵ loss	0.017	0.218	0.040	0.335	0.006	0.028	0.012	0.070	0.017	0.142
Huber	0.017	0.209	0.040	0.339	0.008	0.041	0.014	0.065	0.019	0.226
RBF	0.018	0.109	0.044	0.266	0.009	0.062	0.014	0.083	0.028	0.282

Table 6.1 1S denotes the 1-step prediction error (RMS) on the test set. 100S is the 100-step iterated autonomous prediction. “level” is the ratio between the standard deviation of the respective noise and the underlying time series.

6.2.3 Data Set D from the Santa Fe Competition

Data set D from the Santa Fe competition is artificial data generated from a nine-dimensional periodically driven dissipative dynamical system with an asymmetrical four-well potential and a drift on the parameters [Weigend and Gershenfeld, 1994]. As embedding 20 consecutive points were used. Since the time series is non-stationary, it is first segmented into regimes of approximately stationary dynamics with competing predictors [Müller et al., 1995, Pawelzik et al., 1996]. Only the subset is used for training (327 patterns) which was tagged by the predictor responsible for the data points at the end of the full training set. This allows one to train the RBF networks and the SV machine on quasi stationary data and one avoids to predict the average over all dynamical modes hidden in the full training set (see also Müller et al. [1995], Pawelzik et al. [1996] for further discussion), however at the same time one is left with a rather small training set requiring careful regularization. As in the previous section a validation set (50 patterns) is used to determine the stopping point and (λ, ϵ) respectively.

Table 6.2 shows that the 25 step iterated prediction of the SV regression is 37% better than the one achieved by [Zhang and Hutchinson, 1994], who assumed a stationary model. It is still 29% better than the result of [Pawelzik et al., 1996] that used the same preprocessing as above and simple RBF nets with non-adaptive centers and variances. The results obtained from training on the full (non-stationary) training set (without prior segmentation) are inferior, as expected, however ϵ-insensitive SVR is still better than the previous results on the full set.

experiment	ϵ-ins.	Huber	RBF	Zhang and Hutchinson	Pawelzik et al.
full set	0.0639	0.0857	0.0677	0.0665	-
segmented set	0.0418	0.0425	0.0569	-	0.0596

Table 6.2 Comparison of 25 step iterated predictions (root mean squared errors) on Data set D. "-" denotes: no prediction available. "Full set" means, that the full training set of set D was used, whereas "segmented set" means that a prior segmentation according to [Pawelzik et al., 1996] was done as preprocessing.

6.3 Summing Up

The present chapter showed the applicability of SV machines to two rather different settings: classification on large and noisy datasets and regression in the small sample size case.

The lesson to be learned from the first dataset is that it is not always worth while to train SV machines or other advanced (and computationally expensive) methods on large noisy databases, unless a slight increase in the performance (only 5% in the case discussed above) is essential (which it happens to be in the present case). Base line methods like k nearest neighbours might very well be sufficient.

In the second case, the situation is quite different. For data from the Mackey-Glass equation one could observe that also for SVR it pays to choose the *proper* loss function for the respective noise model. For the data set D benchmark excellent results were obtained for SVR – 37% above the best result achieved during the Santa Fe competition Zhang and Hutchinson [1994]. Clearly, this remarkable difference is mostly due to the segmentation used as preprocessing step to get stationary data, nevertheless still 29% improvement remain compared to a previous result using the same preprocessing step [Pawelzik et al., 1996]. This underlines that one has to consider possible non-stationarities or even mixings in the time series *before* the actual prediction, for which we used SVR or RBF nets.

The experiments show that SV methods work particularly well if the data is *sparse* (i.e. we have little data in a high dimensional space). Moreover small sample size and little noise make SV machines the method of choice. This is largely due to their good inherent regularization properties.

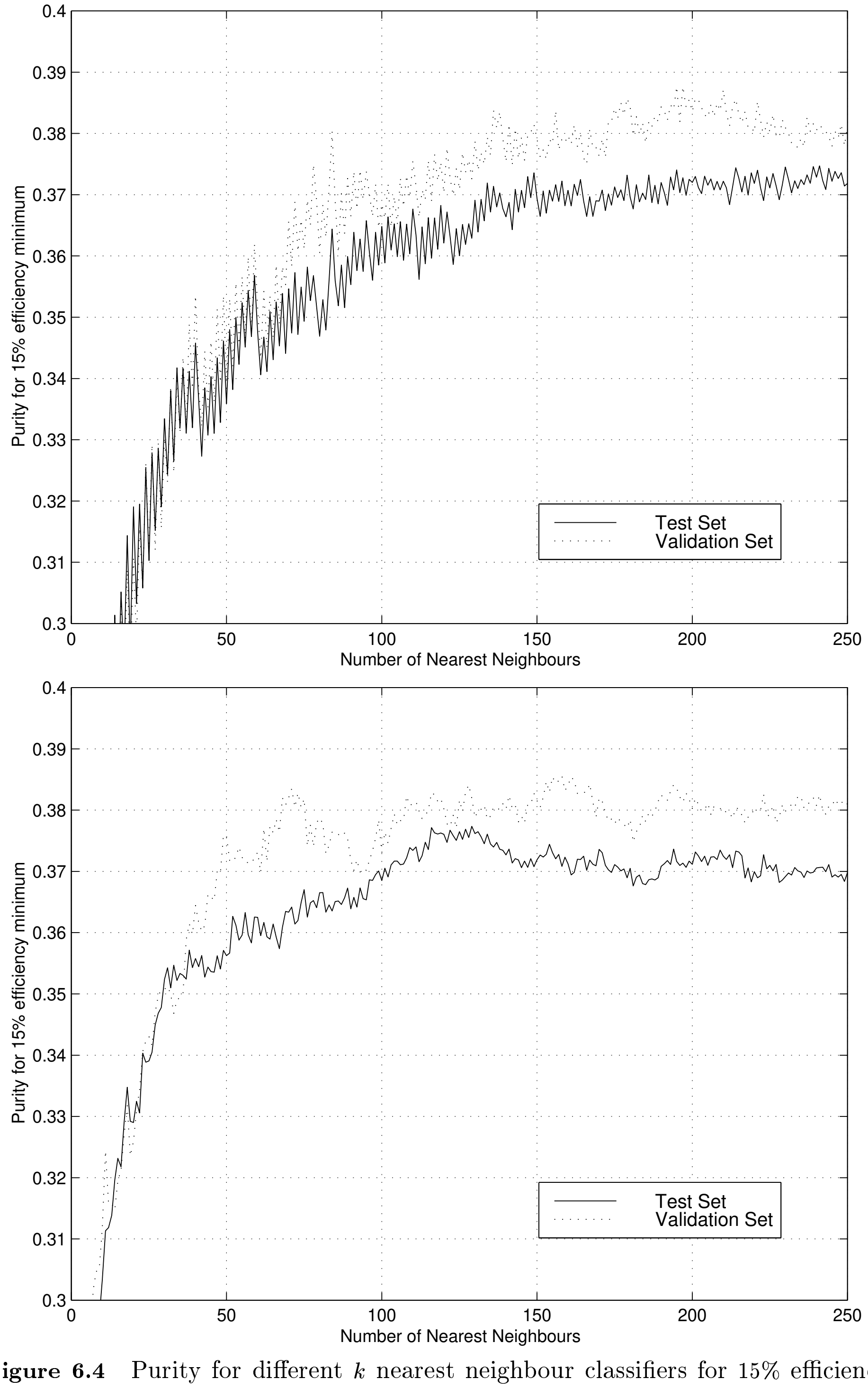

Figure 6.4 Purity for different k nearest neighbour classifiers for 15% efficiency minimum (top: equally weighted patterns, bottom: patterns weighted by the relative frequency of occurence).

II Bounds

The next chapters constitute the learning theory part of this thesis. They can be read rather independently of the previous chapters on algorithms. Some of the results of chapter 3, however, will be needed.

So far, the main concern was to construct computationally feasible algorithms to minimize some regularized risk functional, without too much worry, whether this might be desirable from the point of bounding/minimizing the expected risk (the quantity one ultimately wants to minimize) or not. Hence, what one would like to have, either from theory or some additional experiments, is an expression of the form

$$R[f] \leq R_{\text{emp}}[f] + \mathcal{R}(\mathcal{F}, X, \delta) \quad \text{for all} \quad f \in \mathcal{F}$$

which holds with probability $1 - \delta$. Here $\mathcal{R}(f, X, \delta)$ is some term bounding the deviation between generalization error and empirical error in terms of the chosen model class $\mathcal{F}$, the dataset X, and the degree of confidence δ.

In the following, several types of complexity bounds will be analyzed, with the main emphasis on distribution free bounds of the Vapnik–Chervonenkis type. It will become clear that scale sensitive quantities are needed to deal with estimation problems in the SV context. This is due to the often infinitely dimensional feature spaces where calculations are carried out. After a brief review of existing bounds, a new, functional analytic, approach is taken. This allows *direct* computation of the entropy numbers of a class of functions, instead of taking the detour via the VC dimension. This renders the latter obsolete in cases where the former is available. For instance, for the first time ever, this will allow to state explicitly uniform convergence bounds for SV regression (an implicit version already appeared in

[Anthony and Bartlett, 1994]). Moreover, these techniques will be readily applied to the other types of regularization operators pointed out in part I. Thus, also uniform convergence bounds, independent of the number of basis functions in the expansion will be given for regularization networks, linear programming machines, and regularized principal manifolds.

Roadmap

Chapter 7 gives a brief overview over two methods of controlling capacity (i.e. model selection) in statistical learning theory. First experimental means of estimating the generalization performance are described. This is done, as they are some of the most widespread ones in practice. Next, basic quantities like metrics and covering numbers of sets and classes of functions are introduced. This, together with Hoeffding's inequality, provides the basic toolset necessary for deriving uniform convergence bounds. Several examples thereof are given, for cases of real and binary valued functions, as they are essential for the things to follow in the next chapter. For the sake of completeness, and also to be able to compare the new results with already existing ones, the VC dimension and its scale sensitive counterparts are defined. A brief remark on structural risk minimization concludes this chapter.

Now the scene has been set for the functional analytic approach in chapter 8. It contains the core results of part II. First, some basic properties of entropy numbers (the functional inverse of covering numbers) are described. Next Mercer's theorem is exploited to obtain bounds on the shape of the feature space in kernel methods. In particular, it is shown that entropy numbers can be stated in terms of the eigenspectrum of an integral operator, defined via the kernel. Besides fully analytic methods, also empirical ways of determining entropy numbers are presented. It turns out that the famous bound of Guyon et al. [1993] on the VC dimension of SV classifiers is a weak version of a special case of these considerations. A generalization to more powerful empirical methods will become quite straightforward, thus taking more properties of the data into account. The chapter concludes with an application of the techniques to several kernels in one and higher dimensions. Tight asymptotic rates on entropy numbers are given in this context.

The techniques are sufficiently general to be easily adapted to further schemes of capacity control such as regularization networks. This is done in the last chapter. Also convex combinations, both in terms of kernels and more general function expansions, can be dealt with efficiently. An interesting consequence is that p–convex combinations should be only considered if $p \leq 1$. In all other cases, under quite standard assumptions, the capacity of the system may grow without bound. Moreover, linear programming can be proven to control capacity more efficiently than one would initially think of. The last application of entropy numbers are regularized principal manifolds, a new algorithm for unsupervised learning. Besides uniform convergence bounds, also convergence rates are computed for kernels with exponentially fast rate of decay in the eigenvalues. It turns out to be arbitrarily close to optimal, giving another example of the power of functional analytic methods.

7 Error Bounds

The current chapter recalls the basic tools from statistical learning theory. It contains few new results as its main purpose is to prepare the reader for the subsequent chapters on entropy numbers. It is provided for greater self consistency of the exposition, and to take a slightly modified viewpoint on the quantities involved in bounding the expected risk. Hence readers familiar with the subject might want to skip this chapter entirely.

Roadmap

Empirical methods to estimate the expected risk (and to give maximum deviation bounds) are widespread and popular. Thus, section 7.1 briefly reviews some these techniques and discusses their properties. A close look at crossvalidation (sec. 7.1.1) reveals some of the pitfalls inherent to this method, when used without care. 10–fold crossvalidation and other leave n out estimators are discussed in section 7.1.2. This is done, also to describe the experimental setup of many of the experiments in the first part of the thesis.

Tools from statistical learning theory are introduced in section 7.2. After a brief description of the connection between covering numbers, (level) fat shattering dimensions and the VC dimension some definitions about metrics and norms are stated in section 7.2.1. Covering numbers are introduced in sectin 7.2.2, and Hoeffdings inequality with its implications is stated in section 7.2.3.

A number of different bounds for bounding the expected risk in terms of both covering numbers and the empirical risk is stated in section 7.3. More than the particular constants and exponents, the structure of such statements is important. Two ways to exploit this structure are pointed out subsequently: annealed entropy and growth function are introduced, and methods for obtaining learning curves in terms of dyadic entropy numbers are presented. Finally, section 7.3.4 shows how to deal with a combination of base hypothesis classes and loss functions, as the latter modify the effective resulting hypothesis class.

Section 7.4 introduces the concept of VC dimensions, including the different variants for binary and real valued functions. It is shown that the "plain" VC dimension may not always be the quantity one wants to deal with in SV machines, as the former is unbounded in many cases. Therefore, also the scale sensitive variants of the VC dimension are described and it is shown how to bound the covering numbers (wrt. different metrics) in terms of the (fat) VC dimensions.

An account on structural risk minimization (SRM, see sec. 7.5) concludes the chapter. After describing the basic idea, its implications to SV machines are pointed out. A reason is given, why SVMs in many cases induce data dependent hierarchies, thus making it impossible to apply "plain" SRM in these cases. The rationale is that only limited statements on the shape of the mapped data in feature space $\mathfrak{S}$ can be made, unless one is willing to make assumptions about the domain of interest $\mathcal{X}$.

7.1 Empirical Methods

The first thing to mention are empirical methods. This is due to their simplicity — these methods are straightforward to implement and one can make them work rather well, even without a profound understanding of learning theory. The intuition behind them can be understood rather easily. Therefore, they are methods of choice for many practicioners. Also they serve as a sanity check for more involved bounds. Finally one should note their relatively good performance in most cases (cf. e.g. [Kearns et al., 1997]).

One of the goals of the error bounds is to provide a model selection criterion, which can be used to determine quantities like the kernel width or regularization parameters.

7.1.1 Crossvalidation

The basic idea is quite simple: assume that an estimator $\hat{f}$ depends on two different sets of variables α and β, i.e. $\hat{f} = \hat{f}(\alpha, \beta)$. The qualitative difference between α and β is that α may be many parameters, whose influence on the overall performance may be rather benign, whereas β are structural parameters (e.g. polynomial degree, kernel width, regularization) determining the shape of the model class, from which $\hat{f}$ is drawn. It is intuitively clear that one will want to estimate β quite well, as the latter set of variables seriously affects the overall performance of the estimate.

The conventional wisdom is to split the training set X, Y into an actual training set X_t, Y_t and a validation set X_v, Y_v. Moreover define the training and validation error as

$$R_{\text{train}}[f] := \frac{1}{m_{\text{train}}} \sum_{(x_i, y_i) \in \{X_t, Y_t\}} c(x_i, y_i, f(x_i)) \tag{7.1}$$

$$R_{\text{valid}}[f] := \frac{1}{m_{\text{valid}}} \sum_{(x_i, y_i) \in \{X_v, Y_v\}} c(x_i, y_i, f(x_i)). \tag{7.2}$$

Here c is a cost functions as defined in chapter 2. The crossvalidation rule selects the hypothesis $\hat{f}(\alpha, \beta)$ as follows

$$\beta = \operatorname*{argmin}_{\beta'} \{R_{\text{valid}}[f(\alpha(\beta'), \beta')]\} \tag{7.3}$$

$$\text{where } \alpha(\beta) = \operatorname*{argmin}_{\alpha'} \{R_{\text{train}}[f(\alpha', \beta)]\}. \tag{7.4}$$

At the same time, the term $R_{\text{valid}}[f]$ is assumed to be a good estimate of the expected risk $R[f]$.[1] Recently, several attempts have been made to determine how the split between training and validation set should be done [Amari et al., 1997, Kearns, 1997, Guyon et al., 1998]. The first approach, due to Amari et al. [1997] is based on asymptotic statistics. Unfortunately, the expansions of the error terms obtained for the case of an infinitely large number of samples only hold approximately for the finite sample size case. For instance the experiments of Müller et al. [1996] show that while asymptotic expansions (as one could expect) describe the typical error quite well in the limit case of many samples, they degrade in the small sample size case (which is the more relevant in practice as training data tends to be rare and expensive).

The other approach considers cases where the overall model complexity (in this case the VC dimension) of the joint model $f(\alpha, \beta)$ is bounded (Kearns [1997], Guyon et al. [1998]) and makes use of the bounds. While this assumption is valid by itself, it takes quite sophisticated models from learning theory to show this in practical cases. Moreover in this case, one could use also use methods of VC type to bound the expected risk, without sacrificing a potentially large amount of data.

Another danger arises from the following situation. Imagine a set of variables α that consists only of one scalar, whereas the variables β are of the order of magnitude of the sample size. Not changing the proportion of training and validation set, one might get quite good estimates of α, however the quality of the estimate of β, and in particular of $R_{\text{valid}}[f]$ in comparison to $R[f]$ might be rather bad, as one would effectively be training on the validation set. Finally one might ask why not (α, β) could be found by joint minimization as

$$(\alpha, \beta) = \operatorname*{argmin}_{(\alpha', \beta')} \{R_{\text{emp}}[f(\alpha', \beta')]\} \tag{7.5}$$

which is quite similar to what the EM–algorithm does, thus exploiting the full dataset more effectively to compute (α, β). Hence crossvalidation has to be used with some care. Still, if one needs an easy to implement method to give an estimate of the expected error (especially if β is of low dimensionality and "well behaved"), it may be one of the methods of choice.

7.1.2 Leave n out Estimators

The leave n out estimator, and in particular methods like n–fold crossvalidation try to fix one of the problems brought up by "plain" crossvalidation, namely that

1. This assumption can be removed by splitting the training data into three sets — each one for the parameters α and β, and finally a third one to estimate the expected error $R[f]$. The problem of a reliable estimate, however, remains, and even more, possibly very expensive, training data remains unused for constructing the actual estimator.

one part of the training data is used exclusively to determine α and the other part exclusively for β. This is done as follows (assume the sample size m to be a multiple integer of n):

The dataset is split into m/n parts, each of which is used as validation set to determine β once, whereas the rest is used to determine α. This means that m/n estimators have to be trained instead of a single one, to determine the best value of β_0. Finally an estimate of α is obtained from the complete dataset for this particular value β_0. It is common practice to choose $m/n = 10$, which is also how many experiments in this thesis were carried out. This method is often also referred to as "10–fold crossvalidation".

Unfortunately there are some serious problems with this method, too. Firstly it is extremely computationally expensive and therefore unfeasible where the training times for a single estimator are already in the order of days. Secondly, while being a more reliable method to assess the expected risk, still the same pitfalls, like the (lack of) reliability of the estimate of β, exist as for standard crossvalidation.

There is yet another viewpoint that can be taken, especially for $n = 1$, i.e. for the "Leave 1 Out Estimator". The latter has been proven to provide an unbiased estimate of the expected risk (cf. e.g. Vapnik [1982]). Denote by $\hat{f}_i$ the estimate obtained by estimating $\hat{f}$ based on the dataset $X_i := \{x_1, \ldots, x_{i-1}, x_{i+1}, x_m\}$ and $Y_i := \{y_1, \ldots, y_{i-1}, y_{i+1}, y_m\}$. Then the following equality holds

$$E\left[\frac{1}{m}\sum_{i=1}^{m} c(x_i, y_i, \hat{f}_i(x_i))\right] = R[f]. \tag{7.6}$$

Thus, the leave one out estimate is an unbiased estimator of the expected risk on the dataset. But even worse than before, this method is extremely computationally expensive. However, it provides a useful tool to assess whether a statistical method is robust [Huber, 1981]. In particular, it can be seen as the discretized version of Hampel's influence function [Hampel et al., 1986]. Consequently one can empirically evaluate the robustness of the estimator on the dataset at hand.

After this quick and incomplete overview over empirical methods to bound the expected risk, this chapter will solely focus on arguments of Vapnik–Chervonenkis type to obtain distribution free bounds. Wherever possible, there will be no advocacy for this specific ansatz, and why not other model selection criteria like Bayesian Modelling [MacKay, 1991], Minimum Description Length [Rissanen, 1985], the AIC [Akaike, 1974], or the Network Information Criterion [Murata et al., 1994] are applied to derive error bounds. The only thing to note is that the bounds stated below have been derived using only very few assumptions. The other techniques above are very valuable and valid in situations, where their assumptions are satisfied. This, however, has to be checked *before* comparisons are made.

7.2 Tools from Statistical Learning Theory

Like in many other model selection strategies the aim of Statistical Learning theory is twofold: firstly to give bounds on the expected risk for a particular estimate at hand, and secondly to provide algorithms that generate estimates with low expected risk. Or in other words — the aim is to provide good estimators with a "warranty". For this purpose a set of tools have been developed to quantify the deviation between measured quantities and their expected values.

One of the best known ones is the Vapnik–Chervonenkis (VC) dimension which has the elegant combinatorial interpretation of the maximum number of points that can be shattered by the hypothesis class. In the past 20 years significant efforts were made to bound the VC dimension of all different kinds of estimators, and assessing the quality of the error bounds stated in terms of the VC dimension. The works of Pollard [1984], Cohn and Tesauro [1991], Ji and Psaltis [1992], Maass [1994], Girosi [1995], Karpinski and Macintyre [1995], Hole [1996] and the references therein are examples of this large body of work.

However, the VC dimension enters only at the endpoint of a chain of inequalities to bound the convergence properties of estimators. The consequence thereof is that the bounds derived from the VC dimension (although finiteness of the latter is a necessary and sufficient condition for uniform convergence, cf. Vapnik and Chervonenkis [1991]) may be too lose for practical use.[2] Under this point of view, much of the recent research (and also the contribution of the subsequent chapters) can be seen as an attempt to shorten this sequence of inequalities and bound the more fundamental quantities needed for uniform convergence bounds *directly*.

What follows is a brief sketch of the reasoning, which will be pointed out in more detail in the subsequent sections. The fundamental quantity (at least so far) is the expected covering number, i.e. roughly speaking the expectation of the number $\mathcal{N}$ of functions f from a hypothesis class $\mathcal{F}$ that is needed to represent whole hypothesis class $\mathcal{F}$ with ϵ precision. What actually matters is the logarithm of the expectation of $\mathcal{N}$. To make things more accessible to practical calculations, this quantity is upper bounded by the expectation of the logarithm of $\mathcal{N}$, i.e. the annealed entropy. Finally, to achieve distribution independence, the latter, again, is upper bounded by the sup over all distributions. This is what is called the growth function (cf. e.g. [Vapnik, 1982]). Only the last step is to bound the growth function by a term depending on another quantity, the VC dimension.[3]

A first step towards better bounds was to introduce a *scale sensitive* counterpart of the VC dimension, dubbed the (level) fat shattering VC dimension. It was introduced to statistical learning theory by Kearns and Schapire [1994], the idea

2. Burges [1998b] reports that the bounds derived via a VC dimension argument may be too conservative up to an order of magnitude in SV classification.
3. For a detailed description of the connections, including numerous bibliographical references see [Anthony, 1997, Williamson, 1998, Vapnik, 1998].

of fat shattering itself, however, seems to have been proposed by Kolmogorov in the late 1950's (cf. [Tikhomirov, 1960, pg. 103]) in the context of approximation theory. In a nutshell the idea is that one should analyze functions only at the *scale* at which one is interested in obtaining convergence results. Applications to classification include [Guyon et al., 1993, Shawe-Taylor et al., 1996a,b,c, Schapire et al., 1997]. For the estimation of real valued functions see the work of Bartlett et al. [1996], Lee [1996], Alon et al. [1997]. These results are useful as the growth function at a certain scale can be bounded using the (level) fat shattering dimension at a similar scale. It is worth while noticing that the (level) fat shattering VC dimension can be used to provide *upper* and *lower* bounds on the covering number for certain metrics [Bartlett et al., 1997].

Some special settings also allow computation of the annealed entropy (cf. [Opper, 1999]), and also bounds on the fat shattering VC dimension were obtained by using the techniques of functional analysis by Gurvits [1997] (which one ultimately still has to convert back into covering numbers). The contribution of the present work is that the growth function will be bounded *directly* by functional analytic means, *and* that specific properties of kernel functions will be exploited. Moreover it shows a principled way to obtain these quantities in a rather straightforward fashion.

7.2.1 Metrics and Norms

Most of the following statements hold with respect to pseudo*metric* spaces $(\mathcal{X}, \rho)$. However, unless stated otherwise, the considerations will focus on spaces with a pseudo *norm*, i.e. $(\mathcal{X}, \|\cdot\|)$ (which clearly also induces a metric, simply by setting $\rho(x, y) := \|x - y\|$). This is mainly done for the ease of notation. The ℓ_p^d norm (for $x \in \mathbb{R}^d$) is defined as

$$\begin{aligned} &\text{for } 0 < p < \infty \quad \|x\|_{\ell_p^d} := \|x\|_p = \left(\sum_{j=1}^d |x_j|^p\right)^{1/p} \\ &\text{for } p = \infty \quad \|x\|_{\ell_\infty^d} := \|x\|_\infty = \max_{j=1,\ldots,d} |x_j|. \end{aligned} \tag{7.7}$$

Note that in the following, unlike in some other texts (e.g. Talagrand [1996]) no normalization over d is performed. Now suppose $\mathcal{F}$ is a class of functions on $\mathbb{R}^d$. The ℓ_p^m norm *with respect to* X^m of $f \in \mathcal{F}$ is defined as

$$\|f\|_{\ell_p^{X^m}} := \|(f(x_1), \ldots, f(x_m))\|_p \quad \text{and} \tag{7.8}$$

$$\|f\|_{\ell_\infty^{X^m}} := \max_{i=1,\ldots,m} |f(x_i)|. \tag{7.9}$$

Besides this discrete measure (which induces a pseudometric) on $\mathcal{F}$, one can also think of continuous ones, such as the $L_p(\mu)$ norms: Given a set $\mathcal{X}$, a measure μ on $\mathcal{X}$, some $1 \leq p \leq \infty$ and a function $f\colon \mathcal{X} \to \mathbb{K}$, one can define

$$\|f\|_{L_p(\mu, \mathcal{X}, \mathbb{K})} := \left(\int_{\mathcal{X}} |f(x)|^p d\mu(x)\right)^{1/p} \tag{7.10}$$

if the integral exists, and

$$\|f\|_{L_\infty(\mathcal{X},\mathbb{K})} := \operatorname*{ess\,sup}_{x\in\mathcal{X}} |f(x)|. \tag{7.11}$$

This allows to introduce normed spaces — for $1 \leq p \leq \infty$ let

$$L_p(\mu,\mathcal{X},\mathbb{K}) := \{f : \mathcal{X} \to \mathbb{K} \mid \|f\|_{L_p(\mu,\mathcal{X},\mathbb{K})} < \infty\}. \tag{7.12}$$

In particular, one uses the shorthand $L_p(\mathcal{X})$ for $L_p(\mu,\mathcal{X},\mathbb{R})$ with the uniform measure on $\mathcal{X}$.

7.2.2 Covering Numbers

One of the fundamental quantities needed in the following are covering numbers of sets. They are useful insofar as they allow to replace a set of hypotheses with possibly infinite cardinality by a finite set, as will be briefly pointed out in sections 7.2.3 and 7.3.1.

Definition 7.1 Covering Numbers of a Set
Denote by $(\mathcal{X},\rho)$ a pseudometric space, $B_r(x)$ the closed ball in $\mathcal{X}$ with radius r around x, S a subset of $\mathcal{X}$ and ε some positive constant. Then the *outer* covering number $\mathcal{N}(\epsilon,S,\rho)$ is defined as the minimum cardinality (= number of elements) of a set of points $X \subset \mathcal{X}$ such that

$$S \subseteq \bigcup_{x_i\in X} B_\varepsilon(x_i), \tag{7.13}$$

i.e. such that the maximum difference of any element in S and the closest element in X is less than or equal to ε. Analogously the *inner* covering number[4] $\mathcal{N}_{\text{inner}}(\epsilon,S,\rho)$ is the maximum cardinality of a set X such that

$$\bigcup_{x_i\in X} B_\varepsilon(x_i) \subseteq S \text{ and } B_\varepsilon(x_i)\cap B_\varepsilon(x_j) = \emptyset \text{ for all } i \neq j. \tag{7.14}$$

Thus it is the maximum number of balls of radius ε that 'fit' into S.

One can show (cf. e.g. [Vidyasagar, 1997]) that inner and outer covering numbers are related by

$$\mathcal{N}(2\epsilon,S,\rho) \leq \mathcal{N}_{\text{inner}}(\epsilon,S,\rho) \leq \mathcal{N}(\epsilon,S,\rho). \tag{7.15}$$

In the following, the *outer* covering number will be referred to as *the* covering number. Figure 7.1 gives an example of outer and inner covers of a set.

Next, one needs the covering number of a class of functions $\mathcal{F}$ mapping from $\mathcal{X}$ to $\mathcal{Y}$ with respect to a set X and a metric ρ on $\mathcal{Y}^X$. It is defined as the covering number of the image of X under all function values in $\mathcal{F}$. In the case of an m–sample,

4. This is also referred to as the packing number. Also note that there exists the notion of a maximal system, i.e. a set of elements with at least ε distance from each other.

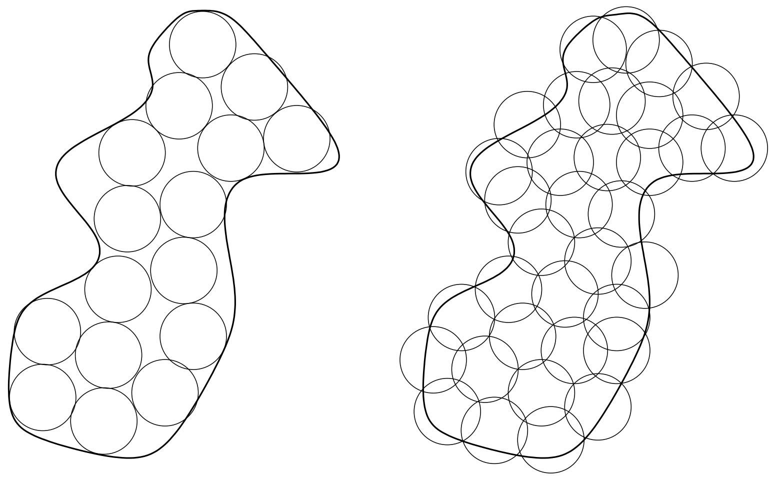

Figure 7.1 Left: Inner cover of a set, Right: Outer cover of a set; displayed are the balls surrounding each of the covering points.

i.e. $X = \{x_1, \ldots, x_m\}$ (sometimes also denoted as X^m) one writes $\mathcal{N}(\varepsilon, \mathcal{F}, \ell_p^{X^m})$. Moreover, in cases where it is clear to which set X^m one is referring to (or where the maximum over all possible sets has been taken) the corresponding term is dropped by writing $\mathcal{N}(\varepsilon, \mathcal{F}, \ell_p^m)$.

Finally, the binary covering number $\pi(\mathcal{F}, X^m)$ is defined as the number of different classifiers that can be generated from $\mathcal{F}$ when restricted to the set X^m. As in this case the function values wrt. X^m are identical or differ by at least 2 (class -1 and 1) one defines[5]

$$\pi(\mathcal{F}, X^m) := \mathcal{N}(1, \mathcal{F}, \ell_\infty^{X^m}). \tag{7.16}$$

7.2.3 Hoeffding's Inequality

Hoeffding [1963] established an inequality, itself a generalization of inequalities derived by Chernoff [1952] and Okamoto [1958]. It is central to the issue of bounding the expected risk in terms of both the model complexity and the empirical error as it allows to bound the expected error of *one* hypothesis by its empirical error and the sample size. It is stated in a form similar to [Devroye et al., 1996].[6]

Theorem 7.2 Hoeffding [1963]
Let $\xi_1, \ldots, \xi_m$ be independent bounded random variables such that ξ_i falls in the interval $[a_i, b_i]$ with probability one. Denote their average by $S_m = \frac{1}{m}\sum_i \xi_i$. Then

5. For real valued functions the definition is analogous — one simply applies the sign function to all elements of $\mathcal{F}$ which yields boolean functions, i.e. one considers the class $\mathcal{F}' = \bigcup_{f \in \mathcal{F}} \operatorname{sgn}(f)$.

6. See [Devroye et al., 1996] for the proof and a discussion of the result.

for any $\varepsilon > 0$ one has

$$\Pr\{S_m - E[S_m] \geq \varepsilon\} \leq e^{-\frac{2m^2\varepsilon^2}{\sum_{i=1}^m (b_i - a_i)^2}} \tag{7.17}$$

and

$$\Pr\{E[S_m] - S_m \geq \varepsilon\} \leq e^{-\frac{2m^2\varepsilon^2}{\sum_{i=1}^m (b_i - a_i)^2}}. \tag{7.18}$$

Therefore, if $\operatorname{card}(\mathcal{F}) = 1$, i.e. if there was only one function to choose from, it would be very easy to bound the expected error, say in pattern recognition, by the empirical error. Consider random variables

$$\xi_i := (1 - \delta_{f(x_i),y_i}) \quad \text{with} \quad 1 \leq i \leq m. \tag{7.19}$$

Clearly $\xi_i \in [0,1]$ with probability one (zero loss for correct classification, 1 for incorrect classification). Hence it follows for the empirical error S_m (defined as above in theorem 7.2) that

$$\Pr\{S_m - E[S_m] \geq \varepsilon\} \leq e^{-2m\varepsilon^2} \text{ and also } \Pr\{S_m - R[f] \geq \varepsilon\} \leq e^{-2m\varepsilon^2} \tag{7.20}$$

This is the starting point for distribution free bounds of arbitrary function classes [Vapnik, 1982, Eq. 6.15]. For this purpose $\mathcal{F}$ has to be replaced by a finite discretization of precision ε, which is exactly the point where $\mathcal{N}$ and π enter.

7.3 Generalization Bounds via Uniform Convergence

What effectively happens is that in order to obtain uniform convergence bounds, $\mathcal{F}$ is replaced by an ε cover of $\mathcal{F}$, and the probability of deviations larger than ε' over $\mathcal{F}$ is bounded by $\mathcal{N}$ times the probability of deviation for a single function. The latter is often referred to as the "union bound".

Several further technical hurdles have to be overcome (e.g. a symmetrization step) to finally achieve the desired goal. For a detailed explanation of the steps in the proof see the review by Anthony [1997], the books of Vapnik [1982, 1995, 1998], Pollard [1984], Devroye et al. [1996], Vidyasagar [1997], or, if courageous, the original works of Vapnik and Chervonenkis [1971, 1974, 1981, 1991]. Next a few practical examples of such bounds are given.

7.3.1 Bounds

Denote by $E_m[f] := \frac{1}{m}\sum_{i=1}^m f(x_i)$ the *empirical mean* of f on the sample $x_1, \ldots, x_m$ drawn with respect to p, $E[f]$ the expectation wrt. P, and $E[\mathcal{N}(\cdot,\cdot,\ell_\infty^{2m})]$ the expectation of $\mathcal{N}$ wrt. a $2m$ sample drawn iid from P.

Lemma 7.3 Alon, Ben-David, Cesa-Bianchi, and Haussler [1997]
Let $\mathcal{F}$ be a class of functions from $\mathcal{X}$ into $[0,1]$ and let P be a distribution over $\mathcal{X}$. then, for all $\epsilon > 0$ and all $m \geq \frac{2}{\epsilon^2}$,

$$\Pr\left\{\sup_{f\in\mathcal{F}} |E_m[f] - E[f]| > \epsilon\right\} \leq 12m \cdot E\left[\mathcal{N}\left(\tfrac{\epsilon}{6}, \mathcal{F}, \ell_\infty^{2m}\right)\right] e^{-\epsilon^2 m/36} \tag{7.21}$$

where Pr denotes the probability wrt. the sample $x_1, \ldots, x_m$ drawn iid from P.

A similar result holds for classification. Analogously to above denote by $E_m[f]$ the average loss of f due to misclassification on an m sample (i.e. the empirical mean), $E[f]$ the expectation wrt. P, and $E[\pi(\cdot, 2m)]$ the expectation wrt. a $2m$ sample drawn iid form P.

Lemma 7.4 Vapnik and Chervonenkis [1971]
Let $\mathcal{F}$ be a class of functions from $\mathcal{X}$ into $\{0;1\}$ and let P be a distribution over $\mathcal{X}$. Then

$$\Pr\left\{\sup_{f\in\mathcal{F}} |E_m[f] - E[f]| > \epsilon\right\} \leq 8E[\pi(\mathcal{F}, 2m)]e^{-\epsilon^2 2m/32}. \tag{7.22}$$

Stronger results are available when the empirical error $R_{\mathrm{emp}}[f]$, for classification, i.e. in this case $E_m[f]$ vanishes for the function f chosen. The original result is due to Vapnik and Chervonenkis [1971], and stated below in a tightened version.

Lemma 7.5 Devroye, Györfi, and Lugosi [1996]
Let $\mathcal{F}$ be a class of functions from $\mathcal{X}$ into $\{0;1\}$, be p a distribution over $\mathcal{X}$ and f chosen such that $E_m(f) = 0$. Then

$$\Pr\{E[f] > \epsilon\} \leq 2E[\pi(\mathcal{F}, 2m)]e^{-\epsilon m/2}. \tag{7.23}$$

Note that the exponent only depends on ϵ instead of ϵ^2. One could continue this list of bounds (cf. e.g. Lee [1996]), however this is not the purpose of this section — the aim is rather to give a brief overview of the basic structure of bounds of this type. The reader interested in details (and proofs) may want to consider [Devroye et al., 1996, Ch. 12]. Summing up, most further uniform convergence result take the form

$$Pr\left\{\sup_{f\in\mathcal{F}} |E_m[f] - E[f]| > \epsilon\right\} \leq c_1(m)E\left[\mathcal{N}(\epsilon, \mathcal{F}, \ell_p^m)\right] e^{-\epsilon^\beta m/c_2}. \tag{7.24}$$

As already seen before, even the exponent in (7.24) depends on the setting: In regression β can be set to 1, however in agnostic learning (i.e. if the function minimizing the expected risk is not in $\mathcal{F}$) [Kearns et al., 1994] in general $\beta = 2$, except if the class is convex, in which case it can be set to 1 [Lee et al., 1998].

7.3.2 Annealed Entropy and Growth Function

The bounds stated in the previous section depend on $E[\mathcal{N}(\cdot,\cdot,2m)]$, which has to be computed wrt. the (usually) unknown probability distribution P. Moreover, taking the logarithm on both sides of (7.24), one may observe that it is not $E[\mathcal{N}(\cdot,\cdot,2m)]$ but $\ln E[\mathcal{N}(\cdot,\cdot,2m)]$ that really matters to state confidence intervals. However, $\ln E[\mathcal{N}]$ is quite hard to compute. Hence one bounds it by the *annealed entropy* (cf. Vapnik [1982])

$$\ln E[\mathcal{N}(\mathcal{F},\varepsilon,m)] \leq E\left[\ln \mathcal{N}(\mathcal{F},\varepsilon,m)\right] =: H(\mathcal{F},\varepsilon,m). \tag{7.25}$$

Expectation and logarithm may be swapped as the latter is concave. However, in many cases, this step is not sufficient yet — also H depends on the probability distribution P. Thus the latter is upper bounded by the growth function $G(\mathcal{F},\varepsilon,m)$ by taking the sup over all m–samples in $\mathcal{X}$, i.e.

$$H(\mathcal{F},\varepsilon,m) \leq \sup_{X^m \subset \mathcal{X}} \ln \mathcal{N}(\mathcal{F},\varepsilon,m). \tag{7.26}$$

Analogous definitions exist for $\pi(\mathcal{F},m)$ in the case of classification.

7.3.3 How to use the Bounds

Often results of the type (7.24) are used by setting the right hand side equal to δ and solving for $m = m(\epsilon,\delta)$ (which is called the sample complexity). Another way to use these results is as a learning curve bound $\bar{\epsilon}(\delta,m)$ where

$$Pr\left\{\sup_{f\in\mathcal{F}} |E_m[f] - E[f]| > \bar{\epsilon}(\delta,m)\right\} \leq \delta. \tag{7.27}$$

Note here that the determination of $\bar{\epsilon}(\delta,m)$ is quite convenient in terms of e_n, the dyadic entropy number associated with the covering number $\mathcal{N}(\epsilon,\mathcal{F},\ell_\infty^m)$ in (7.24). Here $e_{n+1}(\mathcal{F})$ is defined as the functional inverse of $\log_2 \mathcal{N}(\epsilon,\mathcal{F},\ell_\infty^m)$ with respect to ϵ — the additional arguments like $\mathcal{F}$, $\mathcal{X}$, P and ℓ_∞^m have been dropped for compactness of the notation (see the next section for further details). Setting the right hand side of (7.24) equal to δ, yields

$$\begin{aligned}
\delta \quad &= c_1(m)\mathcal{N}(\epsilon,\mathcal{F},\ell_\infty^m)e^{-\epsilon^\beta m/c_2} \\
\Rightarrow \ln\left(\tfrac{\delta}{c_1(m)}\right) + \tfrac{\epsilon^\beta m}{c_2 \ln 2} &= \ln \mathcal{N}(\epsilon,\mathcal{F},\ell_\infty^m) \\
\Rightarrow e_{\log_2\left(\frac{\delta}{c_1(m)}\right)+\frac{\epsilon^\beta m}{c_2 \ln 2}+1} &= \epsilon.
\end{aligned} \tag{7.28}$$

Thus $\bar{\epsilon}(\delta,m) = \{\epsilon\colon (7.28) \text{ holds}\}$. The use of e_n, and later also of ϵ_n, the functional inverse of $\mathcal{N}$ is in fact a convenient thing to do for finding learning curves.

Also observe, as can be seen from (7.28) or the previous uniform convergence theorems, that the bounds are only useful as long as $\ln \mathcal{N}$ is smaller than the sample size. Otherwise the "richness" of the class of functions is too high.

7.3.4 Loss Function Induced Classes

After a close look at the bounds of lemma 7.3, 7.4, and 7.5 one may observe that $E_m[f] \neq R_{\mathrm{emp}}[f]$. Instead, the empirical risk, as defined in (1.19) depends not only on f, but also on the cost function c and the observations x_i, y_i (for the sake of simplicity assume $c(f(x), y, x) = c(f(x) - y)$). Thus one has to deal with a loss function *induced* class of functions.

The following Lemma, which is an improved version of [Bartlett et al., 1996, Lemma 17], is useful in this regard.[7] It deals with loss functions satisfying a Lipschitz condition in the estimate f.

Lemma 7.6 Williamson, Smola, and Schölkopf [1998b]
Denote by $\mathcal{F}$ a set of functions from $\mathcal{X}$ to $[a, b] \subset \mathbb{R}$ and $c : \mathbb{R} \to \mathbb{R}_0^+$ a loss function. Let $Z := \{(x_1, y_1), \ldots, (x_m, y_m)\}$, $c_f|_Z := \{c(f(x_1) - y_1), \ldots, c(f(x_m) - y_m)\}$, $c_{\mathcal{F}}|_Z := \{c_f|_Z : f \in \mathcal{F}\}$ and denote by $\mathcal{N}(\epsilon, c_{\mathcal{F}}|_Z, \ell_p^m)$ the covering number induced by c and $\mathcal{F}$ in terms of the ℓ_p^m norm. Then the following two statements hold:

1. Suppose c satisfies the Lipschitz–condition

$$c(\xi) - c(\xi') \leq C|\xi - \xi'| \text{ for all } \xi, \xi' \in [a - b, b - a]. \tag{7.29}$$

Then for all $\epsilon > 0$

$$\max_{Z \in (\mathcal{X} \times [a,b])^m} \mathcal{N}(\epsilon, c_{\mathcal{F}}|_Z, m) \leq \max_{X \in \mathcal{X}^m} \mathcal{N}\left(\tfrac{\epsilon}{C}, \mathcal{F}, \ell_\infty^{X^m}\right) \text{ and} \tag{7.30}$$

$$\max_{Z \in (\mathcal{X} \times [a,b])^m} \mathcal{N}(\epsilon, c_{\mathcal{F}}|_Z, m) \leq \max_{X \in \mathcal{X}^m} \mathcal{N}\left(\tfrac{\epsilon m}{C}, \mathcal{F}, \ell_1^{X^m}\right). \tag{7.31}$$

2. Suppose that for some $C, \tilde{C} > 0$, c satisfies the "approximate Lipschitz–condition"

$$c(\xi) - c(\xi') \leq \max(C|\xi - \xi'|, \tilde{C}) \text{ for all } \xi, \xi' \in [a - b, b - a] \tag{7.32}$$

then for all $\epsilon > \tilde{C}/C$

$$\max_{Z \in (\mathcal{X} \times [a,b])^m} \mathcal{N}(\epsilon, c_{\mathcal{F}}|_Z, m) \leq \max_{X \in \mathcal{X}^m} \mathcal{N}\left(\tfrac{\epsilon}{C}, \mathcal{F}, \ell_\infty^{X^m}\right). \tag{7.33}$$

Proof One has to show that, for any sequence Z of (x, y) pairs in $\mathcal{X} \times [a, b]$ and any functions f and g, if the restrictions of f and g to x are close, then the restrictions of l_f and l_g to Z are close. Thus, given a cover of $\mathcal{F}|_{X^m}$ one can construct a cover of $l_{\mathcal{F}|Z}$ that is no bigger. For part 1 one gets:

$$\begin{aligned}
\frac{1}{m}\left|\sum_{j=1}^m c(g(x_j) - y_j) - c(f(x_j) - y_j)\right| &\leq \frac{1}{m}\sum_{j=1}^m |c(g(x_j) - y_j) - c(f(x_j) - y_j)| \\
&\leq \frac{1}{m}\sum_{j=1}^m C|g(x_j) - f(x_j)| \\
&= \frac{C}{m}\|g(x^m) - f(x^m)\|_{\ell_1^m} && (7.34) \\
&\leq C\|g(x^m) - f(x^m)\|_{\ell_\infty^m}. && (7.35)
\end{aligned}$$

7. Anthony and Bartlett [1999] derived a similar result based on a Lipschitz condition.

In the second case one proceeds similarly

$$\frac{1}{m}\left|\sum_{j=1}^{m} c(g(x_j)-y_j)-c(f(x_j)-y_j)\right| \leq \frac{C}{m}\sum_{j=1}^{m}\max(|g(x_j)-f(x_j)|,\tilde{C}/C)$$
$$\leq C\epsilon \quad \text{for } \epsilon \geq \tilde{C}/C. \tag{7.36}$$

■

The second case can be useful, when the exact form of the cost function is not known, happens to be discontinuous, or is badly behaved in some other way.[8] It shows how down to a scale of $\tilde{C}/C$ statements about the covering numbers of the loss-function induced class can be made. Applying the result above to polynomial loss leads to the following corollary:

Corollary 7.7 Polynomial Loss Functions
Let the assumptions be as above in lemma 7.6. Then for loss functions of type

$$c(\eta) = \frac{1}{p}\eta^p \text{ with } p > 1 \tag{7.37}$$

one has $C = (b-a)^{(p-1)}$, in particular $C = (b-a)$ for $p = 2$ and therefore

$$\max_{Z \in (\mathcal{X} \times [a,b])^m} \mathcal{N}(\epsilon, c|_Z, m) \leq \max_{X \in \mathcal{X}^m} \mathcal{N}\left(\epsilon(b-a)^{1-p}, \mathcal{F}, \ell_\infty^{X^m}\right) \tag{7.38}$$

This allows one to give an example of an overall theorem for the generalization error in terms of covering numbers of $\mathcal{F}$.

Corollary 7.8 Overall Generalization Bounds
Under the conditions as stated in lemma 7.3 and 7.6 one obtains

$$\Pr\left\{\sup_{f\in\mathcal{F}} |E_m[f,c]-E[f,c]| > \epsilon\right\} \leq 12m \cdot E\left[\mathcal{N}\left(\tfrac{\epsilon}{6C}, \mathcal{F}, \ell_\infty^{2m}\right)\right] e^{-\epsilon^2 m/36}. \tag{7.39}$$

For the special case of corollary 7.7 set $C = (b-a)^{p-1}$.

One could combine the uniform convergence results with the other results quoted above to get overall bounds on generalization performance. No further explicit statement of such a result is made here since the particular uniform convergence result needed depends on the exact set-up of the learning problem. The key novelty of the next chapters, however, is the new way to compute these bounds.

7.4 VC Dimension(s)

In principle one could skip the discussion of the VC dimension completely and proceed directly to the subsequent chapter. However, for the sake of completeness (and for being able to compare the results derived in the following chapters with

8. The two cases could be combined into one by writing the conditions in terms of the modulus of continuity. For the sake of clarity, however, this was not done.

the until now existing ones) some basic definitions and results on the VC dimension are mentioned.

The reason for the popularity of the VC dimension is that until recently the task of bounding $\mathcal{N}$ or π directly was considered too difficult. Moreover Vapnik and Chervonenkis [1971] have shown that the VC dimension h, which has an elegant combinatorial interpretation, is a fundamental quantity for bounding $\mathcal{N}$ and π. Thus there was essentially no other way than to use h when stating uniform convergence bounds.

7.4.1 Definitions

Definition 7.9 VC Dimension
The VC Dimension h of a class of binary valued functions $\mathcal{F}$ with respect to a set S is defined as the maximum cardinality of a subset $X \subseteq S$ that can be shattered by $\mathcal{F}$. More formally

$$h(\mathcal{F}, S) = \max \left\{ \operatorname{card}(X) \;\middle|\; \begin{array}{l} \text{for any } y \in \{-1,1\}^{\operatorname{card}(X)} \text{ there exists an} \\ f \in \mathcal{F} \text{ with } f(x_i) = y_i \text{ for all } x_i \in X \subseteq S \end{array} \right\} \tag{7.40}$$

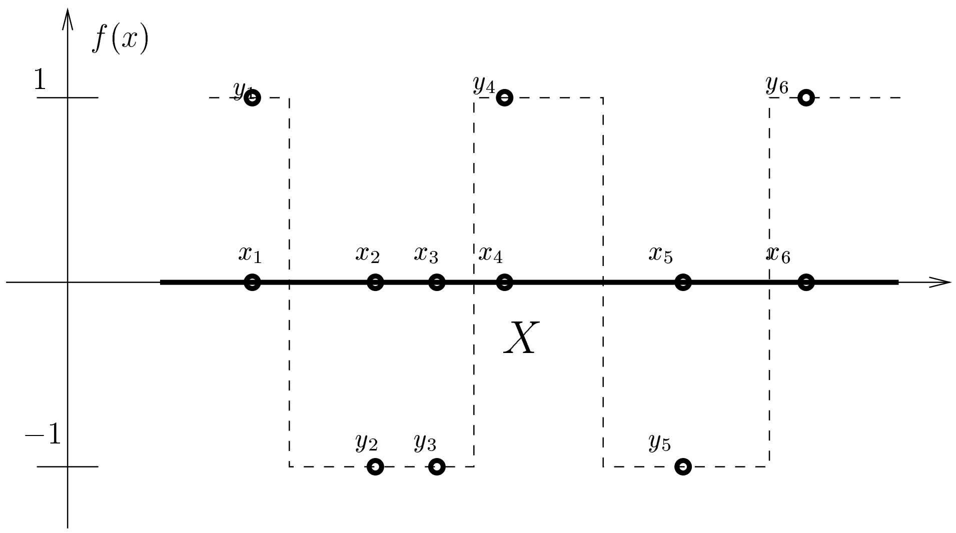

Figure 7.2 A subset X of S that is shattered (i.e. separated) by some $f \in \mathcal{F}$ for some vector $y \in \{-1;1\}^{\operatorname{card}(X)}$.

A well known example is the VC dimension of Hyperplanes in $\mathbb{R}^d$, i.e.

$$\mathcal{F} = \{f | f(x) = \operatorname{sgn}\left(\langle w, x\rangle + b\right) \text{ with } w \in \mathbb{R}^d, b \in \mathbb{R}\} \tag{7.41}$$

Its VC dimension is $h(\mathcal{F}, \mathbb{R}^d) = n + 1$ (cf. e.g. Vapnik [1982]). This is also one of the few cases where the VC dimension can be determined exactly. Note that $h = n$ for the case of Hyperplanes passing through the origin. Moreover h linear independent functions also have VC dimension h.[9] The generalization of the definition to real valued functions basically works in three different ways:

Definition 7.10 VC Dimensions for Real Valued Functions

- Like in the definition of $\pi(\mathcal{F}, m, X^m)$ for real valued functions one could consider $\mathrm{sgn}(f)$ instead of f for all $f \in \mathcal{F}$ and compute the VC dimension of this modified class of functions, i.e.

$$h_s(\mathcal{F}, \mathcal{X}) := \max\left\{\mathrm{card}(X) \,\middle|\, \begin{array}{l} \text{for any } y \in \{-1,1\}^{\mathrm{card}(X)} \text{ there exists an} \\ f \in \mathcal{F} \text{ with } \mathrm{sgn}(f(x_i)) = y_i \text{ for all } x_i \in X \subseteq \mathcal{X} \end{array}\right\} \tag{7.42}$$

- Instead of thresholding $f \in \mathcal{F}$ at 0, an alternative definition of the VC dimension for real valued functions is to allow any arbitrary threshold c and proceed as above:

$$h_l(\mathcal{F}, \mathcal{X}) := \max\left\{\mathrm{card}(X) \,\middle|\, \begin{array}{l} \text{there exists a } c \in \mathbb{R} \text{ such that for any} \\ y \in \{-1,1\}^{\mathrm{card}(X)} \text{ there exists an } f \in \mathcal{F} \text{ with} \\ \mathrm{sgn}(f(x_i) - c) = y_i \text{ for all } x_i \in X \subseteq \mathcal{X} \end{array}\right\} \tag{7.43}$$

This definition is due to Vapnik and Chervonenkis [1991], who proved that finite $h_l(\mathcal{F}, \mathcal{X})$ is necessary and sufficient to obtain uniform convergence.[10]

- Finally, instead of allowing only a scalar c wrt. which a set X has to be shattered one could also allow $c \in \mathbb{R}^{\mathrm{card}(X)}$. This yields

$$h_p(\mathcal{F}, \mathcal{X}) := \max\left\{\mathrm{card}(X) \,\middle|\, \begin{array}{l} \text{there exists a } c \in \mathbb{R}^{\mathrm{card}(X)} \text{ such that for any} \\ y \in \{-1,1\}^{\mathrm{card}(X)} \text{ there exists an } f \in \mathcal{F} \text{ with} \\ \mathrm{sgn}(f(x_i) - c_i) = y_i \text{ for all } x_i \in X \subseteq \mathcal{X} \end{array}\right\} \tag{7.44}$$

This is due to Pollard [1984], and is sometimes dubbed the *pseudo*, or 'Pollard' dimension.

Figure 7.3 shows the difference between the three definitions. Clearly $h_p(\mathcal{F}, \mathcal{X}) \geq h_l(\mathcal{F}, \mathcal{X}) \geq h_s(\mathcal{F}, \mathcal{X})$ as the definitions include each other.
The following proposition shows that neither of the above definitions of the VC dimension (i.e. $h_{\{s,l,p\}}$) are useful quantities in SV regression, neither in the case of ℓ_1, ℓ_2, nor feature space regularization.

---

9. This equality between the number of parameters (n and $n+1$) and h sometimes led to the erroneous conclusion that the number of free parameters is the decisive quantity when computing uniform convergence bounds. However this is not the case as can be seen easily from a counterexample in [Vapnik, 1995] — the family of functions $\sin(ax)$ has infinite VC dimension.
10. Observe that uniform convergence is more than one actually needs in practice — the interest is in *bounding* the maximum deviation by some positive constant.

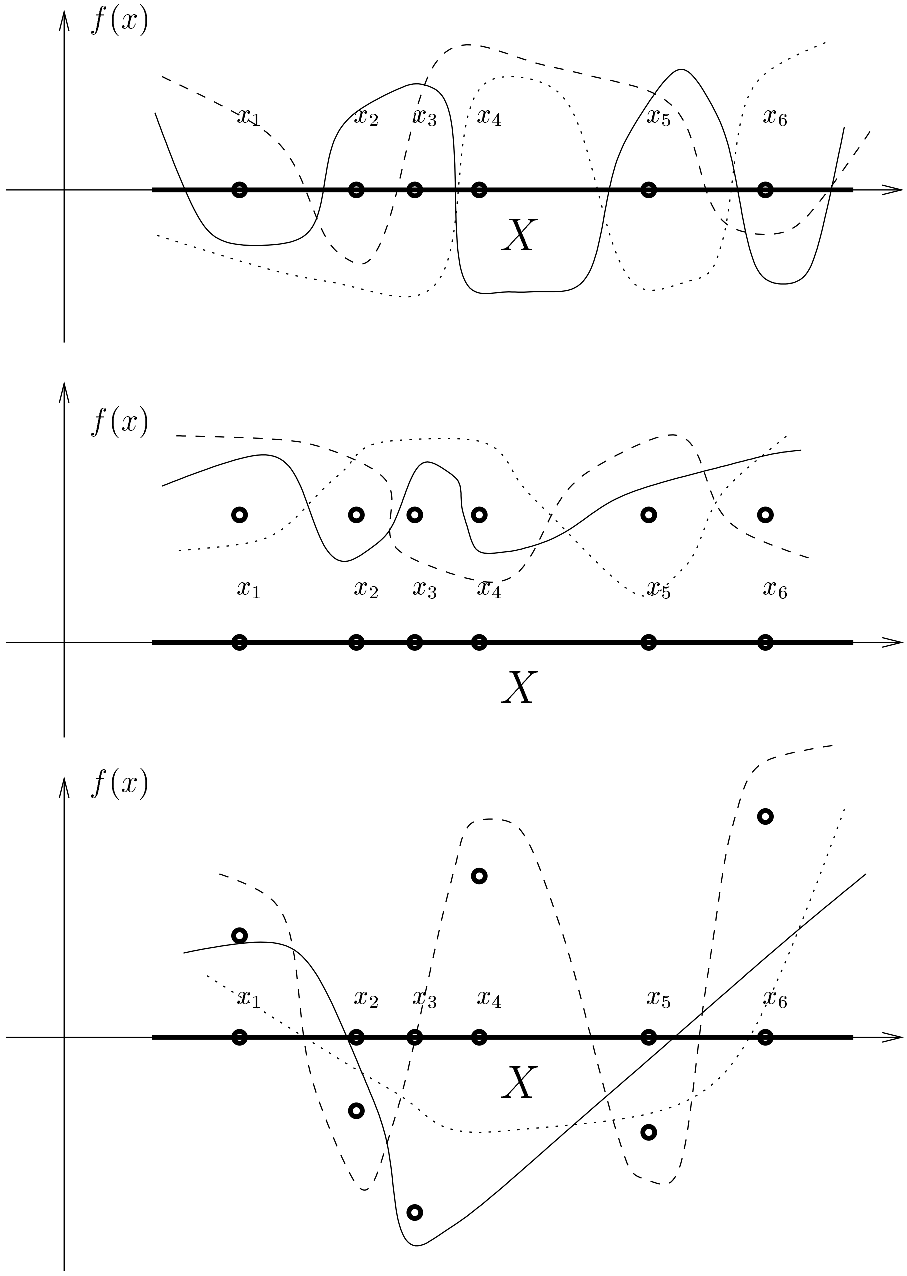

Figure 7.3 A subset X of $\mathcal{X}$ that is shattered by some real valued $f \in \mathcal{F}$. For the ease of presentation only three functions from the set are depicted (solid, dashed, and dotted). Points are classified by the fact that the funcions lie above or below the corresponding pairs $(x_i, 0), (x_i, c), (x_i, c_i)$. Top to bottom — shattering occurs according to the definitions of h_s, h_l, and h_p, hence wrt. to 0 in the upper case (h_s), wrt. an arbitrary but fixed level in the middle case (h_l), and wrt. arbitrary levels in the case on the bottom of the graph (h_p).

Proposition 7.11 Gaussian rbf–kernel with infinite VC dimension
Denote by $C \in \mathbb{R}^d$ a compact set. The class of functions

$$\mathcal{F} := \left\{ f \,\middle|\, f = \sum_i \alpha_i k(x_i, \cdot) \text{ with } x_i \in C, \sum_{i,j} \alpha_i \alpha_j k(x_i, x_j) \leq 1 \right\} \tag{7.45}$$

has infinite VC dimension.

Proof It suffices to show that any arbitrary set $Z = \{(x_1, y_1) \ldots, (x_m, y_m)\} \subset C \times \{-1, 1\}$ of size m can be shattered with a nonzero margin. Without loss of generality require that Z contains no duplicate x_i. It can be shown that there exists a function $\tilde{f} = \sum_i \tilde{\alpha}_i k(x_i, \cdot)$ such that $f(x_i) = y_i$ for all $1 \leq i \leq m$ as the matrix $k(x_i, x_j)$ has full rank for Gaussian rbf-kernels [Micchelli, 1986]. Now set $r := \sum_{i,j} \tilde{\alpha}_i \tilde{\alpha}_j k(x_i, x_j)$ and $f := r^{-1/2} \tilde{f}$ (if $r = 0$ we already have $f \in \mathcal{F}$ without rescaling). By construction $f \in \mathcal{F}$ and $f(x_i) = r^{-1/2} y_i$, hence X is shattered by the margin $r^{-1/2}$. As this holds for arbitrary m, $\mathcal{F}$ has unbounded VC dimension, even though C is compact. ■

A similar proof can be made for $\sum_i |\alpha_i| = 1$ and $\sum_i \alpha_i^2 = 1$. Consequently the VC dimension is not the appropriate quantity in SV regression as the length of the weight vector in feature space is exactly the quantity used in the SV approach. Hence it is essential to use *scale dependent* quantities like the (level) fat VC dimension, or more basic quantities like entropy and covering numbers. The latter will be done in chapter 8 by applying functional analytic tools without taking the detour via some combinatorial reasoning.

But first consider the fat–VC dimension. The basic idea is, that one may only be interested in the behaviour of a function class up to a certain resolution γ, i.e. "wiggles" below the scale γ are not considered — differences have to occur at least at a scale γ. In analogy to definition 7.10 this leads to the so called fat VC dimensions. It is a quantity that determines the complexity of the model class much more precisely, in some cases even allowing *lower* bounds on the covering numbers of model classes (cf. e.g. Bartlett et al. [1997]). Again one may think of three variants.

Definition 7.12 Fat VC Dimensions
With the same assumptions as in def. 7.10 one has for some positive scale constant γ the following cases:

- $$\text{fat–}h_l(\gamma, \mathcal{F}, \mathcal{X}) := \max \left\{ \text{card}(X) \,\middle|\, \begin{array}{l} \text{there exists a } c \in \mathbb{R} \text{ such that for any} \\ y \in \{-1, 1\}^{\text{card}(X)} \text{ there exists an } f \in \mathcal{F} \text{ with} \\ y_i(f(x_i) - c) \geq \gamma \text{ for all } x_i \in X \subseteq \mathcal{X} \end{array} \right\}$$

This quantity is sometimes also called the level fat shattering dimension. See figure 7.4 for a description of the situation.[11]

▪ What remains is a scales sensitive version of the *pseudo* dimension, sometimes also referred to simply as the fat shattering dimension.

$$\text{fat–}h_p(\gamma, \mathcal{F}, \mathcal{X}) := \max \left\{ \text{card}(X) \,\middle|\, \begin{array}{l} \text{there exists a } c \in \mathbb{R}^{\text{card}(X)} \text{ such that for any} \\ y \in \{-1, 1\}^{\text{card}(X)} \text{ there exists an } f \in \mathcal{F} \text{ with} \\ y_i(f(x_i) - c_i) \geq \gamma \text{ for all } x_i \in X \subseteq \mathcal{X} \end{array} \right\}$$

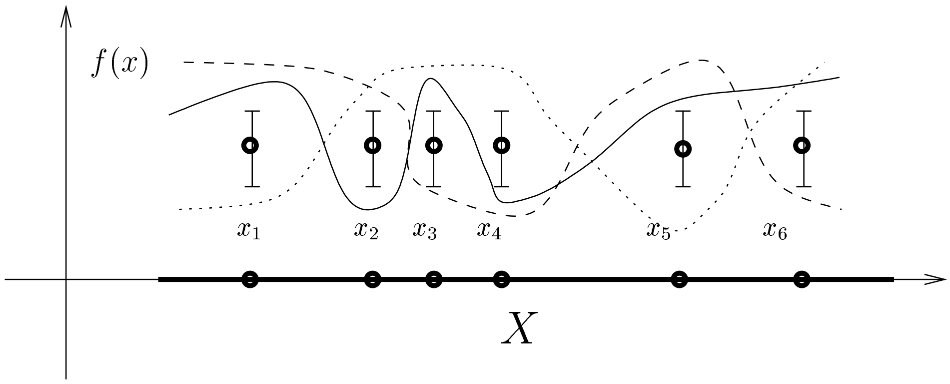

Figure 7.4 A subset X of S is level fat shattered (i.e. separated) by some $f \in \mathcal{F}$ for some vector $y \in \{-1; 1\}^{\text{card}(X)}$ and a margin (indicated by the bars). The functions pass above or below the bars. Only three examples are shown (solid, dashed, dotted), as the full number of functions ($2^6 = 64$) would have obscured the situation.

By construction the functions fat–$h(\gamma, \mathcal{F}, \mathcal{X})$ are decreasing functions of γ with $\lim_{\gamma \to 0} \text{fat–}h(\gamma, \mathcal{F}, \mathcal{X}) = h(\mathcal{F}, \mathcal{X})$. The fact that the fat VC dimension may be significantly smaller than its scale insensitive counterpart is exploited in the construction of large margin classifiers such as SV machines, Boosting and Arcing algorithms, or Mathematical Programming.

In a nutshell the idea works as follows. Assume that a (large margin) classifier $f \in \mathcal{F}$ can be found such that $y_i f(x_i) > \gamma$ (the case of classification *with* errors can be dealt with in a similar manner). Now one may replace $\mathcal{F}$ with an ε–net of precision γ. The latter has $\mathcal{N}(\gamma)$ elements. Moreover there exists some f' in the ε–net with identical classification error, i.e. $f'(x_i)y_i > 0$. Finally $\mathcal{N}(\gamma)$ (or the corresponding fat shattering dimension) can be used as the number of different functions contained in $\mathcal{F}$, thus often significantly improving the bounds. See e.g. [Bartlett and Shawe-Taylor, 1999] for more details on this subject.

11. The straightforward extension with $y_i f(x_i) \geq \epsilon$ would lead to an analogous definition of fat–h_s.

7.4.2 Bounding the Covering Number by the VC dimension

One of the reasons for the study of VC dimension(s) $h(\mathcal{F}, \mathcal{X})$ can be found in the following theorem.[12]

Theorem 7.13 Vapnik and Chervonenkis, Sauer, Shelah
The binary covering number of a class $\mathcal{F}$ can be bounded by its VC dimension (if finite) as follows

$$\pi(\mathcal{F}, m, \mathcal{X}) \leq \sum_{i=0}^{h(\mathcal{F},\mathcal{X})} \binom{m}{i} \leq 2 \frac{m^{h(\mathcal{F},\mathcal{X})}}{h(\mathcal{F}, \mathcal{X})!} \leq \left(\frac{em}{h(\mathcal{F}, \mathcal{X})}\right)^{h(\mathcal{F},\mathcal{X})} \tag{7.46}$$

This is a very useful result, as it allows to state uniform convergence bounds in terms of the VC dimension. Many error bounds for classification, e.g. those stated in [Vapnik, 1982], are derived in this manner. In the case of real valued functions similar results can be obtained. For instance for the $L_1(\mu)$ metric (where μ is an arbitrary probability measure on $\mathcal{X}$, i.e. $\mu(\mathcal{X}) = 1$) one has the following result:

Theorem 7.14 Vidyasagar [1997]
For a class of functions $\mathcal{F}$ taking on values only in the interval $[0, 1]$ with finite $h_p(\mathcal{F}, \mathcal{X}) \geq 2$ the covering number wrt. the $L_1(\mu)$ metric can be bounded as follows

$$\mathcal{N}(\epsilon, \mathcal{F}, L_1(\mu), \mathcal{X}) \leq 2 \left(\frac{2e}{\epsilon} \ln \frac{2e}{\epsilon}\right)^{h_p(\mathcal{F},\mathcal{X})} \tag{7.47}$$

In particular, for a discrete measure $\mu = \frac{1}{m}\sum_{i=1}^{m} \delta_{x_i}$ one obtains ℓ_1^m covering numbers, and subsequently, via the norm equivalence property in finite dimensional spaces, also bounds on the ℓ_m^∞ covering numbers.

What is more interesting for practical purposes, however, are bounds in terms of the scale sensitive dimensions fat–$h_{l,p}(\gamma, \mathcal{F}, \mathcal{X})$. Firstly, one needs a rule to convert fat–h_l into fat–h_p "currency", and vice versa. Secondly, a bound of $\mathcal{N}$ in terms of fat–$h_{l,p}$ is needed to apply the uniform convergence bounds. The following two results were obtained by Alon et al. [1997] (Lemma 2.2 and 3.4).

Lemma 7.15 Alon, Ben-David, Cesa-Bianchi, and Haussler [1997]
For any class $\mathcal{F}$ of $[0, 1]$ valued functions and for all $\gamma > 0$

$$\text{fat–}h_l(\gamma, \mathcal{F}, \mathcal{X}) \leq \text{fat–}h_p(\gamma, \mathcal{F}, \mathcal{X}) \leq \left(2\left\lceil\frac{1}{2\gamma}\right\rceil - 1\right) \text{fat–}h_l\left(\tfrac{\gamma}{2}, \mathcal{F}, \mathcal{X}\right) \tag{7.48}$$

Lemma 7.16 Alon, Ben-David, Cesa-Bianchi, and Haussler [1997]
For any class $\mathcal{F}$ of $[0, 1]$ valued functions and for all $\epsilon > 0$

$$\mathcal{N}(\epsilon, \mathcal{F}, \ell_\infty^m) \leq 2\left(\frac{4m}{\epsilon}\right)^{\text{fat–}h_p\left(\frac{\epsilon}{4}, \mathcal{F}, \mathcal{X}\right) \log\left(\frac{2em}{\text{fat–}h_p\left(\frac{\epsilon}{4}, \mathcal{F}, \mathcal{X}\right)\epsilon}\right)} \tag{7.49}$$

12. This result is often referred to as Sauer's lemma in Computational Learning Theory.

The key thing to note is that $\mathcal{N}$ grows essentially like $(\frac{1}{\epsilon})^{c \cdot \text{fat}-h_p}$. The level fat shattering dimension fat–$h_l(\gamma, \mathcal{F}, \mathcal{X})$ can also be used to give *lower* bounds on the covering number. At least in the case of a $L_1(\mu)$ metric (with $\mu(\mathcal{X}) = 1$) the following theorem holds.

Theorem 7.17 Bartlett, Kulkarni, and Posner [1997]
There are constants c_1 and c_2 such that, for any permissible[13] class $\mathcal{F}$ of $[0,1]$ valued functions defined on $\mathcal{X}$

$$\frac{\text{fat}-h_l(4\varepsilon, \mathcal{F}, \mathcal{X})}{32} \leq \max_{\mu} \log_2 \mathcal{N}(\varepsilon, \mathcal{F}, L_1(\mu)) \leq c_1 \text{fat}-h_l(c_2\varepsilon, \mathcal{F}, \mathcal{X})(\log_2 \varepsilon)^2. \tag{7.50}$$

This means that there exist situations where scale sensitive VC dimensions can be used to give *tight* bounds on the covering numbers. So in many cases one could restrict onseself to studying $h(\gamma, \mathcal{F}, \mathcal{X})$. However, the direct analysis of covering number is a valid goal in its own right as, possibly due to the fewer number of inequalities involved, one may be able to obtain better constants in the bounds, and the functional analytic reasoning may be technically much simpler than a combinatorial approach (however the taste may vary).

Before finishing the discourse on VC dimensions, a small technical result is needed in the case of additive combinations of model classes. This is useful when dealing with function classes plus an offset term (such as in standard SV machines), or a parametric family of models (as in semiparametric SV regression).

Corollary 7.18 Combination of Model Classes
Denote by $\mathcal{F}_1, \mathcal{F}_2$ model classes with a corresponding metric and the induced covering numbers $\mathcal{N}$. Then the covering number of the joint model class is given by

$$\mathcal{N}(\epsilon, \mathcal{F}_1 \oplus \mathcal{F}_2) \leq \mathcal{N}(\tau\epsilon, \mathcal{F}_1)\, \mathcal{N}((1-\tau)\epsilon, \mathcal{F}_2) \text{ for all } \tau \in (0,1). \tag{7.51}$$

The proof follows from the definition of $\mathcal{N}$ and the triangle inequality.

7.5 Structural Risk Minimization

The previous sections set the scene for a technique called Structural Risk Minimization (SRM) developed by Vapnik [1982, 1995]. The results obtained above can be utilized to bound the expected risk $R[f]$ for some *fixed* class of models $\mathcal{F}$. Hence, given a particular estimator and a particular dataset X one is able to determine how well the estimate will perform, based on minimal empirical error and the bound on the generalization error.

13. This is a measurability condition, cf. e.g. Pollard [1984].

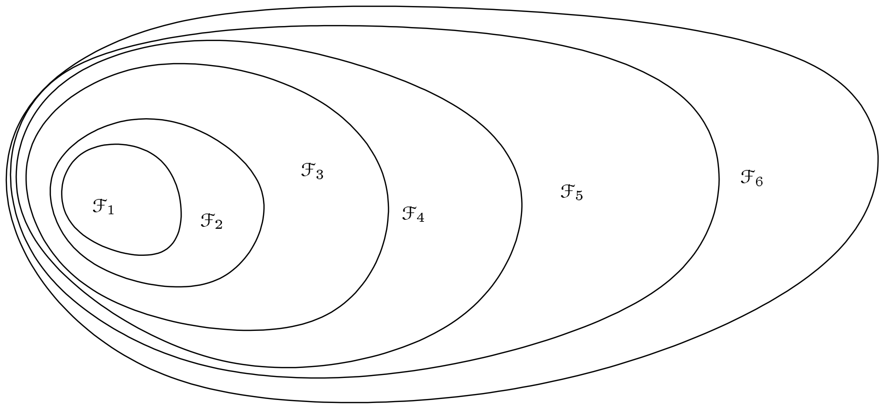

Figure 7.5 A structure defined by a nested subset of hypothesis classes $\mathcal{F}_1, \mathcal{F}_2, \ldots$.

7.5.1 The Basic Idea

What one, however, would like to do is to *choose* some $\mathcal{F}_i$ from a set of hypothesis classes $\mathfrak{F} := \{\mathcal{F}_i\}$ such that

$$R[f] \leq \min_{f \in \mathcal{F}_i} R_{\text{emp}}[f, X] + \mathcal{R}(\mathcal{F}_i, X, \delta) \tag{7.52}$$

is minimized, where $\mathcal{R}(\mathcal{F}_i, X, \delta)$, as already described in the introduction to part II, is a bound on the deviation between empirical and expected risk, obtained by the equations of section 7.3, which holds with probability $1 - \delta$. $\mathcal{F}_i$ is then chosen such that rhs of (7.52) is minimized. Note that the arrangement of sets $\mathcal{F}_i$ has to be chosen *a priori* to make this argument work.
A particularly useful choice (cf. figure 7.5) is to define $\mathfrak{F}$ such that

$$\mathcal{F}_1 \subset \mathcal{F}_2 \subset \ \ldots \ \subset \mathcal{F}_i \subset \mathcal{F}_{i+1} \subset \ldots \tag{7.53}$$

By construction the empirical risk is monotonically decreasing in this case:

$$\min_{f \in \mathcal{F}_i} R_{\text{emp}}[f, X] \leq \min_{f \in \mathcal{F}_j} R_{\text{emp}}[f, X] \quad \text{for} \quad i > j. \tag{7.54}$$

Conversely, also by construction, the bound on the generalization error keeps on increasing as the class of functions becomes richer and richer. Thus there exists some i_0, where the minimum of the overall bound on the expected risk is obtained and the minimizer of R_{emp} in $\mathcal{F}_{i_0}$ is chosen as the overall final hypothesis.

7.5.2 A Note on "Data Dependent Hierarchies" and SV machines

The problem with the structure $\mathfrak{F}$ is that it has to be chosen *before* the actual data comes into play. This, however, may constitute a problem — one has to *guess* a good structure beforehand using common sense.[14]

14. In the Bayesian approach this is often referred to as choosing a suitable prior.

One may show that from a capacity point of view, it does not make much difference whether the regularization operator P (of chapter 3) for translation invariant functions is centered at $\omega = 0$, or at some other particular frequency ω_0. In other words — there is no immediate reason why functions with frequency 0 should be the simplest ones instead of some others. Only the intuition that smooth functions might be best behaved can be referred to as a guideline.

This is probably one of the reasons why Bayesian methods work well in many cases: these have means of largely adjusting the class of hypotheses $\mathcal{F}_i$ according to the data. It is achieved by what is called multiple *hyperparameters* [MacKay, 1991], which significantly modify the shape of the function classes $\mathcal{F}_i$. Automatic Relevance Detection (ARD) (cf. [Bishop, 1995]), i.e. automatic rescaling of the inputs, thus modifying the width of the kernels, is an example thereof.

But also for bounds of VC type Shawe-Taylor et al. [1996b, 1998] introduced a framework to deal with data dependent structures, which allows more flexible hypothesis classes. Basically one has to pay an extra price (i.e. the confidence is decreased) for the additional flexibility of $\mathfrak{F}$ in terms of some model control parameters. This works quite well if only a few additional data dependent parameters exist. What is still missing, however, is a principled way to deal with this additional flexibility of the now partially ordered set of hypothesis classes (instead of having a simple inclusive structure as proposed by Vapnik [1982]). This would play the role, covering numbers take *within* a fixed hypothesis class $\mathcal{F}$.

This discourse may seem quite unrelated to practical problems, but it is not. It is (still) a heated topic of discussions in what respect SV machines need data dependent techniques to derive bounds. There are three cases where this might occur.

- Many bounds on $\mathcal{N}$ are expressed in terms of the length of the weight vector $\|w\|^2$. For fixed $C = \frac{1}{\lambda}$ this depends on the data, indeed. However fixing C was just done by for computational convenience — considering the argument in section 1.2.4 one may devise an equivalent algorithm where $\|w\|^2$ is chosen beforehand. Thus one can transform the estimates such that $\mathfrak{F}$ does not depend on X via w.
- Next, the expansion of f in terms of kernel functions depends on the training set X. However, also this point can be ruled out by considering functions in feature space. In this view it is just a coincidence (described by theorem 3.14), that f, being the optimal solution of linear functions in feature space, depends on the particular datapoints in X.
- Unfortunately the third reason cannot be come by so easily. Most current bounds on the capacity of SV machines [Guyon et al., 1993, Vapnik, 1995] make use of the fact that the data, mapped into feature space, is contained in some region (a ball or a box). Williamson et al. [1998b] and also chapter 8 state conditions under which the region in feature space can be related to a corresponding region in input space, whereas in the former works it is just *assumed* that the data lie inside a ball of radius r. Unfortunately r *is* a data *dependent* quantity, thus the corresponding tools have to be used.

- Finally, if one applies a stratification of the hypotheses in terms of the margin, i.e. if one formulates structural risk minimization over the hierarchies in terms of the margin on the data points, then clearly the hierarchy is data dependent and corresponding tools [Shawe-Taylor et al., 1996a,b,c] have to be used.

7.6 Summing Up

This chapter provided a brief overview over some empirical methods to assess the quality of an estimate f and precautions necessary when using them. The subsequent exposition served the purpose of clarifying some notions from statistical learning theory which will be needed in the following when deriving new capacity bounds for SV machines. It is one of the simplest quantities from the toolset of statistical learning theory, the covering numbers (or more precisely their functional inverse, the entropy numbers) that will be upper bounded in the following.

It was pointed out in which way covering numbers enter uniform convergence bounds. This will prove useful in later calculations. Moreover it was shown that due to the multitude of different bounds, depending on the particular setting one is analyzing, it is probably not best to state overall bounds for certain function classes but keep the two parts: bounds *on* entropy numbers, and bounds *based on* entropy numbers separate and make use of the combination only in the very last step.

Finally, VC dimensions and several variants thereof were introduced. This was done for several reasons. First, as they provided, until recently (besides few exceptions) the only way to upper bound the entropy numbers. Secondly due to their large popularity, and finally due to their simple combinatorial interpretation.

The latter is their advantage but also their weakness. Combinatorial reasonings for continuous classes of functions may require a serious amount of advanced technical tools and are difficult to formalize such that they could be used as a straightforward technique (of course, this may be a personal preference). What will be shown in the following is that continuous quantities derived by functional analytic tools may offer a way out of this dilemma, as there the reasoning can be reduced to “graph drawing” (as will be seen in chapter 9) and thus formalized in a simple way. An ulterior reason to use entropy numbers *directly* is that they involve one bounding step less than VC dimensions, hence one may hope to obtain tighter bounds.

It is up to future research to find simpler methods relying on more basic quantities such as the expected covering numbers, that would lead to even better statements.

8 Entropy Numbers, Kernels, and Operators

The present chapter presents a novel technique in bounding the covering (and entropy) numbers of kernel based expansions.[1] It is important to note that the main focus is not to obtain yet another theorem on the generalization error of kernel machines, but rather to provide building blocks which serve to improve the multitude of generalization bounds that are already presently available. SV machines are used as the model of choice to demonstrate the reasoning — statements for other settings like a restatement for regularization networks [Girosi et al., 1993], convex combinations [Bennett, 1999, Weston et al., 1999], kernel PCA [Schölkopf et al., 1998a] (see also section 5.1), or a modified principal curves algorithm [Hastie and Stuetzle, 1989, Kégl et al., 1999] (see also section 5.3) are deferred to chapter 9.

8.1 Introduction

In chapter 3 a qualitative analysis of regularization properties in terms of kernels was given (such as the correspondence between their frequency filtering characteristic and the degree of smoothness of functions). This provides insight into the regularization properties of SV kernels. However, it does not completely settle the issue of how to select a kernel for a given learning problem, and how using a specific kernel might influence the performance of a SV machine. Here, the very same methods will be exploited to make statements about the capacity of the estimators.

In particular it is shown that properties of the spectrum of the kernel can be used to make statements about the generalization error of the associated class of learning machines. Unlike in previous SV learning studies, the kernel is no longer merely a means of broadening the class of functions used, e.g. by making a nonseparable dataset separable in a feature space nonlinearly related to input space. Rather, it can be viewed as a constructive handle to control the generalization error.

A key feature is the manner in which the covering numbers can be bounded *directly*. It is achieved by viewing the relevant class of functions as the image of a unit ball under a particular compact operator. In contrast to that, the "classical" approach uses a combinatorial quantity(like the (fat–shattering) VC-dimension, and subsequent application of a general result relating the latter to covering numbers.

1. This chapter follows largely [Williamson et al., 1998b] with some minor rearrangements.

Roadmap

The chapter is organized as follows. Entropy numbers of sets and operators are introduced in section 8.2, and their connection to covering numbes is recalled.

Section 8.3 contains the main idea how to obtain bounds on the entropy numbers via functional analytic tools. For this purpose, bounds on the *shape* of feature space are obtained, based on Mercer's theorem (Th. 3.1). The connection is the decay property of the eigenvalues of integral operators corresponding to SV kernels k. These results are used to construct operators generating the class of functions common in SV machines as mappings from feature space into the class of real valued functions on $\mathcal{X}$. The proof proceeds by a factorization argument and the use of Maurey's theorem. Finally, examples of asymptotic rates (with the exact constants computed in the appendix) for entropy numbers are given in the case of given rates of decay of the eigenvalues of kernels.

Section 8.4 uses the bounds established before, however with a different approach to describe the shape of data mapped into feature space. Empirical methods, based on Kernel PCA are pointed out, and the connection to the bound of Guyon et al. [1993] is made by showing that the latter is a special case of the new methods.

The purpose of the next two sections (sec. 8.5 and sec. 8.6) is to state bounds on the rate of decay of eigenvalues in (translation invariant) integral operators, and to obtain bounds for these rates of decay. This resolves the problem when dealing with kernels such as $k(x) = e^{-x^2}$ which do not have a discrete spectrum.

Finally, section 8.7 extends the results to several dimensions, i.e. to vector valued inputs of kernels. For this purpose, some auxiliary results for degenerate systems have to be proven. The connection to the problem of sphere packing is pointed out. For the sake of concreteness, bounds for rbf kernels in multidimensional input settings are given. A summary (sec. 8.8) and an appendix with proofs concludes the chapter.

8.1.1 No Master Generalization Theorem

No single master generalization error theorem is presented for several reasons:

- The main focus of this chapter lies in the computation of covering numbers themselves.
- The particular statistical result one needs to use depends on the specific problem situation (e.g. agnostic learning).
- Many of the results obtained are in a form which, whilst quite amenable to ready computation on a computer, do not provide much direct insight by merely looking at them, except perhaps in the asymptotic sense.
- Some applications (such as classification) where further quantities like margins are estimated in a data dependent fashion, need an additional luckiness argument [Shawe-Taylor et al., 1998, Shawe-Taylor and Williamson, 1999] to apply the bounds.

Thus, although the goal has been theorems, one is ultimately forced to resort to a computer to make use of the results (at least if one is not willing to sacrifice part of the tightness of the bounds). This is not necessarily a disadvantage — it is a both a strength and a weakness of Structural Risk Minimization [Vapnik, 1982] that a good generalization error bound is both necessary and sufficient to make the method work well.

The expectation is that the refined (and significantly more tight) bound on covering numbers, obtainable by the methods presented in this chapter will be exploitable in SRM algorithms — they could be used for example for model selection. If one is running a computer program anyway, there is little point in expending a large effort to make the generalization error bounds directly consumable (but probably less tight) in a pencil and paper sense.

8.2 Entropy Numbers

Let $\mathfrak{L}(E, F)$ be the set of all bounded linear operators T between the normed spaces $(E, \|\cdot\|_E)$ and $(F, \|\cdot\|_F)$, i.e. operators such that the image of the (closed) unit ball $U_E := \{x \in E: \|x\|_E \leq 1\}$ is bounded. The smallest such bound is called the *operator norm*,

$$\|T\| := \sup_{x \in U_E} \|Tx\|_F. \tag{8.1}$$

- The nth *entropy number of a set* $M \subset E$, for $n \in \mathbb{N}$, is

$$\epsilon_n(M) := \inf \left\{ \epsilon > 0 \;\middle|\; \begin{array}{l} \text{there exists an } \epsilon\text{-cover for } M \text{ in } E \\ \text{containing } n \text{ or fewer points} \end{array} \right\} \tag{8.2}$$

Thus it is the functional inverse of the covering number $\mathcal{N}(\epsilon)$ defined in the previous chapter.

- The *entropy numbers of an operator* $T \in \mathfrak{L}(E, F)$ are defined as

$$\epsilon_n(T) := \epsilon_n(T(U_E)). \tag{8.3}$$

Note that $\epsilon_1(T) = \|T\|$, and that $\epsilon_n(T)$ certainly is well defined for all $n \in \mathbb{N}$ if T is a *compact operator*, i.e. if $T(U_E)$ is precompact.[2]

- The *dyadic entropy numbers of an operator* are defined by

$$e_n(T) := \epsilon_{2^{n-1}}(T), \text{ with } n \in \mathbb{N}; \tag{8.4}$$

Similarly, the dyadic entropy numbers of a set are defined from its entropy numbers.

2. Hence, for any $\epsilon > 0$ there exists a finite cover of $X = TU_E$ with no covering element larger than ϵ.

The *ϵ-covering number of $\mathcal{F}$ with respect to the metric d* denoted $\mathcal{N}(\epsilon, \mathcal{F}, d)$ is the smallest number of elements of an ϵ-cover of $\mathcal{F}$ using the metric d.

In this chapter, E and F will always be *Banach spaces*, i.e. complete normed spaces (for instance ℓ_p^d spaces). In some cases, they will be *Hilbert spaces* H, i.e. Banach spaces endowed with a dot product $\langle \cdot, \cdot \rangle_H$ giving rise to its norm via $\|x\|_H = \sqrt{\langle x, x \rangle_H}$.

8.3 Entropy Numbers via Functional Analysis

This section focuses on functional analytic methods to compute entropy numbers, i.e. by studying the properties of precompact operators. In particular machines, where the mapping into feature space is defined by Mercer kernels $k(x, y)$ are studied, as they are easier to deal with. Moreover the tools are introduced to construct such bounds. Section 8.4 then, in turn, will analyze with *empirical* methods to achieve the same goal.

8.3.1 The Shape of Feature Space

To analyze the shape of the image of $\Phi(\cdot)$ one can use Mercer's theorem as stated in chapter 3. From theorem 3.1, statement 2 it follows that there exists some constant $C_k \in \mathbb{R}^+$ depending on $k(\cdot, \cdot)$ such that[3]

$$|\psi_j(x)| \leq C_k \text{ for all } j \in \mathbb{N} \text{ and } x \in \mathcal{X}. \tag{8.5}$$

Moreover from statement 3 it follows that $k(x, y)$ corresponds to a dot product in ℓ_2 i.e. $k(x, y) = \langle \Phi(x), \Phi(y) \rangle_{\ell_2}$ with

$$\begin{aligned} \Phi : \mathcal{X} &\to \ell_2 \\ \Phi : x &\mapsto (\phi_j(x))_j := (\sqrt{\lambda_j}\psi_j(x))_j \end{aligned} \tag{8.6}$$

for all $x \in \mathcal{X}$. In the following (without loss of generality) assume the sequence of $(\lambda_j)_j$ be sorted in nonincreasing order (this is always possible due to the ℓ_1 summability of $(\lambda_j)_j$). From the argument above one can see that $\Phi(\mathcal{X})$ lives not only in ℓ_2 but in an axis parallel parallelepiped with lengths $2C_k\sqrt{\lambda_j}$.

It will be useful to consider maps $\Phi(\mathcal{X})$ into balls of some radius R centered at the origin. The following proposition shows that the class of all these maps is determined by elements of ℓ_2 and the sequence of eigenvalues $(\lambda_j)_j$.

3. If k is continuous and $\mathcal{X}$ is compact, then the eigenfunctions of the integral operator $Tf := \int_{\mathcal{X}} k(x, y)f(y)dy$ are also continuous. See e.g. [Ash, 1965] for a proof. Hence we may require the bounds to hold for all $x \in \mathcal{X}$ instead of ignoring sets of measure zero.

Proposition 8.1 Mapping $\Phi(x)$ into ℓ_2
Let S be the diagonal map (in sequence space)

$$\begin{aligned} &S : \mathbb{R}^{\mathbb{N}} \to \mathbb{R}^{\mathbb{N}} \\ &S : (x_j)_j \mapsto S(x_j)_j = (s_j x_j)_j \text{ with } s_j \in \mathbb{R}. \end{aligned} \tag{8.7}$$

Then S maps $\Phi(\mathcal{X})$ into a ball of finite radius R_S centered at the origin if and only if $(\sqrt{\lambda_j} s_j)_j \in \ell_2$.

Proof
[$\Leftarrow$] Suppose $(s_j\sqrt{\lambda_j})_j \in \ell_2$ and let $R_S^2 := C_k^2 \|(s_j\sqrt{\lambda_j})_j\|_{\ell_2}^2 < \infty$. For any $x \in \mathcal{X}$,

$$\|S\Phi(x)\|_{\ell_2}^2 = \sum_{j\in\mathbb{N}} s_j^2 \lambda_j |\psi_j(x)|^2 \leq \sum_{j\in\mathbb{N}} s_j^2 \lambda_j C_k^2 = R_S^2. \tag{8.8}$$

Hence $S\Phi(\mathcal{X}) \subseteq \ell_2$ and in particular $S\Phi(\mathcal{X}) \subseteq R_S U_{\ell_2}$.
[$\Rightarrow$] Suppose $(s_j\sqrt{\lambda_j})_j$ is not in ℓ_2. Hence the sequence $(A_n)_n$ with $A_n := \sum_{j=1}^{n} s_j^2 \lambda_j$ is unbounded. Now define

$$a_n(x) := \sum_{j=1}^{n} s_j^2 \lambda_j |\psi_j(x)|^2. \tag{8.9}$$

Then $\|a_n(\cdot)\|_{L_1(\mathcal{X})} = A_n$ due to the normalization condition on ψ_j. However, as $\mu(\mathcal{X}) < \infty$ there exists a set $\tilde{\mathcal{X}}$ of nonzero measure such that

$$a_n(x) \geq \frac{A_n}{\mu(\mathcal{X})} \quad \text{for all } x \in \tilde{\mathcal{X}}. \tag{8.10}$$

Combining the left side of (8.8) with (8.9) one obtains $\|S\Phi(x)\|_{\ell_2}^2 \geq a_n(x)$ for all $n \in \mathbb{N}$ and all $x \in \mathcal{X}$. Since $a_n(x)$ is unbounded for a set $\tilde{\mathcal{X}}$ with nonzero measure in $\mathcal{X}$, one can see that $S\Phi(\mathcal{X}) \not\subset \ell_2$ as there exists no ball with finite radius containing $S\Phi(\mathcal{X})$. ■

Finally, one can show that no other orthonormal basis could give better rates of decay for the sidelengths of a parallelepiped than the basis chosen from the eigenfunctions of k. This can be seen by contradiction. Assume that there exists another orthonormal basis $\{e_1', e_2', \ldots\}$ and nonnegative constants c_i such that

$$|\langle\Phi(x), e_i'\rangle| \leq c_i \text{ for all } i \in \mathbb{N} \text{ and } x \in \mathcal{X}. \tag{8.11}$$

Moreover assume that

$$\lim_{i\to\infty} \frac{c_i}{\sqrt{\lambda_i}} = 0, \tag{8.12}$$

i.e. that the new constraints decay more rapidly than those obtained from the previous proposition. It follows from (8.11) that

$$c_i^2 \geq \frac{1}{\mu(\mathcal{X})} \int_{\mathcal{X}} \langle\Phi(x), e_i'\rangle^2 dx = e_i'^{\top} \left(\frac{1}{\mu(\mathcal{X})} \int_{\mathcal{X}} \Phi(x)\Phi(x)^{\top} d\mu(x) \right) e_i'. \tag{8.13}$$

Recalling the definition of the empirical covariance matrix in feature space (5.1) one can see that integral in (8.13) is the expected covariance operator obtained from the normalized uniform measure on $\mathcal{X}$, and thus (8.13) itself the projection of the covariance operator onto the direction $_i$. As e'_i forms an orthonormal basis, (8.12) combined with (8.13) mean that the eigenvalues of the corresponding covariance matrix decay at least as fast as c_i^2 and thus faster than the eigenvalues of the inducing kernel k. This is impossible. Repeating the reasoning in section 5.1, one can see that the eigenvalues of the continuous "covariance matrix" are the eigenvalues of the integral operator T_k as defined in theorem 3.1, scaled by $\frac{1}{\mu(\mathcal{X})}$. As these are $\frac{\lambda_i}{\mu(\mathcal{X})}$, they cannot decay faster than λ_i.

The consequence of this reasoning is that there exists no *axis parallel* ellipsoid $\mathcal{E}$ not completely containing the (also) axis parallel parallelepiped $\mathcal{B}$ of sidelength $(2C_k\sqrt{\lambda_j})_j$, such that $\mathcal{E}$ would contain $\Phi(\mathcal{X})$.

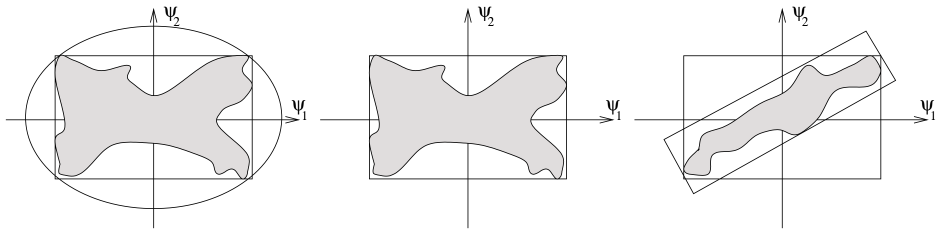

Figure 8.1 Left: mapped $\mathcal{X}$ in feature space $\mathfrak{S}$ which is contained inside a box, which again is contained inside an ellipsoid. Note that $\mathcal{X}$ fills the corners of the box. Middle and Right: observe that the mapped data in $\mathfrak{S}$ does not simply fill *some* corners such that a new coordinate system instead of ϕ_i would yield tighter boxes. In other words - the situation depicted on the right *never* occurs. Instead, the coordinate system by the eigenfunctions of the kernel is optimal up to a constant scaling factor. However, note that for actual datasets the above situation may still occur.

Hence $\Phi(\mathcal{X})$ contains a set of nonzero measure of elements near the corners of the parallelepiped.[4] This situation is described in figure 8.1. Note that the figure is only a crude approximation of the situation as the important property is that the radius has to be bounded for an infinite dimensional parallelepiped. Once it is known that $\Phi(\mathcal{X})$ "fills" the parallelepiped described above one can use this result to construct a mapping $\hat{A}$ from the unit ball U_{ℓ_2} in ℓ_2 to an ellipsoid $\mathcal{E}$ such that

4. This might be a way to improve the bounds derived in the following: one would have to compute entropy numbers of the boxes instead of the ellipsiods containing the box.

$\Phi(\mathcal{X}) \subset \mathcal{E}$ as in the following diagram.

$$\begin{tikzcd} \mathcal{X} \arrow[r, "\Phi"] & \Phi(\mathcal{X}) \subset \ell_2 \arrow[r, "\hat{A}^{-1}"] \arrow[d, phantom, "\cap"] & U_{\ell_2} \subset \ell_2 \arrow[dl, "\hat{A}"] \\ & \ell_2 \supset \mathcal{E} & \end{tikzcd} \tag{8.14}$$

The operator $\hat{A}$ will be useful for computing the entropy numbers of concatenations of operators. (Knowing the inverse will allow to compute the forward operator, and the latter can be used to bound the covering numbers of the class of functions, as shown in the next subsection.) One thus seeks an operator $\hat{A} : \ell_2 \to \ell_2$ such that

$$\hat{A}^{-1}\Phi(\mathcal{X}) \subset U_{\ell_2}. \tag{8.15}$$

This means that $\mathcal{E} := AU_{\ell_2}$ will be such that $\Phi(\mathcal{X}) \subset \mathcal{E}$. The latter can be ensured by constructing $\hat{A}$ such that

$$\hat{A}: (x_j)_j \mapsto (R_{\hat{A}} \cdot a_j \cdot x_j)_j \text{ with } R_{\hat{A}}, a_j \in \mathbb{R}^+ \tag{8.16}$$

where C_k and a_j are chosen with respect to a specific kernel and where $R_{\hat{A}} := C_k \|(\sqrt{\lambda_j}/a_j)_j\|_{\ell_2}$. From Proposition 8.1 it follows that all operators $\hat{A}$ with $R_{\hat{A}} < \infty$ satisfy (8.15). In the following, such scaling operators will be called *admissible*.

8.3.2 Functional Analysis Tools

The next step is to compute the entropy numbers of the operator A and use this to obtain bounds on the entropy numbers for kernel machines like SV machines. The following theorem due to Gordon, König and Schütt [Gordon et al., 1987, p. 226] (stated in the present form in [Carl and Stephani, 1990, p. 17]) is useful in this regard.

Theorem 8.2 Gordon, König, and Schütt [1987]
Let $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_j \geq \cdots \geq 0$ with $\sigma_i \in \mathbb{R}^+$ be a non–increasing sequence of non–negative numbers and denote

$$Dx = (\sigma_1 x_1, \sigma_2 x_2, \ldots, \sigma_j x_j, \ldots) \tag{8.17}$$

for $x = (x_1, x_2, \ldots, x_j, \ldots) \in \ell_p$ the diagonal operator from ℓ_p into itself, generated by the sequence $(\sigma_j)_j$, where $1 \leq p \leq \infty$. Then for all $n \in \mathbb{N}$,

$$\sup_{j\in\mathbb{N}} n^{-\frac{1}{j}} (\sigma_1\sigma_2\cdots\sigma_j)^{\frac{1}{j}} \leq \epsilon_n(D) \leq 6 \sup_{j\in\mathbb{N}} n^{-\frac{1}{j}} (\sigma_1\sigma_2\cdots\sigma_j)^{\frac{1}{j}}. \tag{8.18}$$

One can exploit the freedom in choosing $\hat{A}$ to minimize an entropy number as the following corollary shows. This will be a key ingredient of the calculation of the covering numbers for SV classes, as shown below.[5]

5. Guo et al. [1999] show that for Mercer kernels the minimization as required in (8.19) is well defined, i.e. there exists some operator A such that the infimum of $\epsilon_n(\hat{A})$ over all

Corollary 8.3 Entropy numbers for $\Phi(\mathcal{X})$
Let $k: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a Mercer kernel and let $\hat{A}$ be defined by (8.16). Then there exists an operator A such that for all $n \in \mathbb{N}$

$$\epsilon_n(A: \ell_2 \to \ell_2) \leq \inf_{(a_s)_s : \left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \in \ell_2} \sup_{j \in \mathbb{N}} 6C_k \left\| \left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \right\|_{\ell_2} n^{-\frac{1}{j}} (a_1 a_2 \cdots a_j)^{\frac{1}{j}}. \tag{8.19}$$

This result follows immediately by identifying D and A and exploiting the freedom in choosing a particular operator A among the class of admissible ones. With some abuse of notation we will assume that A is the operator where (8.19) attains its inf.

As already described section 1.1 and 3 the hypotheses that a SV machine generates can be written as $\langle w, x\rangle + b$ where both w and x are defined in feature space $\mathfrak{S} = \operatorname{span}(\Phi(\mathcal{X}))$ and $b \in \mathbb{R}$. The "$+b$" term can be dealt with by using corollary 7.18 and treating b as a separate model class; for now consider the class

$$\mathcal{F}_\Lambda := \{f_w : \mathfrak{S} \to \mathbb{R} \text{ with } f_w(x) = \langle w, x\rangle,\ w \in \mathfrak{S},\ \|w\| \leq \Lambda\} \subseteq \mathbb{R}^{\mathfrak{S}}. \tag{8.20}$$

Note that $\mathcal{F}_\Lambda$ depends implicitly on k since $\mathfrak{S}$ does. What one seeks are the ℓ_∞^m covering numbers for the class $\mathcal{F}_\Lambda$ induced by the kernel in terms of the parameter Λ which is the inverse of the size of the margin in feature space, or equivalently, the size of the weight vector in feature space as defined by the dot product in $\mathfrak{S}$ (see [Vapnik and Chervonenkis, 1974, Vapnik, 1995] for details). In the following, such hypothesis classes with length constraint on the weight vectors in feature space will be called *SV classes*. Denote T the operator $T = S_{x^m}\Lambda$ where $\Lambda \in \mathbb{R}^+$ and the operator S_{x^m} is defined by

$$\begin{aligned} S_{x^m} : \ell_2 &\to \ell_\infty^m \\ S_{x^m} : w &\mapsto (\langle x_1, w\rangle, \ldots, \langle x_m, w\rangle). \end{aligned} \tag{8.21}$$

with $x_j \in \Phi(\mathcal{X})$ for all j. The following theorem is useful when computing entropy numbers in terms of T and A. Originally due to Maurey [1981], it was extended by Carl [1985].

Theorem 8.4 [Carl and Stephani, 1990, p. 246]
Denote by H a Hilbert space, $\mathfrak{L}(H, \ell_\infty^m)$ the space of linear operators mapping from H to ℓ_∞^m, and let $S \in \mathfrak{L}(H, \ell_\infty^m)$. Then there exists a constant $\mathfrak{c} > 0$ such that for all $m \in \mathbb{N}$

$$e_n(S) \leq \mathfrak{c}\|S\| \left(n^{-1} \log\left(1 + \frac{m}{n}\right)\right)^{1/2}. \tag{8.22}$$

An alternative proof of this result, given in Williamson et al. [1998a], provides a small explicit value for the constant: $\mathfrak{c} \leq 103$. However there is reason to believe that $\mathfrak{c}$ should be 1.86, the constant obtainable for identity maps from ℓ_2 into ℓ_∞^m.[6]

$\hat{A}$ is obtained.

6. Observe that (8.22) is just one of three terms to be considered when bounding $e_n(S)$. However, the other two terms are merely improvements of (8.22) for the case of small n

The restatement of Theorem 8.4 in terms of $\epsilon_{2^{n-1}} = e_n$ will be useful in the following. Under the assumptions above one has

$$\epsilon_n(S) \leq \mathfrak{c}\|S\| \left((\log n + 1)^{-1} \log \left(1 + \frac{m}{\log n + 1} \right) \right)^{1/2}. \tag{8.23}$$

Theorem 8.4 allows to combine the bounds on entropy numbers of A and $S_{\Phi(x)^m}$ to obtain bounds for SV classes. For this the following lemma is useful.

Lemma 8.5 [Carl and Stephani, 1990, p. 11]
Let E, F, G be Banach spaces, $R \in \mathfrak{L}(F, G)$, and $S \in \mathfrak{L}(E, F)$. Then, for $n, t \in \mathbb{N}$,

$$\epsilon_{nt}(RS) \leq \epsilon_n(R)\epsilon_t(S) \tag{8.24}$$

$$\epsilon_n(RS) \leq \epsilon_n(R)\|S\| \tag{8.25}$$

$$\epsilon_n(RS) \leq \epsilon_n(S)\|R\|. \tag{8.26}$$

Note that the latter two inequalities follow directly from (8.24) and the fact that $\epsilon_1(R) = \|R\|$ for all $R \in \mathfrak{L}(F, G)$.

Theorem 8.6 Bounds for SV classes
Let k be a Mercer kernel, let Φ be induced via (8.6) and let $T := S_{\Phi(x^m)}\Lambda$ where $S_{\Phi(x^m)}$ is given by (8.21) and $\Lambda \in \mathbb{R}^+$. Let A be defined as in corollary 8.3 and suppose $x_j = \Phi(x_j)$ for $j = 1, \dots, m$. Then the entropy numbers of T satisfy the following inequalities:

$$\epsilon_n(T) \leq \mathfrak{c}\|A\|\Lambda \log^{-1/2} n \log^{1/2} \left(1 + \tfrac{m}{\log n}\right) \tag{8.27}$$

$$\epsilon_n(T) \leq \Lambda \epsilon_n(A) \tag{8.28}$$

$$\epsilon_{nt}(T) \leq \mathfrak{c}\Lambda \log^{-1/2} n \log^{1/2} \left(1 + \tfrac{m}{\log n}\right) \epsilon_t(A) \tag{8.29}$$

where $\mathfrak{c}$ is defined as in theorem 8.4.

This result offers several options for bounding $\epsilon_n(T)$. Examples later will show that the best inequality to use depends on the rate of decay of the eigenvalues of k. The result gives effective bounds on $\mathcal{N}^m(\epsilon, \mathcal{F}_\Lambda)$ since

$$\epsilon_n(T: \ell_2 \to \ell_\infty^m) \leq \epsilon_0 \ \Rightarrow\ \mathcal{N}^m(\epsilon_0, \mathcal{F}_\Lambda) \leq n. \tag{8.30}$$

(since $e_n(S) \leq \|S\|$ by definition) and for n in the order of m or larger. This leads to rates decaying exponentially, i.e. $e_n = O(2^{-n/m})$, however this case is not interesting for learning theoretical studies, since it corresponds to overly complex models.

Proof Consider the following factorization of T:

$$\begin{tikzcd} U_{\ell_2} \subset \ell_2 \arrow[r, "T"] \arrow[d, "\Lambda"'] & \ell_\infty^m \\ \Lambda U_{\ell_2} \subset \ell_2 \arrow[r, "A"] \arrow[ur, "S_{\Phi(x^m)}"] & \Lambda \mathcal{E} \subset \ell_2 \arrow[u, "S_{(A^{-1}\Phi(x^m))}"'] \end{tikzcd} \tag{8.31}$$

The top arrow in the diagram describes the action of T, the left and diagonal part follow immediately from its definition as $T = S_{\Phi(x^m)}\Lambda$. Finally the remainder can be seen as follows: For any $x \in \mathcal{X}$

$$\langle w, \Phi(x)\rangle = \langle w, AA^{-1}\Phi(x)\rangle = \langle Aw, A^{-1}\Phi(x)\rangle. \tag{8.32}$$

The lower left triangle commutes as, since A is diagonal, it is self-adjoint and so (8.32) holds. Summing up, instead of computing the entropy number of $T = S_{\Phi(x^m)}\Lambda$ directly, which is difficult or wasteful, as the bound on $S_{\Phi(x^m)}$ does not take into account that $x \in \mathcal{E}$ but just makes the assumption of $\Phi(x) \in \rho U_{\ell_2}$ for some $\rho > 0$, T will be represented as $S_{(A^{-1}\Phi(x^m))}A\Lambda$. This is more efficient, as A is constructed in such a way that $A^{-1}\Phi(\mathcal{X}) \in U_{\ell_2}$ fills a larger proportion of the ball than just $\frac{1}{\rho}\Phi(\mathcal{X})$.

By construction of A and the Cauchy-Schwarz inequality one has $\|S_{A^{-1}\Phi(x^m)}\| = 1$. Thus applying lemma 8.5 to the factorization of T and using Theorem 8.4 proves the theorem. ■

As shall be seen in Section 8.6, one can give asymptotic rates of decay in n for $\epsilon_n(A)$. (In fact these rates on $\epsilon_n(A)$ are non-asymptotic results with explicitly evaluable constants. The computation of the latter is relegated to the appendix.) It is thus of some interest to give overall asymptotic rates of decay of $\epsilon_n(T)$ in terms of the order of $\epsilon_n(A)$.

Lemma 8.7 Rate bounds on ϵ_n
Let k be a Mercer kernel and suppose A is the scaling operator associated with it as defined by (8.16).

1. If $\epsilon_n(A) = O(\log^{-\alpha} n)$ for some $\alpha > 0$ then

$$\epsilon_n(T) = O(\log^{-(\alpha+2)} n). \tag{8.33}$$

2. If $\log \epsilon_n(A) = O(\log^{-\beta} n)$ for some $\beta > 0$ then

$$\log \epsilon_n(T) = O(\log^{-\beta} n). \tag{8.34}$$

This lemma shows that in the first case, Maurey's result (theorem 8.4) allows an improvement in the exponent of the entropy number of T, whereas in the second, it affords none (since the entropy numbers decay so fast anyway).[7] The Maurey result may still help in that case, though, for nonasymptotic n.

Proof From theorem 8.4 one knows that[8] $\epsilon_n(S) = O(\log^{-1/2} n)$. Now use (8.24), splitting the index n in the following way (ignoring $n \in \mathbb{N}$):

$$n = n^{\tau} n^{(1-\tau)} \text{ with } \tau \in (0,1). \tag{8.35}$$

For the first case this yields

$$\begin{aligned}\epsilon_n(T) &= O(\log^{-1/2} n^{\tau}) O(\log^{-\alpha} n^{\tau-1}) \\ &= \tau^{-1/2}(1-\tau)^{-\alpha} O(\log^{-(\alpha+1/2)} n) = O(\log^{-(\alpha+2)} n).\end{aligned} \tag{8.36}$$

In the second case one gets

$$\log \epsilon_n(T) = \log\left((\tau^{-2}) O(\log^{-2} n)\right) + (1-\tau)^{-\beta} O(\log^{-\beta} n) = O(\log^{-\beta} n). \tag{8.37}$$

and thus no improvement in the overall rate. ■

In a nutshell, one can always obtain rates of convergence better than those due to Maurey's theorem because one is not dealing with *arbitrary* mappings into infinite dimensional spaces. In fact, for a logarithmic dependency of $\epsilon_n(T)$ on n, the effect of the kernel is so strong that it completely dominates the $1/\epsilon^2$ behaviour for arbitrary Hilbert spaces. An example of such a kernel is $k(x, y) = \exp(-(x-y)^2)$; see Proposition 8.10 and also Sec. 8.5 for the discretization question.

8.4 Entropy Numbers via Empirical Methods

Instead of theoretically determining the shape of $\Phi(\mathcal{X})$ *a priori* one could use the training and/or test data to empirically estimate its shape and use this quantity to compute an operator B_{emp} in analogy to $\hat{A}$ (8.14). This section sketches some possible approaches — the full development of these ideas will require considerable further work.

While functional analytic tools can be used to give theoretical guarantees for convergence properties of the algorithm *independent* of the data, empirical methods are a means to take advantage of (possibly) well behaved data, thus to state tighter bounds in a *data dependent*, i.e. problem specific, fashion.

7. This result is due to the factorization of $T = T_1 T_2$. Note that here T_1 maps from ℓ_2 into ℓ_∞^m whereas $T_2 : \ell_2 \to \ell_2$. This is acceptable as the concatenation $T : \ell_2 \to \ell_\infty^m$.
8. We will use $f(x) = O(g(x))$ and $f(x) = \Omega(g(x)$ in the following way: there exist constants $c_O, c_\Omega \in \mathbb{R}^+$ such that $|f(x)| \leq c_O g(x)$ and likewise $|f(x) \geq c_\Omega g(x)$ for any x. Note the modulus in the expressions.

8.4.1 On a Bound of Vapnik

Vapnik [1995] suggested the following bound (stated in its tighter form as proven by Bartlett and Shawe-Taylor [1999]) on the fat shattering dimension fat–$h_l(\gamma)$ of SV machines.

$$\text{fat–}h_l(\gamma, \mathcal{F}, \mathcal{X}) \leq \frac{R^2\|w\|^2}{\gamma^2} \tag{8.38}$$

Here R is the radius of a ball containing the data in *feature space*, w is the weight vector in feature space, and γ the margin with respect to the classifier output.

Ignoring the fact that (8.38) has been stated for fat–h_l instead of fat–h_p (thus a weaker statement) and combining (8.38) with (7.49) one obtains a bound on the corresponding covering number $\mathcal{N}$.[9]

$$\ln \mathcal{N}(\epsilon, \mathcal{F}, \ell_\infty^m) \leq \ln 2 + \ln\left(\tfrac{4m}{\epsilon}\right) \cdot \tfrac{16R^2\|w\|^2}{\epsilon^2} \log\left(\tfrac{2em\epsilon}{R^2\|w\|^2}\right) \tag{8.39}$$

Thus $\ln \mathcal{N}$ can be bounded by an expression essentially of the order $O(\epsilon^{-2}\log\epsilon)$. This is as good as it gets with existing combinatorial reasonings. However one can do better with functional analytic tools. From theorem 8.6, (8.27) one obtains directly

$$\epsilon_n(\mathcal{F}, \mathcal{X}) \leq \mathfrak{c} R\Lambda \log^{-1/2} n \log^{1/2}\left(1 + \tfrac{m}{\log n}\right) \text{ where } \|w\| \leq \Lambda \tag{8.40}$$

A quick calculation shows that for the relevant range of interest (i.e. $\log n \leq m$) $\mathcal{N}$ can be bounded by a term of order $O(\epsilon^{-2})$, thus yields much better scaling behaviour in terms of ϵ. The important thing to note is that this improved result has been obtained by applying the simplest of all cases of functional analytic tools. This gives an idea of the things to come. Before moving on to more elaborate methods a brief note seems to be in order.

The bounds on the entropy numbers like (8.40) rely on the assumption that the image of *all* data that may eventually be fed into the estimator *is* contained in a ball of radius R. If this is not the case, additional precautions like a luckiness argument [Shawe-Taylor et al., 1998] have to be taken to give reliable bounds. This issue, although being critical, is sometimes neglected in capacity calculations.

Regarding the algorithmic part how the radius R can be found, consider [Schölkopf et al., 1995, Burges, 1998b] and the references therein. Basically one may formulate a quadratic programming problem whose solution yields both the radius of an enclosing sphere and the points on the shell.

9. Gurvits [1997] proves a result stating that the bound of Guyon et al. [1993] actually can be used to bound h_p instead of h_l.

8.4.2 Multiple Radii

The obvious step (in hindsight) to improve the bound of section 8.4.1 is to take into account that $\Phi(\mathcal{X})$ may be contained inside some ellipsoid of which the first j radii $r_1, \ldots, r_j$ are known.[10] They are assumed to be in nonincreasing order. More formally denote by $\{\mathbf{e}_1, \ldots, \mathbf{e}_{j-1}\} \subset \ell_2$ a set of orthogonal vectors pointing in the directions given by the radii and P_j the projector onto $(\mathrm{span}\{\mathbf{e}_1, \ldots, \mathbf{e}_{j-1}\})$. Due to the ellipsoid condition the following inequality holds for all x_s:

$$\sum_{t=1}^{j-1} \frac{\langle \mathbf{e}_t, \Phi(x_s)\rangle^2}{r_t^2} \leq 1 \tag{8.41}$$

and moreover $\|P_t \Phi(x_s)\|_{\ell_2} \leq r_t$ for all $1 \leq t < j$. Hence for a scaling operator ${B_{\mathrm{emp}}}^{-1}$ scaling the first $j-1$ directions $\mathbf{e}_1, \ldots, \mathbf{e}_{j-1}$ by r_t^{-1}, and the rest by r_j^{-1}, ${B_{\mathrm{emp}}}^{-1}\Phi(x)$ would still be enclosed in $\sqrt{2}U_{\ell_2}$: the first $j-1$ rescaled directions are enclosed in a sphere of radius 1 by construction, so does the rest (this was the initial assumption). Moreover the two subspaces are orthogonal, therefore the overall radius can be bounded by $\sqrt{2}$. Hence the situation is quite similar to the case where all eigenvalues have been computed analytically in an explicit fashion. Setting $r_t := r_j$ for $t > j$ and applying corollary 8.3 leads to the following upper bound on the entropy of B_{emp}

$$\epsilon_n(B_{\mathrm{emp}}) \leq 6\sqrt{2} \sup_{1 \leq t \leq j} n^{-\frac{1}{t}} (r_1 r_2 \cdots r_t)^{\frac{1}{t}} \tag{8.42}$$

The aim is now to find that particular value of n for which one may make most usage of all radius estimates, i.e. for which the sup is attained at $t = j$ (for one has the liberty of distributing the covering numbers arbitrarily between the shrinkage operator B_{emp} and the actual evaluation operator S_{x^m} as shown in section 8.3). In other words, one is looking for that particular value of j where $\sup_j$ is taken on for the smallest radius estimated. Ignoring the fact that $n \in \mathbb{N}$ for a moment one arrives at:

$$n^{-\frac{1}{j}} (r_1 r_2 \cdots r_j)^{\frac{1}{j}} = n^{-\frac{1}{j+1}} (r_1 r_2 \cdots r_j \cdot r_j)^{\frac{1}{j+1}} \tag{8.43}$$

Solving for n yields and taking $n \in \mathbb{N}$ into account yields

$$n_j = \left\lceil \frac{r_1 r_2 \cdots r_j}{r_j^j} \right\rceil \text{ with } \epsilon_{n_j}(B_{\mathrm{emp}}) \leq 6\sqrt{2} r_j \tag{8.44}$$

10. The problem with fitting an ellipsoid around *all* patterns is that this approach is very susceptible to long tailed distributions (sometimes also called outliers). This means that the difference between the growth function computed from all datapoints may be significantly larger than the annealed entropy (or the log of the expectation of the covering numbers).

This calculation is valid as n_j is a nondecreasing function of j: $\frac{n_{j+1}}{n_j} \approx \left(\frac{r_j}{r_{j+1}}\right)^j$. If this assumption failed to hold one would have to redefine B_{emp} to scale only the first $\tilde{j}$ directions for which this happened to hold — it gains one little to scale in directions where the decay rate is too slow. Thus one can apply (8.29) of theorem 8.6 accordingly to get

$$\epsilon_n(T) \leq 6\sqrt{2}\mathfrak{c}\Lambda r_j \log^{-1/2} \lceil n/n_j \rceil \log^{1/2}\left(1 + \frac{m}{\log\lceil n/n_j \rceil}\right) \text{ with } \|w\| \leq \Lambda \tag{8.45}$$

where n_j is defined in (8.44). Essentially the difference between (8.40) and (8.45) lies in the fact that the term $\|w\|R$ has been replaced by $\|w\|r_j$ at the expense that instead of n one has to use n/n_j. See Schölkopf et al. [1999a] for a detailed reasoning and experimental results.

8.4.3 How to Estimate Multiple Radii

Unfortunately the same problems as already discussed in section 8.4.1 occur — one has to compute or *estimate* the radii r_i, and provide a statistical argument why this can be done reliably. For the latter see [Shawe-Taylor and Williamson, 1999]. There, a bound on the covering numbers of a $2m$ sample in terms of the m sample is derived and applied to the problem of empirically computing covering numbers.

The question of reliability of the estimate can be avoided in the following case. Assume one has both training data and (unlabelled) test data at hand. Then the radii of the enclosing ellipsoid can be computed from both datasets, thus giving a reliable upper bound.

The next option is that one may have additional (unlabelled) data, drawn from the same probability distribution, at hand, however not from the test set. Then this also can be used to give better estimates, however now only with some probability $1-\eta$. Finally, if real data is scarce, but, say, one knows that it is contained in some compact set $\mathcal{C}$, one might use this information to artificially generate additional data in a Monte–Carlo like fashion, and get upper bounds on the radii by doing so.

This is also useful when no analytic expansion in terms of eigenvalues of the operator can be obtained, or where it would be too tedious to obtain explicitly. In cases with a sufficient amount of computational power available this may even be a more practical and faster way than computing the spectrum given by k analytically. The latter, at least in order to obtain optimal bounds, would have to be done for each learning problem anew. The method proposed here thus would obviate the need for such detailed theoretical calculations which may be impractical to carry out in some instances. Hence the problem boils down to numerically computing (bounding) the eigenvalues of an integral operator defined by a SV kernel. Existing methods in this domain could (and should) be leveraged.

But now back to the case of an m–sample of datapoints $x := \{x_1, \ldots, x_m\} \subset \mathcal{X}$, not necessarily only from the training. A possible algorithm based on properties of Kernel-PCA (cf. Schölkopf et al. [1998a] and section 5.1) is sketched in the following. The crucial thing to observe is that one just needs a rough estimate of the directions

$\mathbf{e}_1, \ldots, \mathbf{e}_{j-1}$ (bad estimates simply will yield too large r_i but not lead to wrong results). These can be given by considering the first j principal directions obtained from Kernel-PCA. While the latter only makes statements on the second order moments of the distribution of the images of $\mathcal{X}$ in feature space, a rapid decrease of eigenvalues (which was noticed quite often) means that also the maximum projections onto $\mathbf{e}_j$ have to decrease rapidly (this follows directly from the norm equivalence property in $\mathbb{R}^m$), as they are the rescaled entries of the eigenvectors in coefficient space. Thus, possibly after some ordering, $\sqrt{\lambda_i} \geq r_i = \sqrt{\lambda_i} \max_j |\alpha_{ij}|$.

Finally, another fact worth while noticing is that effectively one only needs to estimate the quantities $(r_1 r_2 \cdots r_j)/r_j^j$ and r_j, but not all single radii separately.

8.5 Discrete Spectra of Convolution Operators

The results presented in the previous two sections show that, knowing the eigenvalue sequence $(\lambda_i)_i$ of a compact operator, one can bound its entropy numbers. Whilst it is usually possible to assume that the *data* fed into a SV machine have bounded support, the same can not be said of the kernel $k(\cdot, \cdot)$; a commonly used kernel is $k(x, y) = \exp(-(x-y)^2)$, which has noncompact support. The induced integral operator

$$(T_k f)(x) = \int_{-\infty}^{\infty} k(x, y) f(y) dy \tag{8.46}$$

then has a continuous spectrum and thus T_k is not compact [Ash, 1965, p.267]. The question arises: can one make use of such kernels in SV machines and still obtain generalization error bounds of the form developed above? A further motivation stems from the fact that by a theorem of Widom [1964], the eigenvalue decay of any convolution operator defined on a a compact set via a kernel having compact support can decay no faster than $\lambda_j = O(e^{-j^2})$. Thus if one seeks very rapid decay of eigenvalues (with concomitantly small entropy numbers), one must use convolution kernels with noncompact support.

These issues are resolved in the present section. Before doing so, first consider the case that $\operatorname{supp} k \subseteq [-a, a]$ for some $a < \infty$. Suppose further that the data points x_j satisfy $x_j \in [-b, b]$ for all j. If $k(\cdot, \cdot)$ is a convolution kernel (i.e. $k(x, y) = k(x-y, 0)$ which allows us to write with some abuse of notation $k(x-y) := k(x-y, 0)$), then the SV hypothesis $h_k(\cdot)$ can be written

$$h_k(x) := \sum_{j=1}^{m} \alpha_j k(x, x_j) = \sum_{j=1}^{m} \alpha_j k_v(x, x_j) =: h_{k_v}(x) \tag{8.47}$$

for $v \geq 2(a + b)$ where $k_v(\cdot)$ is the v-periodic extension of $k(\cdot)$ (analogously $k_v(x-y) := k_v(x-y, 0)$):

$$k_v(x) := \sum_{j=-\infty}^{\infty} k(x - jv). \tag{8.48}$$

Now one may relate the eigenvalues of T_{k_v} to the Fourier transform of $k(\cdot)$. This is done for $d = 1$ — the general case will be stated later.

Lemma 8.8
Let $k: \mathbb{R} \to \mathbb{R}$ be a symmetric convolution kernel, let $K(\omega) = F[k(x)](\omega)$ denote the Fourier transform of $k(\cdot)$ (see (3.45)) and k_v denote the v–periodical kernel derived from k (also assume that k_v exists). Then k_v has a representation as a Fourier series with $\omega_0 := \frac{2\pi}{v}$ and

$$k_v(x-y) = \sum_{j=-\infty}^{\infty} \frac{\sqrt{2\pi}}{v} K(j\omega_0) e^{ij\omega_0 x} \tag{8.49}$$
$$= \frac{\sqrt{2\pi}}{v} K(0) + \sum_{j=1}^{\infty} \frac{2}{v} \sqrt{2\pi} K(j\omega_0) \cos(j\omega_0(x-y)).$$

Moreover $\lambda_j = \sqrt{2\pi} K(j\omega_0)$ for $j \in \mathbb{Z}$ and $C_k = \sqrt{\frac{2}{v}}$.

The consequence of this result is that for (periodical) translation invariant kernels with corresponding rate of decay in the Fourier spectrum one gets the same rates for the integral operator defined on a compact domain. This is the link that will be exploited when dealing with the *discrete* spectrum induced by kernels.

Proof Clearly the Fourier series coefficients K_j of k_v exist (as k_v exists) with

$$K_j := \frac{1}{\sqrt{v}} \int_{v/2}^{v/2} e^{-ij\omega_0 x} k_v(x) dx \tag{8.50}$$

and therefore by the definition of k_v and the existence of $K(\omega)$ one concludes

$$K_j = \frac{1}{\sqrt{v}} \int_{v/2}^{v/2} \sum_{b=-\infty}^{\infty} e^{-ij\omega_0 x} k(x - bv)$$
$$= \frac{1}{\sqrt{v}} \sum_{b=-\infty}^{\infty} \int_{v/2}^{v/2} e^{-ij\omega_0 x} k(x - bv) = \sqrt{\frac{2\pi}{v}} K(j\omega_0). \tag{8.51}$$

This, and the fact that $\{x \mapsto v^{-1/2} e^{ij\omega_0 x} : j \in \mathbb{Z}\}$ forms an orthogonal basis in $L_2([-\frac{v}{2}, \frac{v}{2}], \mathbb{C})$ proves (8.49). Furthermore, one is interested in real valued basis functions for $k(x-y)$. The functions

$$\begin{aligned} \psi_0(x) &:= \tfrac{1}{\sqrt{v}} \\ \psi_j(x) &:= \sqrt{\tfrac{2}{v}} \cos(j\omega_0 x) \text{ for all } j \in \mathbb{N} \\ \psi_{-j}(x) &:= \sqrt{\tfrac{2}{v}} \sin(j\omega_0 x) \text{ for all } j \in \mathbb{N} \end{aligned} \tag{8.52}$$

form an eigensystem of the integral operator defined by k_v with the corresponding eigenvalues $\sqrt{2\pi} K(j\omega_0)$. Finally one can see that $C_k = \sqrt{\frac{2}{v}}$ by computing the max over $j \in \mathbb{N}$ and $x \in [-v/2, v/2]$. ■

Thus even though T_k may not be compact, T_{k_v} may be (for instance if $(K(j\omega_0))_{j \in \mathbb{N}} \subset \ell_2$). The above lemma can be applied whenever one can form $k_v(\cdot)$

from $k(\cdot)$. Clearly $k(x) = O(x^{-(1+\epsilon)})$ for $\epsilon > 0$ suffices to ensure that the sum in (8.48) converges.

Now consider how to choose v. Note that the Riemann-Lebesgue lemma implies that for integrable $k(\cdot)$ of bounded variation (surely any kernel one would use would satisfy that assumption), one has $K(\omega) = O(1/\omega)$. There is an tradeoff in choosing v in that for large enough ω, $K(\omega)$ is a decreasing function of ω (at least as fast as $1/\omega$) and thus by Lemma 8.8, $\lambda_j = \sqrt{2\pi}K(2\pi j/v)$ is an increasing function of v. This suggests one should choose a small value of v. But a small v will lead to high empirical error (as the kernel "wraps around" and its localization properties are lost) and large C_k.

Remark 8.9

The above Lemma can be readily extended to d dimensions. Assume $k(x)$ is v-periodic in each direction ($x = (x_1, \ldots, x_d)$). Moreover denote $\mathbf{j}$ a multi index, i.e. $\mathbf{j} = (j_1, \ldots j_d)$. One gets

$$\lambda_{\mathbf{j}} = (2\pi)^{\frac{d}{2}} K(\omega_0 \mathbf{j}) = (2\pi)^{\frac{d}{2}} K(\omega_0 \|\mathbf{j}\|) \tag{8.53}$$

for radially symmetric k and finally for the eigenfunctions $C_k = (2/v)^{\frac{d}{2}}$.

Note how the choice of a different bandwidth of the kernel, i.e. letting $k^{(\sigma)}(x) := \sigma^d k(\sigma x)$, affects the eigenspectrum of the corresponding operator. $K^{(\sigma)}(\boldsymbol{\omega}) = K(\boldsymbol{\omega}/\sigma)$, hence scaling a kernel by σ means more densely spaced eigenvalues in the spectrum of the integral operator $T_{k^{(\sigma)}}$.

8.6 Entropy Numbers for Given Decay Rates

After that most of the tools have been prepared it is now time to obtain some concrete results for specific rates of decay (and kernels). In particular in this section it will become clear how the asymptotic behaviour of $\epsilon_n(A\colon \ell_2 \to \ell_2)$, where A is the scaling operator introduced before, depends on the eigenvalues of T_k.

A similar analysis has been carried out by Prosser [1966], in order to compute the entropy numbers of integral operators. However, all of his operators mapped into $L_2(\mathfrak{X}, \mathbb{C})$. Furthermore, whilst the present propositions are stated as asympotic results as his were, the proofs actually give non-asympototic information with explicit constants (see the appendix for details).

Note that one needs to sort the eigenvalues in a nonincreasing manner because of the requirements in corollary 8.3. If the eigenvalues were unsorted one could obtain far too small numbers in the geometrical mean of $\lambda_1, \ldots, \lambda_j$. Many one-dimensional kernels have nondegenerate systems of eigenvalues in which case it is straightforward to explicitly compute the geometrical means of the eigenvalues as will be shown below. Note that whilst all of the examples below are for convolution kernels, i.e. $k(x, y) = k(x - y)$, there is nothing in the formulations of the propositions themselves that requires this. When considering the d-dimensional case it will become clear that with rotationally invariant kernels, degenerate systems of eigenvalues are generic. Section 8.7.1 deals with that case in a systematic way.

Consider the special case where $(\lambda_j)_j$ decays asymptotically with some polynomial or exponential degree. In this case one can choose a sequence $(a_j)_j$ for which one may evaluate (8.19) explicitly. In this context it is understood that when referring to the eigenvalues of a kernel k, it is the eigenvalues of the induced integral operator T_k that are being referred to.

Proposition 8.10 Polynomial Decay
Let k be a Mercer kernel with eigenvalues $\lambda_j = O(j^{-(\alpha+1)})$ for some $\alpha > 0$. Then for any $\delta \in (0, \alpha/2)$ we have

$$\epsilon_n(A\colon \ell_2 \to \ell_2) = O(\ln^{-\frac{\alpha}{2}+\delta} n). \tag{8.54}$$

The rate obtained is tight within logarithmic factors and can be bounded from below by $\Omega(\ln^{-\frac{\alpha}{2}} n)$.

An example of such a kernel is $k(x) = e^{-x}$. The proof can be found in the appendix.

The next theorem covers a wide range of practically used kernels, namely those with exponential polynomial decay in their eigenvalues. For instance the Gaussian kernel $k(x) = e^{-x^2}$ has exponential quadratic decay in λ_i. The "damped harmonic oscillator" kernel $k(x) = \frac{1}{1+x^2}$ is another example, this time with just exponential decay in its eigenvalues.

Proposition 8.11 Exponential–Polynomial Decay
Suppose k is a Mercer kernel with $\lambda_j = O(e^{-\alpha j^p})$ for some $\alpha, p > 0$. Then

$$\ln \epsilon_n^{-1}(A\colon \ell_2 \to \ell_2) = O(\ln^{\frac{p}{p+1}} n). \tag{8.55}$$

The rate is tight.

See the appendix for a proof. Whilst this theorem gives the guarantees on the learning rates of estimators using such types of kernels (which is theoretically pleasing and leads to desirable sample complexity rates), it may not always be wise to use the theoretically obtained bounds. Instead, one should take advantage of the estimates based on an analysis of the distribution of the training data since the rates obtained by the latter may turn out to be far superior wrt. the theoretical predictions. (Recall the remarks at the beginning of Section 8.5 and [Schölkopf et al., 1999a].)

8.7 Higher Dimensions

8.7.1 Degenerate Systems

Computing the Fourier transform for a given kernel k yields the continuous spectrum. As pointed out in Sec. 8.5, the interesting quantity is the discrete spectrum of integral kernels defined on $\mathcal{X}$. This means that the eigenvalues are defined on the grid $\omega_0\mathbb{Z}^d$ with $\omega_0 = 2\pi/v$. Assuming $k(x)$ is rotationally invariant, so is $K(\boldsymbol{\omega})$ and therefore also the eigenvalues $\lambda_{\mathbf{j}} = (2\pi)^{\frac{d}{2}} K(\mathbf{j}\omega_0)$ as shown in Lemma 8.8. Consequently this leads to degeneracies in the point spectrum of the integral operator given by k (or k_v respectively) as all $\mathbf{j}\omega_0$ with equal length will have the same eigenvalue. In order to deal with this case efficiently one has to modify slightly theorem 8.2. The following theorem allows proper account to be taken of the multiplicity of eigenvalues, and thus allows the straightforward calculation of the sought for entropy numbers.

Theorem 8.12
Let $(s_t)_t \in \mathbb{N}^{\mathbb{N}_0}$ be an increasing sequence with $s_0 = 1$ and $(\sigma_j)_j \in \mathbb{R}^{\mathbb{N}}$ be a non–increasing sequence of non–negative numbers with

$$\sigma_{s_j} \geq \sigma_{s_{\bar{j}}} \text{ for } j < \bar{j} \text{ and } \sigma_j = \sigma_{s_t} \text{ for } s_{t-1} < j \leq s_t \tag{8.56}$$

and let

$$Dx = (\sigma_1 x_1, \sigma_2 x_2, \ldots, \sigma_j x_j, \ldots) \tag{8.57}$$

for $x = (x_1, x_2, \ldots, x_j, \ldots) \in \ell_p$ be the diagonal operator from ℓ_p into itself, generated by the sequence $(\sigma_j)_j$, where $1 \leq p \leq \infty$. Then for all $n \in \mathbb{N}$,

$$\sup_{t\in\mathbb{N}} n^{-\frac{1}{s_t}} (\sigma_1\sigma_2\cdots\sigma_{s_t})^{\frac{1}{s_t}} \leq \epsilon_n(D) \leq 6 \sup_{t\in\mathbb{N}} n^{-\frac{1}{s_t}} (\sigma_1\sigma_2\cdots\sigma_{s_t})^{\frac{1}{s_t}}. \tag{8.58}$$

See the appendix for a proof. This theorem allows to obtain a similar result to corollary 8.3.

Corollary 8.13 Entropy Numbers for Degenerate Systems
Let $k: \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a Mercer kernel and let A be defined by (8.16) with the additional restriction that the coefficients a_j match the degeneracy of λ_j, i.e. $a_{s_j} \geq a_{s_{\bar{j}}}$ for $j < \bar{j}$ and $a_j = a_{s_t}$ for $s_{t-1} < j \leq s_t$. Then

$$\begin{aligned}&\epsilon_n(A: \ell_2 \to \ell_2) \leq \\ &\inf_{(a_j)_j : \left(\sqrt{\lambda_j}/a_j\right)_j \in \ell_2} \sup_{t\in\mathbb{N}} 6C_k \left\| \left(\sqrt{\lambda_j}/a_j\right)_j \right\|_{\ell_2} n^{-\frac{1}{s_t}} (a_1 a_2 \ldots a_{s_t})^{\frac{1}{s_t}}\end{aligned} \tag{8.59}$$

This result by itself may not appear too useful. However, this is exactly what one needs for the degenerate case (it is slightly tighter than the original statement, as the sup effectively has to be carried out only over a subset of $\mathbb{N}$). Finally one has to compute the degree of multiplicity that occurs for different indices $\mathbf{j}$. For this purpose consider shells of radius r in $\mathbb{R}^d$ centered at the origin, i.e. rS^{d-1}, which contain a nonzero number of elements of $\mathbb{Z}^d$. Denote the corresponding radii by r_j and let $n(r_j, d)$ be the number of elements on these shells. Observe that $n(r, d) \neq 0$ only when $r^2 \in \mathbb{N}$. Thus

$$\begin{aligned} n(r,d) &:= |\mathbb{Z}^d \cap rS^{d-1}| \\ N(r,d) &:= \sum_{\{0\leq\rho\leq r: \rho^2\in\mathbb{N}\}} n(\rho, d). \end{aligned} \tag{8.60}$$

The determination of $n(r,d)$ is a classical problem which is completely solved by the use of the θ-series. (see e.g. Grosswald [1985]):

Theorem 8.14 Occupation Numbers of Shells
Let the formal power series $\theta(x)$ be defined by

$$\theta(x) := \sum_{j=-\infty}^{\infty} x^{j^2} = 1 + 2\sum_{j=1}^{\infty} x^{j^2}. \tag{8.61}$$

Then $(\theta(x))^d = \sum_{j=1}^{\infty} n(\sqrt{j}, d) x^j$.

This theorem allows one to readily compute $n(r,d)$ exactly; see [Williamson et al., 1998b] for some Maple code to do so.[11] Now one can construct an index of the eigenvalues which satisfies the required ordering (at least for nonincreasing functions $K(\omega)$) and one obtains the following result:

Corollary 8.15 Radially Symmetric Systems on a Lattice
Let $k\colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ be a Mercer kernel with eigenvalues given by a radially symmetric nonincreasing function on a lattice, i.e. $\lambda_{\mathbf{j}} = \lambda(\|\mathbf{j}\|)$ with $\mathbf{j} \in \mathbb{Z}^d$ and let A be defined by (8.16) with the additional restriction that the coefficients $a_{\mathbf{i}}$ have to match the degeneracy of $\lambda_{\mathbf{j}}$, i.e. $a_j = a_{s_t}$ for $s_{t-1} \le j \le s_t$. Then[12]

$$\epsilon_n(A\colon \ell_2 \to \ell_2) \le \inf_{(a_{\mathbf{j}})_{\mathbf{j}}\colon \left(\frac{\sqrt{\lambda_{\mathbf{j}}}}{a_{\mathbf{j}}}\right)_{\mathbf{j}} \in (\ell_2)^d} \sup_{t \in \mathbb{N}} 6C_k \left\| \left(\frac{\sqrt{\lambda_{\mathbf{j}}}}{a_{\mathbf{j}}}\right)_{\mathbf{j}} \right\|_{(\ell_2)^d} n^{-\frac{1}{N(r_t,d)}} \left(\prod_{q=1}^{t} a(r_q)^{n(r_q,d)}\right)^{\frac{1}{N(r_t,d)}}. \tag{8.62}$$

Note that this result, although it may seem straightforward, cannot be obtained from corollary 8.3 directly, as there the sup would have to be carried out over $\mathbb{N}$ instead of $(N(r_t,d))_t$. The different formulation allows one to compute bounds on the entropy numbers more easily.

8.7.2 Bounds for Kernels in $\mathbb{R}^d$

Recall the Fourier transforms of kernels in $\mathbb{R}^d$ as derived in section 3.4, determining the eigenvalue sequences for kernels typically used in SV machines. These can then be used to evaluate the right hand side in corollary 8.15.

In particular by using Theorem 8.14, Corollary 8.15 and Remark 8.9 one may compute the entropy numbers numerically for a particular kernel and a particular set of parameters. This may seem unsatisfactory from a theoretician's point of view. However, as the ultimate goal is to use the obtained bounds for model selection, it is desirable to obtain as tight bounds (also in the constants) as possible. Hence if

11. Note that whilst there do exist closed form asymptotic approximate formulae for $n(r,d)$ [Grosswald, 1985, p. 155], they are inordinately complicated and of little use for the purpose of bounding entropy numbers.
12. See Guo et al. [1999] for a proof that (8.62) is well defined for kernels of trace class.

much more precise bounds can be obtained by some not too expensive numerical calculation it is definitely worth while to use those instead of a theoretically nice but not sufficiently tight upper bound. The computational effort to calculate these quantities is typically negligible in comparison to training the actual learning machine.

Notwithstanding the above, in order to give a feeling for the effect of the decay of the Fourier transform of the kernel on the entropy numbers of the A operator, the exposition on entropy numbers for SV kernel machines concludes with the following general result, the proof of which is relegated to the appendix.

Proposition 8.16 Polynomial Exponential Decay in $\mathbb{R}^d$
For kernels $k(\cdot,\cdot)$ in $\mathbb{R}^d \times \mathbb{R}^d$ with $\lambda(\omega) = O(e^{-\alpha\|\omega\|^p})$ with $\alpha, p > 0$ the following bound on the entropy number of the corresponding scaling operator holds.

$$\ln \epsilon_n^{-1}(A\colon \ell_2 \to \ell_2) = O(\ln^{\frac{p}{p+d}} n) \tag{8.63}$$

8.8 Summing Up

The reasoning of this chapter provides tools to compute tighter generalization bounds of learning machines by improving the bounds on covering numbers. For this purpose methods from functional analysis and theory of Banach spaces were exploited. As an example SV kernels have been used. In particular the present approach relied on the fact that the mapping into feature space $\mathfrak{S}$, or more precisely the shape of the mapped domain, exhibits certain decay properties to ensure rapid convergence and a constraint on the size of the weight vector in feature space. It means that the corresponding algorithms have to restrict exactly this quantity to ensure good generalization performance. Exactly this is done in SV machines.

The actual application of the results, perhaps for model selection using structural risk minimization, is somewhat involved. Below several possible paths are lined out. As said before, the functional analytic *viewpoint* in this chapter is new. A codification into a single master generalization theorem would have obscured the conceptually new part, and might become obsolete once tighter bounds on the entropy numbers are obtained. Applications to other algorithms from supervised and unsupervised learning will be looked at in the next chapter. This will show the flexibility of the "feature space mathematics", as one might call the new set of methods. From the practical viewpoint, finally, the results of this chapter could be used in the following ways:

Possible Procedures to use the Results of this Chapter

Choose k and σ The kernel k may be chosen for several reasons, about which nothing additional will be said here. The choice of σ, i.e. the width of rbf kernels, should take account of the discussion in Sec. 8.5, i.e. that larger σ will lead to faster decaying eigenvalues of T_k but also to a closer spacing of the eigenvalues, since the

the domain of interest is larger as well.

Choose the period v of the kernel One suggested procedure is outlined in Sec. 8.5. Specific results for periodical functions can be obtained in this context.

Bound $\epsilon_n(A)$ This can be done using Corollary 8.3 (for the case $d = 1$) or Corollary 8.13 or 8.15 for the case $d > 1$. Some examples of this sort of calculation are given in Sec. 8.6.

Bound $\epsilon_n(T)$ Using Theorem 8.6, i.e. by combination of the bounds on $\epsilon_n(A)$ and of the evaluation operator itself.

Take account of the loss function Using Lemma 7.6 for instance, or possibly by modifying the above reasoning for other covering numbers than those with respect to the ℓ_∞^m metric.

Plug into a uniform convergence result See chapter 7 and the pointers to the literature therein. To obtain good bounds, it is equally crucial to take most advantage of the overall learning situation (e.g. regression vs. agnostic learning, convex function classes, zero loss, etc.).

8.9 Appendix: Proofs

8.9.1 Proof of Proposition 8.10

Proof The proof uses Corollary 8.3. Since $\lambda_j = O(j^{-\alpha-1}$ there exists some $\beta \in \mathbb{R}+$ with $\lambda_j \leq \beta^2 j^{-\alpha-1}$. In this case all sequences $(a_j)_j = (j^{-\frac{\tau}{2}})_j$ with $0 < \tau < \alpha$ lead to an admissible scaling property. One has

$$\left\| \left(\frac{\sqrt{\lambda_j}}{a_j} \right)_j \right\|_{\ell_2} = \beta \left\| \left(j^{\frac{\tau-\alpha-2}{2}} \right)_j \right\|_{\ell_2} = \beta \sqrt{\zeta(\alpha - \tau + 1)} \tag{8.64}$$

where $\zeta(\cdot)$ is Riemann's zeta function. Moreover one can bound $\zeta(\cdot)$ by

$$x + \gamma \leq \zeta\left(1 + \tfrac{1}{x}\right) \leq x + 1 \tag{8.65}$$

where γ is Euler's constant. The next step is to evaluate the expression

$$(a_1 a_2 \cdots a_j)^{\frac{1}{j}} = \left(\prod_{s=1}^{j} s^{-\frac{\tau}{2}} \right)^{\frac{1}{j}} = (j!)^{-\frac{\tau}{2j}} = \Gamma(j+1)^{-\frac{\tau}{2}} \tag{8.66}$$

The Gamma function $\Gamma(x)$ can be bounded as follows

$$\ln j - 1 \leq \frac{1}{j} \ln \Gamma(j+1) \leq \ln j. \tag{8.67}$$

Hence one may bound $\epsilon_n(A)$

$$\epsilon_n(A) \geq C_k \beta \inf_{\tau \in (0,\alpha)} \sup_{j \in \mathbb{N}} n^{-\frac{1}{j}} \left(\tfrac{1}{\alpha-\tau} + \gamma \right)^{\frac{1}{2}} j^{-\frac{\tau}{2}} \tag{8.68}$$

$$\epsilon_n(A) \leq 6 C_k \beta \inf_{\tau \in (0,\alpha)} \sup_{j \in \mathbb{N}} n^{-\frac{1}{j}} \left(\tfrac{1}{\alpha-\tau} + 1 \right)^{\frac{1}{2}} e^{\frac{\tau}{2}} j^{-\frac{\tau}{2}} \tag{8.69}$$

In order to avoid unneeded technicalities we will replace $\sup_{j \in \mathbb{N}}$ by $\sup_{j \in [1,\infty)}$. This is no problem when computing the upper bound, but this is an issue for the lower bound. However, $j^{-\frac{\tau}{2}}$ on $[1, \infty)$ is within a constant factor of $2^{-\frac{\tau}{2}}$ of its corresponding values on the integer domain $\mathbb{N}$, the biggest discrepancy being at $[1, 2]$.[13] Thus we may safely ignore the concern. Next we compute

$$\sup_{j \in [1,\infty)} n^{-\frac{1}{j}} j^{-\frac{\tau}{2}} = \sup_{j \in [1,\infty)} e^{-\frac{1}{j} \ln n - \frac{\tau}{2} \ln j} = \left(\tfrac{2e \ln n}{\tau} \right)^{-\frac{\tau}{2}}. \tag{8.70}$$

The maximum of the argument is obtained for $j = \frac{2 \ln n}{\tau}$, hence (8.70) holds for all $\ln n \geq \frac{\tau}{2}$, which is fine since we want to compute bounds on $\epsilon_n(A)$ as $n \to \infty$. For

13. One may show [Schölkopf et al., 1999a] that $a_{j^*+1} \leq \sup_{j \in \mathbb{N}} n^{-\frac{1}{j}} (a_1, \ldots, a_j)^{\frac{1}{j}} \leq a_{j^*}$ for that particular j^* where $\sup_{j \in \mathbb{N}}$ is actually obtained. Hence the maximum quotient a_{j+1}/a_j, which in the present case is $2^{-\frac{\tau}{2}}$, determines the value by which the bound has to be lowered in order to obtain a true lower bound.

the lower bounds on $\epsilon_n(A)$ we obtain

$$\begin{aligned}\epsilon_n(A) &\geq C_k\beta(2e)^{-\frac{\tau}{2}} \inf_{\tau\in(0,\alpha)} \left(\tfrac{1}{\alpha-\tau}+\gamma\right)^{\frac{1}{2}} \left(\tfrac{2\ln n}{\tau}\right)^{-\frac{\tau}{2}} \\ &\geq C_k\beta(2e)2^{-\frac{\tau}{2}} \inf_{\tau\in(0,\alpha)} \left(\tfrac{1}{\alpha-\tau}+\gamma\right)^{\frac{1}{2}} \inf_{\tau\in(0,\alpha)} \left(\tfrac{2\ln n}{\tau}\right)^{-\frac{\tau}{2}} \\ &= C_k\beta(2e)2^{-\frac{\tau}{2}} \left(\tfrac{1}{\alpha}+\gamma\right)^{\frac{1}{2}} \left(\tfrac{2\ln n}{\alpha}\right)^{-\frac{\alpha}{2}}. \end{aligned} \tag{8.71}$$

This shows that $\epsilon_n(A)$ is always bounded from below by $\Omega\left(\ln^{-\frac{\alpha}{2}} n\right)$. Computation of the upper bound is slightly more effort, since one has to evaluate

$$\epsilon_n(A) \leq 6C_k\beta \inf_{\tau\in(0,\alpha)} \left(\tfrac{1}{\alpha-\tau}+1\right)^{\frac{1}{2}} \left(\tfrac{2\ln n}{\tau}\right)^{-\frac{\tau}{2}}. \tag{8.72}$$

Clearly for any fixed $\tau \in (0,\alpha)$ we are able to obtain a rate of $\epsilon_n(A) = O\left(\ln^{-\frac{\tau}{2}} n\right)$, thus the theorem follows. For practical purposes a good approximation of the inf can be found as $\frac{1}{\alpha-\tau} = \ln(2\ln n)$ by computing the derivative of the argument in (8.72) wrt. τ and dropping all terms independent of τ and n. However, numerical minimization of (8.72) is more advisable when small values of $\epsilon_n(A)$ are crucial. ■

8.9.2 Proof of Proposition 8.11

For the proof one needs the following Lemma in order to bound summations by the corresponding integrals.

Lemma 8.17 Summation and Integration in $\mathbb{R}^1$
Suppose $f : \mathbb{R} \to \mathbb{R}$ is a nonincreasing integrable function. Then the following inequality holds for any $a \in \mathbb{Z}$

$$\int_a^\infty f(x)dx \leq \sum_{n=a}^\infty f(n) \leq \int_{a-1}^\infty f(x)dx. \tag{8.73}$$

Proof The proof relies on the fact that

$$f(n) \geq \int_n^{n+1} f(n)dn \geq f(n+1) \tag{8.74}$$

due to the monotonicity of f and a decomposition of the integral $\int\limits_0^\infty = \sum\limits_{n=0}^\infty \int\limits_n^{n+1}$. The lemma is a direct consequence thereof. ■

Proof: Theorem 8.11. Since $\lambda_j = O(e^{-\alpha j^p})$ there exists some $\beta \in \mathbb{R}+$ with $\lambda_j \leq \beta^2 e^{-\alpha j^p}$. Similarly to before we now use a series $(a_j)_j = e^{-\tau/2j^p}$. Then one can bound by applying lemma 8.17 we have that for any $\tau \in [0,\alpha)$,

$$\left\| \left(\frac{\sqrt{\lambda_j}}{a_j}\right)_j \right\|_{\ell_2} = \beta \left(\sum_{j=0}^\infty e^{(\tau-\alpha)j^p}\right)^{\frac{1}{2}} \begin{cases} \leq \beta\sqrt{1+\frac{\Gamma(1/p)}{p(\alpha-\tau)^{1/p}}} \\ \geq \beta\sqrt{\frac{\Gamma(1/p)}{p(\alpha-\tau)^{1/p}}} \end{cases} \tag{8.75}$$

Next we have to apply a similar bound to the product of the first j diagonal entries of the scaling operator A.

$$(a_1 a_2 \dots a_j)^{\frac{1}{j}} = e^{-\frac{1}{2j}\tau \sum_{s=1}^{j} s^p} \begin{cases} \geq e^{-\frac{\tau}{2(p+1)} j^p} \\ \leq e^{-\frac{\tau}{2(p+1)} j^p + \frac{\tau}{2j(p+1)}} \leq e^{-\frac{\tau}{2(p+1)} j^p + \frac{\tau}{2(p+1)}} \end{cases} \tag{8.76}$$

The last inequality holds since $j \in \mathbb{N}$. Next we have to compute $\sup_{j\in\mathbb{N}} n^{-\frac{1}{j}} e^{-\frac{\tau}{2(p+1)} j^p} = \sup_{j\in\mathbb{N}} e^{-\frac{1}{j}\ln n - \frac{\tau}{2(p+1)} j^p}$. Differentiation of the exponent wrt. j leads to

$$j^{-2}\ln n - \frac{\tau p}{2(p+1)} j^{p-1} = 0 \Rightarrow j^{-1}\ln n = \frac{\tau p}{2(p+1)} j^p \tag{8.77}$$

and thus

$$\sup_{j\in[1,\infty)} n^{-\frac{1}{j}} e^{-\frac{\tau}{2(p+1)} j^p} = e^{-\left(\frac{\tau}{2}\right)^{1/(p+1)} \left(\frac{p+1}{p}\ln n\right)^{\frac{p}{p+1}}}. \tag{8.78}$$

Replacing the domain from $\sup_{j\in\mathbb{N}}$ to $\sup_{j\in[1,\infty)}$ is not a problem when it comes to computing upper bounds on $\epsilon_n(A)$. As for the lower bounds, again, a similar reasoning like in the previous proof would have to be applied.[14] However, it is left out for the sake of clarity. Thus $\epsilon_n(A)$ can be bounded from below as follows

$$\begin{aligned} \epsilon_n(A) &\geq C_k\beta \inf_{\tau\in(0,\alpha)} \sqrt{\frac{\Gamma(1/p)}{p(\alpha-\tau)^{1/p}}} e^{-\left(\frac{\tau}{2}\right)^{1/(p+1)} \left(\frac{p+1}{p}\ln n\right)^{\frac{p}{p+1}}} \\ &\geq C_k\beta \inf_{\tau\in(0,\alpha)} \sqrt{\frac{\Gamma(1/p)}{p(\alpha-\tau)^{1/p}}} \inf_{\tau\in(0,\alpha)} e^{-\left(\frac{\tau}{2}\right)^{1/(p+1)} \left(\frac{p+1}{p}\ln n\right)^{\frac{p}{p+1}}} \\ &= C_k\beta \sqrt{\frac{\Gamma(1/p)}{p\alpha^{1/p}}} e^{-\left(\frac{\alpha}{2}\right)^{1/(p+1)} \left(\frac{p+1}{p}\ln n\right)^{\frac{p}{p+1}}} \end{aligned} \tag{8.79}$$

Thus a lower bound on the rate of $\log \epsilon_n$ is $O(\log^{\frac{p}{p+1}} n)$. Moreover, for the upper bound we obtain

$$\epsilon_n(A) \leq 6C_k\beta \inf_{\tau\in(0,\alpha)} \sqrt{1 + \frac{\Gamma(1/p)}{p(\alpha-\tau)^{1/p}}} e^{-\left(\frac{\tau}{2}\right)^{1/(p+1)} \left(\frac{p+1}{p}\ln n\right)^{\frac{p}{p+1}} + \frac{\tau}{2j(p+1)}} \tag{8.80}$$

Whilst evaulation of (8.80) is best carried out numerically for practical applications, one can see that for any fixed $\tau \in (0,\alpha)$ the rate of $\log \epsilon_n(A)$ can be bounded by $O(\log^{\frac{p}{p+1}} n)$, which shows that the obtained rates are tight. ■

8.9.3 Proof of Theorem 8.12

Proof The first part of the inequality follows directly from theorem 8.2 as it is a weaker statement than the original one. The second part is proven by closely

---

14. As in the previous theorem, the problem reduces to bounding the quotient a_{j^*+1}/a_{j^*} where j^* is the variable for which $\sup_{j\in\mathbb{N}}$ is obtained. However, here the quotient can only be bounded by $e^{-\frac{\tau p}{2} j^{p-1}}$. Fortunately this is of lower order than the remaining terms, hence it will not change the *rate* of the lower bounds.

mimicking the proof in [Carl and Stephani, 1990, p. 17]. Define

$$\delta(n) := 8 \sup_{t \in \mathbb{N}} n^{-\frac{1}{s_t}} (\sigma_1 \sigma_2 \cdots \sigma_{s_t})^{\frac{1}{s_t}} \tag{8.81}$$

Next one has to show that for all n there is an index s_j with $\sigma_{s_j+1} \leq \frac{\delta(n)}{4}$. For this purpose choose an index r such that $n \leq 2^{s_j+1}$ and thus $1 \leq 2n^{-1/(s_j+1)}$. Moreover one has

$$\sigma_{s_j+1} \leq (\sigma_1 \sigma_2 \cdots \sigma_{s_j+1})^{\frac{1}{s_j+1}} \tag{8.82}$$

because of the monotonicity of $(\sigma_j)_j$ and finally

$$\sigma_{s_j+1} \leq 2n^{-1/(s_j+1)} (\sigma_1 \sigma_2 \cdots \sigma_{s_j+1})^{\frac{1}{s_j+1}}. \tag{8.83}$$

Using the definition of $\delta(n)$ one thus can conclude $\sigma_{s_j+1} \leq \delta(n)/4$. If this happens to be the case for σ_1 one immediately obtains $\epsilon_n(D) \leq \sigma_1$ which proves the theorem.

If this is not the case there exists an index s_j such that $\sigma_{s_j+1} \leq \delta(n)/4 < \sigma_{s_j}$. Hence the corresponding sectional operator

$$\begin{aligned} &D_{s_j} : \ell_p \to \ell_p \text{ with} \\ &D_{s_j}(x_1, x_2, \ldots, x_{s_j}, x_{s_j+1}, \ldots) = (\sigma_1 x_1, \sigma_2 x_2, \ldots, \sigma_{s_j} x_{s_j}, 0, 0, \ldots) \end{aligned} \tag{8.84}$$

is of rank s_j and the image $D_{s_j}(U_p)$ of the closed unit ball U_p of ℓ_p is isometric to the subset $D^{(s_j)}(U_p^{(s_j)})$ of $\ell_p^{s_j}$. In any case $D_{s_j}(U_p)$ is a precompact subset of ℓ_p. So let $y_1, y_2, \ldots, y_N$ be a maximal system of elements in $D_{s_j}(U_p)$ with

$$\|y_j - y_{\bar{j}}\| > \delta(n)/2 \text{ for } j \neq \bar{j}. \tag{8.85}$$

The maximality of this system guarantees that

$$D_{s_j}(U_p) \subseteq \bigcup_{j=1}^{N} \left\{ y_j + \frac{\delta(n)}{2} U_p \right\} \tag{8.86}$$

and thus $\epsilon_N(D_{s_j}) \leq \delta(n)/2$. In order to get an estimate for $\epsilon_N(D)$ split the operator D into two parts $D = (D - D_{s_j}) + D_{s_j}$ which allows one to bound

$$\epsilon_N(D) \leq \|D - D_{s_j}\| + \epsilon_N(D_{s_j}). \tag{8.87}$$

Using $\|D - D_{s_j}\| = \sigma_{s_j+1} \leq \delta(n)/4$ and the bound on $\epsilon_N(D_{s_j})$ one arrives at

$$\epsilon_N(D) \leq \frac{3}{4} \delta(n). \tag{8.88}$$

The final step is to show that $N \leq n$ as then by substituting in the definition of $\delta(n)$ into (8.88) yields the result. This is again achieved by a comparison of volumes. Consider the sets $\{y_j + (\delta(n)/4) U_p^{s_j}\}$ as subsets of the space $\ell_p^{s_j}$ which is possible since $y_j \in D_{s_j}(U_p)$ and $D_{s_j}(U_p) = D^{(s_j)}(U_p^{s_j})$. These sets are obviously pairwise disjoint. On the other hand one has

$$\bigcup_{j=1}^{N} \left\{ y_j + \frac{\delta(n)}{4} U_p^{s_j} \right\} \subseteq D^{(s_j)}(U_p^{s_j}) + \tfrac{\delta(n)}{4} U_p^{s_j}) \subseteq 2D^{(s_j)}(U_p^{s_j}) \tag{8.89}$$

as $\delta(n)/4 < \sigma_1$. Now a comparison of the d-dimensional Euclidean volumes vol_d provides

$$N\left(\frac{\delta(n)}{4}\right)^{s_j} \mathrm{vol}_{s_j}(U_p^{s_j}) \leq 2^{s_j}\sigma_1\sigma_2\cdots\sigma_{s_j}\mathrm{vol}_{s_j}(U_p^{s_j}) \tag{8.90}$$

and therefore $N \leq (8/\delta(n))^{s_j}\sigma_1\sigma_2\cdots\sigma_{s_j}$. Using the definition of $\delta(n)$ this yields $N \leq n$. ■

8.9.4 Proof of Proposition 8.16

Proof **(sketch only)** In the following completely ignore the fact that we are actually dealing with a countable set of eigenvalues on a lattice and replace all summations by integrals without further worry.[15] Of course this is not accurate but still will yield the correct rates for the entropy numbers.

Denote $1/\Lambda := (2\pi/v)^{\frac{d}{2}}$ the size of a unit cell, i.e. $\Lambda = (v/(2\pi))^{\frac{d}{2}}$ the density of lattice points in frequency space as given in section 8.5. Then one obtains for infinitesimal volumes dV and numbers of points dN in frequency space

$$dV = S_{d-1}r^{d-1}dr \text{ and therefore } dN = \Lambda S_{d-1}r^{d-1}dr \tag{8.91}$$

(here S_{d-1} denotes the volume of the $d-1$ dimensional unit sphere) leading to

$$N(r,d) = \frac{1}{d}\Lambda S_{d-1}r^d. \tag{8.92}$$

Next introduce a scaling operator whose eigenvalues decay like $a(\omega) = e^{-\frac{\tau}{2}\|\omega\|^p}$ for $\tau \in [0,\alpha)$. It is straightforward to check that all these values lead to both useful and admissible scaling operators. Moreover one may now estimate the separate factors in (8.62).

$$\begin{aligned}\left\|\left(\frac{\sqrt{\lambda_{\mathbf{i}}}}{a_{\mathbf{i}}}\right)_{\mathbf{i}}\right\|_{\ell_2}^2 &\approx \int dN(\omega)\frac{\lambda(\omega)}{a^2(\omega)} = S_{d-1}\Lambda\int_0^\infty r^{d-1}\beta^2 e^{-(\alpha-\tau)\|\omega\|^p} \\ &= S_{d-1}\Lambda\beta^2(\alpha-\tau)^{-\frac{d}{p}}\tau\left(\tfrac{d}{p}\right)p^{-1}\end{aligned} \tag{8.93}$$

Next one has $\ln\left(n^{-\frac{1}{N(r,d)}}\right) = -\frac{d}{\Lambda S_{d-1}r_0^d}\ln n$ and

$$\begin{aligned}\ln\left(a_1\cdot a_2\cdots a_{N(r,d)}\right)^{\frac{1}{N(r,d)}} &= -\frac{d}{\Lambda S_{d-1}r_0^d}\sum_{j=1}^{N(r,d)}\ln a_j \\ &\approx dr^{-d}\int_0^r \omega^{d-1}\ln a(\omega)d\omega & (8.94) \\ &= -dr^{-d}\int_0^r \omega^{d-1}\frac{\tau}{2}\omega^p d\omega = -\frac{\tau}{2}\frac{d}{d+p}r^p. & (8.95)\end{aligned}$$

15. This is the reason why the current reasoning can only be considered as a sketch.

This leads to

$$\begin{aligned}\epsilon_n \quad &\leq \quad 6C_k\beta\sqrt{\tfrac{S_{d-1}\Lambda\Gamma\left(\frac{d}{p}\right)}{p}} \quad \times \\ &\quad \inf_{\tau\in[0,\alpha)}(\alpha-\tau)^{-\frac{d}{2p}} \sup_{r\in\mathbb{R}^+} \exp\left(-\tfrac{d}{\Lambda S_{d-1}r^d}\ln n - \tfrac{\tau}{2}\tfrac{d}{d+p}r^p\right).\end{aligned} \tag{8.96}$$

Computing the $\sup_{r\in\mathbb{R}^+}$ yields $r = \left(\frac{2}{\tau\Lambda S_{d-1}}\frac{(d+p)d}{p}\ln n\right)^{\frac{1}{d+p}}$ and therefore

$$\begin{aligned}\epsilon_n \quad &\leq \quad 6C_k\beta\sqrt{\tfrac{S_{d-1}\Lambda\Gamma\left(\frac{d}{p}\right)}{p}} \quad \times \\ &\quad \inf_{\tau\in[0,\alpha)}(\alpha-\tau)^{-\frac{d}{2p}} \exp\left(-\left(\tfrac{\tau}{2}\right)^{\frac{d}{d+p}}\left(\tfrac{(d+p)d}{p}\tfrac{\ln n}{\Lambda S_{d-1}}\right)^{\frac{p}{d+p}}\right).\end{aligned} \tag{8.97}$$

Already from this expression one can observe the rate bounds on ϵ_n. What remains to be done is to compute the $\inf_\tau$. This can be done by differentiating (8.97) w.r.t. τ. Define

$$T_n := \left(\frac{(d+p)d}{p}\frac{\ln n}{\Lambda S_{d-1}}\right)^{\frac{p}{d+p}} \tag{8.98}$$

which leads to the optimality condition on τ

$$(\alpha-\tau)\tau^{-\frac{p}{d+p}} = \frac{d+p}{2T_n p}2^{\frac{d}{d+p}} \text{ with } \tau\in(0,\alpha]. \tag{8.99}$$

This can be solved numerically. ■

9 Applications of Covering Numbers

The functional analytic tools presented in the previous chapter are not only restricted to SV kernels. The following sections contain examples how these methods may be applied to regularization networks, convex combinations of hypotheses, and principal curves type algorithms.

Roadmap

The extension to regularization networks, as done in section 9.1 is quite straightforward. The only modification is that weight vectors in feature space $\mathfrak{S}$ now become functions in a reproducing kernel Hilbert space (rkhs). Besides that, most results carry over directly. In particular, it allows to apply a result of Makovoz [1996] to state strong convergence rates for the approximation of functions in the rkhs. This is astonishing as the initial reason for dealing with covering numbers was to obtain convergence bounds. The practical implication of this result is that it allows to give approximation properties for reduced set algorithm (cf. Burges [1996], Burges and Schölkopf [1997], Osuna and Girosi [1999], Schölkopf et al. [1998b]).

A second application is presented in section 9.2. There functional analytic tools are applied to ℓ_p regularization operators. Beyond bounds for general smoothness constraints (sec. 9.2.2), which are only slightly better than existing ones, much stronger results are obtained by making explicit use of the properties of the properties of kernel expansions. In particular, these results (sec. 9.2.4) are at least as good, in terms of asymptotic rates, as those obtained for SV regularization (i.e. regularization in feature space). Finally, a brief remark on traditional weight decay shows that p–convex combinations are only recommendable for $p \leq 1$. Otherwise, under quite generic circumstances, the function classes may be unbounded (provided, one is willing to allow an infinite number of basis functions).

Finally, the new tools are used to state uniform convergence proofs for the regularized principal manifold algorithm (sec. 9.3), introduced in section 5.3. This reasoning is interesting in two respects: first it deals with multidimensional *output* data, secondly, it is an unsupervised learning algorithm. Whilst using basic techniques of [Kégl et al., 1999], one may obtain better uniform convergence bounds due to the properties of the kernel. In particular, for kernels with exponentially fast decaying eigenvalues, learning rates, optimal within log factors compared to supervised learning (cf. [Amari et al., 1997]) are obtained.

9.1 Regularization Networks

One of the immediate applications of capacity control via entropy numbers is the treatment of regularization networks. For function expansions in terms of the Green's functions one can directly apply the equivalence as described in section 3 without further ado.

9.1.1 Entropy Numbers for Regularization Networks

If this does not happen to be the case, one has to establish a connection between the length of w in feature space, the value of $\|Pf\|^2$ and the entropy number of the latter for functions expanded not in terms of the Green's functions of P^*P. Assume an arbitrary expansion

$$f(x) = \sum_i \alpha_i f_i(x). \tag{9.1}$$

In this case

$$\|Pf\|^2 = \sum_{i,j} \alpha_i \alpha_j \langle Pf_i, Pf_j \rangle. \tag{9.2}$$

All one has to show is that under some assumptions on the basis functions f_j and the regularization operator P there exists an equivalent expansion in terms of a kernel function k being the Green's function of P, such that the reasoning of SV kernels can be applied.

A property of reproducing kernel Hilbert spaces H (cf. definition 3.13) is that all $f \in H$ can be expanded in terms of kernel functions k. This holds as the latter span H. In fact one can identify H with $\mathfrak{S}$. This is the key property — the strategy for converting (9.1) into a kernel expansion now works as follows: Assume $f_i \in H$ for all i. Now each f_i can be expressed in terms of a kernel expansion, thus also linear combination thereof, i.e. f. This brings one back to the initial setting of expansions in terms of kernel functions and one may apply the tools of chapter 8 directly.

A cautionary note is necessary for the case of $f_i \notin H$. There two things may happen: either the expression $\langle f_i, f_i \rangle_H$ is unbounded, which has the consequence that the corresponding bounds become useless (this for instance is the case when using the Yuille and Grzywacz [1988] operator with Gaussian rbf–functions with kernel width σ chosen too small).

The other situation contains a more serious problem. Assume that P^*P contains a nontrivial null space, i.e. functions $f \neq 0$ with $f \in L_2(\mathcal{X})$ but $\langle Pf, Pf \rangle = \langle f, f \rangle_H = 0$. A simple example of such an operator is the gradient operator — constant functions will yield $\langle f, f \rangle_H = 0$. If some of the basis functions f_i happen to have nonvanishing projections onto the null space of H, the concept of feature space breaks down and the uniform convergence results cannot be applied any more.

9.1.2 An Application of a Result of Makovoz

The new function analytic viewpoint can also be used to give stronger statements on the approximation properties of kernel expansions. There the basic question is, how many basis functions, say $k(x_i, \cdot)$ are needed in order to approximate an arbitrary function f with ε precision. The following theorem, itself a generalization of a result of Barron [1993] (he proved a similar result with $2\varepsilon_n(G)$ replaced by 1), makes the connection between approximation properties and entropy numbers explicit.

Recall a convex combination of $G := \{g_i\}_i$ is an element of

$$\mathrm{co}_n(G) := \left\{ \sum_{i=1}^{n} \alpha_i g_i | \alpha_i \geq 0, g_i \in G \text{ for all } 1 \leq i \leq n \text{ and } \sum_{i=1}^{n} \alpha_i = 1 \right\}. \tag{9.3}$$

Moreover denote $\mathrm{co}\,(G) = \cup_{n \in \mathbb{N}} \mathrm{co}_n(G)$, $\overline{\mathrm{co}}(G)$ the closure of $\mathrm{co}\,(G)$ and H a Hilbert space with corresponding 2–norm.

Theorem 9.1 Makovoz [1996]
Suppose for all $g \in G$, $\|g\| \leq 1$, and $G \subset H$. Let $f^* \in \overline{\mathrm{co}}(G)$. Then for every $n > 1$, there exists $f_n \in \mathrm{co}_n(G)$ such that

$$\|f^* - f_n\| \leq \frac{2\varepsilon_n(G)}{\sqrt{n}} \tag{9.4}$$

In order to apply the theorem one has to translate it into the language of kernel expansions and feature spaces. An arbitrary element f^* of an RKHS can be expanded in terms of kernel functions, thus is an element of $\overline{\mathrm{co}}(G)$ for a suitably chosen G. The latter consists of a scaled version of $\Phi(\mathcal{X})$, as the images of the input data mapped into feature space $\mathfrak{S}$ span the feature space, (or in other words the RKHS). Once, the scaling factors are fixed (this depends on f^*), one may apply the theorem above.

Now recall from section 8.3.1 that $\Phi(\mathcal{X})$ can be viewed as contained in the image of the ℓ_2 unit ball U_{ℓ_2} under some operator A. Thus $G \subset c\Phi(\mathcal{X}) \subset cAU_{\ell_2}$ for some positive constant, and consequently $\varepsilon_n(G) \leq c\varepsilon_n(A)$.

The last step is to plug in the asymptotic rates on ε_n for different kernels, to obtain, e.g. for one dimensional kernels with eigenvalues decaying polynomially with an exponent of $n^{-\alpha-1}$ a rate of $n^{-\frac{\alpha}{2}}$ for ϵ_n (cf. prop. 8.10). Thus the overall rate would be $n^{-\frac{\alpha+1}{2}}$ in that case, or in other words $n^{-\frac{m}{2}}$, where m indicates the degree of differentiability. This is significantly better than the rates of Corradi and White [1995], who obtain $n^{-\frac{2m}{2m+1}}$.

Another consequence is that for reduced set algorithms, where one wants to approximate the solution found by a SV algorithm with a smaller number of basis functions from the same RKHS, one thus can determine minimal rates of convergence for given precisions. The only problem, however, is, that finding this function expansion is a hard nonlinear optimization problem with many local minima. What remains to be done is to find a greedy algorithm, like the one of

Jones [1990] to actually achieve the rates which are theoretically possible.

A possible solution for kernel expansions consists of generating an approximation using the algorithm of Jones [1992], and then truncating the expansion subsequently to achieve the rate of theorem 9.1. The latter is possible since the discrete set of kernel functions can be covered at least as efficiently as the whole class of functions $\mathcal{F}$, thus the corresponding design matrix will have very rapidly decaying corresponding eigenvalues, which will finally allow the elimination of a subset of basis functions while conserving a large part of the approximation quality.

After the application of the functional analytic tools to quadratic regularizers it is interesting to see whether the techniques can be adapted to other regularization terms like ℓ_p norms, to bound the entropy numbers of the corresponding class of functions.

9.2 Convex Combinations

Recently new methods have been proposed [Bennett, 1999, Weston et al., 1999, Bradley and Mangasarian, 1998, Frieß and Harrison, 1998] to compute SV like expansions using linear programming algorithms.[1] Moreover boosting algorithms [Schapire et al., 1997] also deal with function spaces which are convex combinations of some basis functions. Thus it seems interesting to consider function classes of the type

$$\mathcal{F}_p := \left\{ f \,\middle|\, f(x) = \sum_j \alpha_j k(x_j, x) \text{ with } \sum_j |\alpha_j|^p \leq 1 \right\} \tag{9.5}$$

and the entropy/covering numbers thereof. Naively, one could, at least in the case of SV kernel expansions, try to use bounds for SV machines (cf. chapter 8) for model selection purposes in this case, but this would be very wasteful since the shape of the hypothesis class is quite different (one is dealing with scaled convex combinations of base hypotheses represented by the kernels $k(x_i, x)$). Furthermore, the kernels may not even satisfy Mercer's condition, in which case the standard reasoning fails completely. Nevertheless, this new class of algorithms has been reported to yield competitive generalization performance which creates the need for an explanation of this effect and for some new model selection rules.

There are essentially two cases to distinguish. First, one might only be able to assume a general smoothness constraint (the case which will be discussed below). It will become clear in the following that not much can be gained beyond the bounds determined by Theorem 8.4. Conversely, if specific properties of the kernel function k are exploited, the advantage may be significant, leading to bounds on entropy numbers with even faster rates of decay in n than for the SV regularization.

1. This section is an extended version of Smola et al. [1998f].

In the following assume $p = 1$, i.e. only classes of type $\mathfrak{F} := \mathfrak{F}_1$ will be considered, as the entropy numbers for classes with $p > 1$ may be unbounded. The reason for this will be made explicit in section 9.2.3.

9.2.1 More Functional Analytic Tools

More functional analytic tools to bound entropy numbers of convex hulls in terms of the entropy numbers of the base model class are necessary. One can use a special case of [Carl et al., 1997, Proposition 4.4] to obtain

Proposition 9.2 Entropy Numbers for Convex Hulls
For all Banach spaces X and all precompact subsets $A \subset X$ satisfying the bound $\epsilon_n(A) \leq cn^{-\frac{1}{d}}$ with $c, d > 0$ there exists a constant $\rho(d)$ such that for the convex hull of A ($\operatorname{co}(A)$) the following inequality holds

$$\epsilon_{2^n}(\operatorname{co}(A)) \leq c\rho(d)n^{-\frac{1}{d}}. \tag{9.6}$$

The following proposition gives bound on entropy numbers of compact sets in finite dimensional spaces. It follows directly from volume considerations.

Proposition 9.3 Entropy numbers of compact sets
Given a d–dimensional Banach space X and a compact set C there exists a constant $c(C, X) > 0$ such that the entropy number is bounded as follows.

$$\epsilon_n(C) \leq c(C, X)\operatorname{vol}^{\frac{1}{d}}(C)n^{-\frac{1}{d}} \tag{9.7}$$

The constants depend on the geometrical properties of the space, e.g. whether C is a box or a ball.

9.2.2 General Smoothness Constraint

The strategy of proof is as follows. For kernels satisfying a Lipschitz condition the covering number of a compact set in index space can be used to determine a covering number of the set of kernel functions. Next one applies proposition 9.2 to obtain bounds on the entropy numbers of the convex combinations of kernel functions.

Now for the Lipschitz condition: the overall result will be formulated in terms of the Lipschitz constant $c_L(k, C)$ of a class of kernels

$$c_L(k, C) := \inf \{c_L | d(k(x_1, y), k(x_2, y)) \leq c_L d(x_1, x_2) \text{ for } x_1, x_2 \in C, y \in X\}. \tag{9.8}$$

Here $d(\cdot, \cdot)$ represents the metric on the index set C and the image of k respectively. All commonly used translation invariant SV kernels (e.g. $k(x, y) = \exp(-\|x - y\|^2)$) satisfy this property. A more general formulation in terms of the modulus of continuity $\omega(k, \delta, C)$ is straightforward but has been omitted for the sake of simplicity. The above considerations lead to the following proposition.

Proposition 9.4 Entropy Numbers in L_∞ Spaces
Let X be a d-dimensional Banach space, $C \subset X$ a compact index set, $k(\cdot,\cdot)$ a kernel function defined on $C \times X$ with finite $c_L(k,C)$ and $k(x,y) \leq 1$ for all $x \in C$, $y \in X$. There exists a positive constant $c(C,d,X) > 0$ such that

$$\epsilon_{2^n}(\mathcal{F}_1, L_\infty) \leq c(C,d,X) c_L(k,C) n^{-\frac{1}{d}}. \tag{9.9}$$

In other words — if the kernel satisfies a Lipschitz condition, the entropy numbers of the kernel induced class $\mathcal{F}_1$ scale like the logarithm of the entropy numbers of their index set.

Proof The first step is to compute an upper bound on $\epsilon_n(\mathcal{K}) = \epsilon_n(\mathcal{K}, L_\infty)$ where

$$\mathcal{K} := \{k(x,\cdot) | x \in C\} \tag{9.10}$$

in terms of the entropy numbers of C. By definition one has

$$\|k(x_i,\cdot) - k(x_j,\cdot)\|_{L_\infty} \leq c_L(k,C) d(x_i,x_j) \text{ for } x_i, x_j \in C \tag{9.11}$$

and therefore

$$\epsilon_n(\mathcal{K}) \leq c_L(k,C)\epsilon_n(C). \tag{9.12}$$

As one is interested in the absolute convex combination, i.e. $\mathcal{F} = \operatorname{co}(\mathcal{K} \cup -\mathcal{K})$ one has to take into account that the set of base hypotheses is at most twice as large as $\mathcal{K}$. From proposition 9.3 one may derive a bound on $\epsilon_n(C)$ to obtain

$$\epsilon_{2n}(\mathcal{K} \cup -\mathcal{K}) \leq c_L(k,C) c(C,X) \operatorname{vol}^{\frac{1}{d}}(C) n^{-\frac{1}{d}} \tag{9.13}$$

Now apply proposition 9.2 to obtain

$$\epsilon_{2^n}(\operatorname{co}(\mathcal{K} \cup -\mathcal{K})) \leq \rho(d) 2^{\frac{1}{d}} c(C,X) \operatorname{vol}^{\frac{1}{d}}(C) c_L(k,C) n^{-\frac{1}{d}}. \tag{9.14}$$

Collecting the constants into $c(C,d,X)$ gives the desired result. ∎

Next one has to bound ϵ_n on an arbitrary m-sample $X^m := \{x_1, \ldots x_m\} \subset X$, in the ℓ_∞^m metric for $\Lambda\mathcal{F}$, i.e. $\mathcal{F}$ scaled by the constant Λ. This is done in a way much similar to (8.21). Analogously to section 8.3.2 introduce the *evaluation operator* S_{X^m} as

$$\begin{aligned} S_{X^m} : L_\infty(X) &\rightarrow \ell_\infty^m \\ S_{X^m} : f &\mapsto (f(x_1), \ldots, f(x_m)) \end{aligned} \tag{9.15}$$

Note that S_{X^m} is linear and has norm 1 due to the L_∞ norm. Hence one can apply (8.24) of lemma 8.5 and theorem 8.4 to bound $\epsilon_n(S_{X^m}\Lambda\mathcal{K})$ by $\epsilon_{n_1}(S_{X^m})\epsilon_{n_2}(\Lambda\mathcal{K})$ with $n_1 n_2 \geq n$. This is formalized in the following proposition.

Proposition 9.5
The entropy number of the hypothesis class $\Lambda\mathcal{K}$ evaluated at m arbitrary points $\{x_1, \ldots, x_m\} \subset X$ in the ℓ_∞^m metric is bounded by

$$\epsilon_n(S_{X^m}(\Lambda\mathcal{K})) \leq \Lambda\tilde{c}(C, d, X)c_L(k, C) \inf_{n_1, n_2 \in \mathbb{N}, n_1 \cdot n_2 \geq n} \left(n_1^{-1} \log\left(1 + \frac{m}{n_1}\right)\right)^{\frac{1}{2}} n_2^{-\frac{1}{d}} \tag{9.16}$$

for some constant $\tilde{c}(C, d, X) > 1$. Hence the rate in ϵ_n is of order $O(n^{-\frac{1}{2}-\frac{1}{d}})$ as can be checked by carrying out the inf or by applying lemma 8.7.

Since X^m was arbitrary, one can thus again use the bounds on ϵ_n to bound $\mathcal{N}^m(\epsilon, \mathcal{F})$. Although it is certainly not explicitly obvious here, it turns out that the bounds obtained here are tighter than those in [Gurvits, 1997, Bartlett, 1998], derived using the fat-shattering dimension and a version of Maurey's theorem. A comparision of the bounds could be done via a reasoning similar to that in section 8.4.1.

The reason why no excessive care has been spent on obtaining good constants in the present case is that, as shall become clear subsequently, one can do much better by exploiting properties of kernels in a more explicit way. Still, for methods other than kernel expansions (say boosting) proposition 9.5 may constitute a significant improvement of the bounds. But before doing so, it is useful to see why p–convex combinations with $p > 1$ can be ruled out.

9.2.3 A Remark on Traditional Weight Decay

One might conjecture that a similar result could be established for p–convex combinations with $p > 1$, i.e. for any $\mathcal{F}_p$. Training large neural networks with weight decay ($p = 2$) is such a setting. However, under the assumption of an arbitrarily large number of basis functions, this conjecture is wrong. It is sufficient to show that $\mathcal{F}_p$ is unbounded in L_∞. Consider an infinite index set $I \subset C$ for which, for some other set M of nonzero measure and some constant $\kappa > 0$

$$k(x_i, x) \geq \kappa \quad \text{for all} \quad x_i \in I, x \in M. \tag{9.17}$$

An example is $k(x, y) = e^{-(x-y)^2}$. Any compact sets I, M satisfy (9.17).

$$f(x) := \sum_j \alpha_j k(x_i, x) \geq \kappa \sum_j \alpha_j \tag{9.18}$$

for $\alpha_j \geq 0, x_i \in I, x \in M$. Now let

$$f_n(\cdot) := \sum_{j=1}^n n^{-1/p} k(x_i, \cdot) \in \mathcal{F}_p. \tag{9.19}$$

By construction, the ℓ_p^n norm of the coefficients equals 1, however $f_n(x) \geq \kappa n^{1-1/p}$ for all $x \in M$. Thus

$$\lim_{n\to\infty} \|f_n\|_{L_\infty} = \infty \tag{9.20}$$

and therefore $\mathcal{F}_p$ contains unbounded elements for $p > 1$, which leads to infinitely large covering numbers for $\mathcal{F}_p$. Hence $\mathcal{F}_p$ with $p > 1$ is not a suitable choice as a hypothesis class (in the absence of further regularization). In other words — one should avoid using traditional weight decay ($p = 2$), if the number of basis functions may grow without bound.

This leads to the question why, despite the previous reasoning, weight decay has been found to work in practice. One reason is that in standard neural networks setttings the number of basis functions is limited (either by construction, via some penalty term, etc.), thus the above described situation might not occur.

Secondly, e.g. in rbf–networks, a clustering step for finding the centers is inserted before training the final weights. This means that the basis functions are sufficiently different from each other — observe that the similarity of some basis functions was explicitly exploited in the counterexample above.

Finally, also by the distance of the basis functions, penalization with a diagonal matrix is not too different from penalization via a kernel matrix (provided the widths of the basis functions is equal, and not significantly larger than the distance between the centers) — the main diagonal elements will be 1 and the off diagonal elements rather small, thus an approximation by the unit matrix is not too unrealistic.

There exists, however, a case where this reasoning might go wrong in practice. Assume one wants to modify a boosting algorithm in such a way that instead of convex combinations one would like to have p–convex combinations with $p > 1$. After iterating a sufficiently long time the situation described above might occur as the number of basis functions (i.e. weak learners) keeps on increasing with the iterations.

9.2.4 The Kernel Advantage

To get better bounds one has to take a completely different view of the kernels. The strategy is to view the hypothesis class as contained in the image of a linear operator (as in the SV case), however with possibly different constraints. It will become clear that the statements about the image of the data in feature space can be kept. The weight vector instead is not constrained any more to a ball of some fixed radius Λ but to a convex set, identical in shape to the images of the data, i.e. a box with rapidly decaying sidelengths.

In addition to the previous assumptions require that k is symmetric, bounded, and that the kernel expansion consists only of functions with $k(x_i, \cdot)$ where $x_i \in \mathcal{X}$. If one requires that the training data be constrained to some compact set $C \subset X$ one can find an expansion of k in terms of its eigenfunctions by

$$k(x, y) = \sum_i \lambda_i \phi_i(x) \phi_i(y) \tag{9.21}$$

similar to the expansion in theorem 3.1 — the assumption of positive symmetric kernel is losened to a solely symmetric kernel (and corresponding integral operator).

Thus positivity of the eigenvalues cannot be ensured any more. However the restrictions on $\phi_i(x)$ still apply. Without loss of generality, assume that the coefficients λ_i have been ordered in decreasing order of their absolute value.

Positivity of the kernel is a central condition in the SV approach but irrelevant in the Linear Programming setting (cf. (1.24)). The only thing one cannot assume in the general case any more is that the bilinear form has positive signature. But this is not a major restriction as this effect could be taken care of by a redefinition of the weight vector. The advantage is, that nearly any symmetric function $k(x, y)$ can be brought into this form. For instance B_q–splines of even order which do not satisfy Mercer's condition can be employed in linear programming type learning algorithms.

In any case functions $f \in \mathcal{F}$ can be viewed as dot products in feature space by transforming

$$f(x) = \sum_{i=1}^{\ell} \alpha_i k(x_i, x) = \sum_{i=1}^{\ell} \alpha_i \sum_j \lambda_j \phi_j(x_i)\phi_j(x) = \langle w, \Phi(x) \rangle. \tag{9.22}$$

Here w and $\Phi(x)$ are defined as follows (for SV kernels this definition coincides with the one of the chapters 3 and 8):

$$\Phi(x) := \left(\sqrt{|\lambda_1|}\phi_1(x), \sqrt{|\lambda_2|}\phi_2(x), \ldots\right) \tag{9.23}$$

$$w := \left(\sqrt{|\lambda_1|}\,\mathrm{sign}\,(\lambda_1) \sum_{i=1}^{\ell} \alpha_i \phi_1(x_i), \sqrt{|\lambda_2|}\,\mathrm{sign}\,(\lambda_2) \sum_{i=1}^{\ell} \alpha_i \phi_2(x_i), \ldots\right) \tag{9.24}$$

Moreover one can reuse the reasonings of section 8.3.1 to infer that that $\Phi(C) \cup -\Phi(C)$ is contained in some hyperellipsoid $\mathcal{E}$ in feature space. The exact shape of $\mathcal{E}$ depends on the kernel k at hand. This is exactly the property one has to take advantage of to derive good bounds. In particular, one can construct an operator A analogous to (8.14) mapping the unit ball in ℓ_2 to $\mathcal{E}$, i.e. $\mathcal{E} = AU_{\ell_2}$. Also note that the absolute convex combination, i.e. $\mathrm{co}\,(\Phi(C) \cup -\Phi(C)) \subset \mathcal{E}$ as $\mathcal{E}$ is convex, and the weight vector w is contained in a scaled version of the hyperellipsoid, i.e. $w \in \Lambda\mathcal{E}$ with $\Lambda = \sum_{i=1}^{\ell} |\alpha_i|$ by construction.

Hence the situation (see Fig. 9.1) is quite similar to the SV case of chapter 8. The mapped data is contained inside some hyperellipsoid. The weight vector, however, is constrained to a ball in the SV case and to a hyperellipsoid of the same shape as the original data in the LP case. This means that while in SV machines capacity is allocated equally in all directions, in the convex combination algorithms much capacity is allocated in those directions where the data is spread out a lot and little capacity where there is little spread. Before doing the exact calculations one has to define an appropriate sampling operator like in (8.21) or (9.15) S_{X^m}. Be $\Phi(X^m) := \{\Phi(x_1), \ldots, \Phi(x_m)\} \subset \mathcal{E}$. Then $S_{\Phi(X^m)}$ can be defined as follows:

$$\begin{array}{lll} S_{\Phi(X^m)}: U_{\ell_2} & \to & \ell_\infty^m \\ S_{\Phi(X^m)}: w & \mapsto & (\langle w, \Phi(x_1)\rangle, \ldots, \langle w, \Phi(x_m)\rangle) \end{array} \tag{9.25}$$

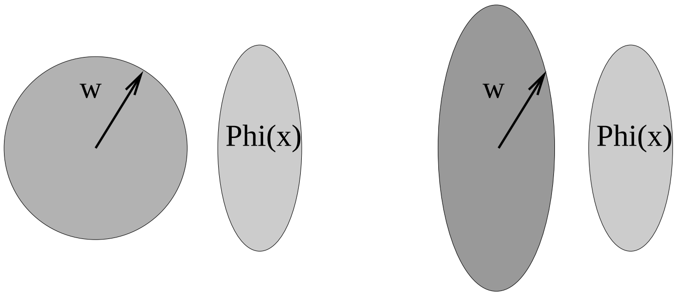

Figure 9.1 Left: In the SV case the weight vector w is contained in a ball of some (given) radius and the data lies inside some hyperellipsoid. Right: In the convex combination algorithms the weight vector is contained in a scaled version of the convex hull of the data, i.e. a hyperellipsoid of identical shape but different size.

This definition looks identical to (8.21), however the maps Φ and the vector w are defined differently. The present considerations lead to the following theorem for Linear Programming capacity control in analogy to theorem 8.6.

Theorem 9.6 Bounds for Linear Programming Machines
Let k be a symmetric bounded kernel, let Φ be induced via (9.23) and let $T := S_{\Phi(X^m)}\Lambda$ where $S_{\Phi(X^m)}$ is given by (9.25) and $\Lambda \in \mathbb{R}^+$. Let A be defined by (8.16). Then the entropy numbers of T (and hence the entropy numbers of $\Lambda\mathcal{F}_1$) satisfy the following inequalities:

$$\epsilon_n(T) \leq \mathfrak{c}\|A\|^2\Lambda \log^{-1/2} n \log^{1/2}\left(1 + \tfrac{m}{\log n}\right) \tag{9.26}$$

$$\epsilon_n(T) \leq 6\Lambda\epsilon_n(A^2) \tag{9.27}$$

$$\epsilon_{nt}(T) \leq 6\mathfrak{c}\Lambda \log^{-1/2} n \log^{1/2}\left(1 + \tfrac{m}{\log n}\right)\epsilon_t(A^2) \tag{9.28}$$

where $\mathfrak{c}$ is a constant defined in theorem 8.4.

This result is quite similar to theorem 8.6, just that the weight vector is constrained to a different set, thus the double appearance of the operator A.

Proof The proof proceeds quite similarly to the one of theorem 8.6. Equation (9.29) indicates line of reasoning which shall be followed in bounding $\epsilon_n(T\colon \ell_2 \to \ell_\infty^m)$.

$$\begin{tikzcd}
U_{\ell_2} \subset \ell_2 \arrow[rr, "T"] \arrow[d, "\Lambda"] & & \ell_\infty^m \\
\Lambda U_{\ell_2} \subset \ell_2 \arrow[r, "A"] & \Lambda\mathcal{E} = \Lambda A U_{\ell_2} \subset \ell_2 \arrow[r, "A"] \arrow[ur, "S_{\Phi(X^m)}"] & \Lambda A^2 U_{\ell_2} \subset \ell_2 \arrow[u, "S_{A^{-1}\Phi(X^m)}"']
\end{tikzcd} \tag{9.29}$$

In order to bound ℓ_∞^m entropy numbers of the hypothesis class evaluated on an $m-$sample test set X^m, one has to bound $\epsilon_n(S_{\Phi(X^m)}(\Lambda\mathrm{co}\,(\Phi(C)\cup -\Phi(C))))$. Evaluating the diagram in (9.29) one has to make one more pass (namely to the lower right) than in (8.31). Thus the diagram yields

$$\begin{aligned} S_{X^m}(\Lambda \mathrm{co}\,(\Phi(C) \cup -\Phi(C))) &\subset S_{X^m}(\Lambda \mathcal{E}) \\ &= S_{X^m}(\Lambda A U_{\ell_2}) \\ &= S_{A^{-1}X^m}(\Lambda A A U_{\ell_2}). \end{aligned} \tag{9.30}$$

where for the last step the equality $S_{X^m}Ax = S_{A^{-1}X^m}x$ was used, which holds due to the representation of f as a linear functional in some feature space. Using (9.30) and proposition 8.5 one obtains

$$\begin{aligned} \epsilon_n(S_{\Phi(X^m)}\Lambda \mathrm{co}\,(\Phi(C) \cup -\Phi(C))) &\leq \epsilon_n(T) \\ &\leq \Lambda \epsilon_n(S_{A^{-1}\Phi(X^m)}A^2) \\ &\leq \inf_{n_1,n_2 \in \mathbb{N}, n_1 n_2 \geq n} \epsilon_{n_1}(S_{A^{-1}X^m})\epsilon_{n_2}(A^2). \end{aligned} \tag{9.31}$$

Combining the factorization properties obtained above with lemma 8.5 yields the desired results: by construction, due to the Cauchy–Schwartz inequality $\|S_{A^{-1}\Phi(X^m)}\| = 1$. Since $S_{A^{-1}\Phi(X^m)}$ is an operator mapping from a Hilbert space ℓ_2 into an m–dimensional Banach space ℓ_∞^m one can use Maurey's theorem (see theorem 8.4 for the special case $p = 2$). ■

This allows one to leverage the results from chapter 8, obtained on the properties of $\mathcal{E}$ and consequently also A, for specific kernels. In particular one gets the following two propositions which follow immediately from their counterparts (propositions 8.10 and 8.16) for the case of SV regularization.

Proposition 9.7 Polynomial Decay
Let k be a symmetric kernel with eigenvalues $\lambda_j = O(j^{-(\alpha+1/2)}$ for some $\alpha > 0$ and be $\delta \in (0, \alpha)$. Then

$$\epsilon_n(A^2\colon \ell_2 \to \ell_2) == O(\ln^{-\alpha+\delta} n). \tag{9.32}$$

This result can be seen as follows. As A is a diagonal scaling operator, the scaling factors of A^2 are simply those of A squared, i.e. decaying twice as fast. Comparing the result with the one of proposition 8.10 shows that the condition on the eigenvalues was changed from $j^{-(\alpha/2+1)}$ into $j^{-(\alpha+1/2)}$. The conclusions and the method of proving this, however, remain unchanged. A similar result can be stated for exponentially decaying eigenvalues of k.

Proposition 9.8 Polynomial Exponential Decay in $\mathbb{R}^d$
For translation invariant kernels $k(x, x') = k(x - x')$ in $\mathbb{R}^d \times \mathbb{R}^d$ with Fourier transform satisfying $F[k](\omega) = O(e^{-\alpha\|\omega\|^p})$ with $\alpha, p > 0$ and corresponding operator A one has

$$\ln \epsilon_n^{-1}(A^2\colon \ell_2 \to \ell_2) = O(\ln^{\frac{p}{p+d}} n) \tag{9.33}$$

Analogous results hold for the other propositions obtained in the previous chapter. Note that whereas in the first case an improvement of the rates in n was achievable, in the latter case no such thing happened — this is due to the fact that in the latter case one is dealing with bounds on the rate of $\ln \epsilon_n$ instead of ϵ_n. It is worth while noticing that the constants, however, do change and thus make the overall bounds behave significantly better than before.

It might look like as if, due to the considerations above, linear programming machines should be preferred to Support Vector regularization machines. This need not necessarily be the case — the rate bounds only tell how "well-behaved" a certain class of models is, not how small the empirical error for a comparable bound of the generalization error might be.

What is happening is that the capacity is distributed *differently* among the class of kernel expandable functions, i.e. a different structure $\mathfrak{F}$ is chosen. More emphasis is put on the first eigenfunctions of the kernel. If one has experimental evidence that this might be useful (say, e.g. from compression experiments [Schölkopf et al., 1998c]), one should consider using such a regularizer.

9.3 Regularized Principal Manifolds and Principal Curves

Another application of the concept of covering numbers can be found in bounding the expected quantization error in principal curves type algorithms.[2] The current reasoning relies heavily on the work of Kégl et al. [1999] — in fact it just provides a "drop in" replacement of some of their results whilst keeping the main thread of their approach.

The strategy is as follows: firstly one needs uniform convergence statements for the expected quantization error in terms of its empirical counterpart and some covering number of the class of admissible functions. In the current case, similarly to [Kégl et al., 1999], $L_\infty(\ell_2^d)$ bracket covers will be used. These are determined by functional analytic methods. The novelty in the case of regularized principal manifolds is that one has to deal with vector valued functions, which requires a slight change of the methods. Finally, the combination of the previous two steps allows to state rate results for unsupervised learning of vector valued functions.

9.3.1 Basic Tools

To avoid several technicalities (like boundedness of some moments of the distribution $P(x)$ [Vapnik, 1982]), assume that there exists a ball of radius r such that $Pr\{\|x\| \leq r\} = 1$ for all x. Kégl et al. [1999] showed that under these assumptions also the prinicipal manifold γ is contained in the ball $B_r(0)$ of radius r, hence the quantization error will be no larger than $(2r)^2$ for all x. The following bounds will

2. See [Smola et al., 1999] for an extended version of the results of this section.

be stated in terms of the $L_\infty(\ell_2^d)$ metric on $\mathcal{F}$. This is defined by

$$\|\gamma\|_{L_\infty(\ell_2^d)} := \sup_{b \in \mathcal{B}} \|\gamma(b)\|_{\ell_2^d} \tag{9.34}$$

Here $\|\cdot\|_{\ell_2^d}$ denotes the Euclidean norm in d dimensions.

The next proposition is similar in its form and strategy of proof to the bounds obtained in Kégl et al. [1999], however slightly streamlined, as it is independent of some technical conditions on the class of functions $\mathcal{F}$ needed in [Kégl et al., 1999]. It serves the purpose of providing uniform convergence bounds (as in section 7.3.1), however for the case of unsupervised learning, i.e. regularized quantization functionals.

Proposition 9.9 Smola, Williamson, Mika, and Schölkopf [1999]
Denote by $\mathcal{F}$ a class of functions from $\mathcal{B}$ into $\mathcal{X} \subseteq B_r$, be P a distribution over $\mathcal{X}$, $\gamma, \gamma^*, \gamma^*_{\text{emp}} \in \mathcal{F}$, γ^* the overall optimizer of the quantization functional and γ^*_{emp} the empirical counterpart, and m the sample size. Then, for all $\eta > 0, \epsilon \in (0, \eta/2)$

$$\Pr\left\{\sup_{\gamma \in \mathcal{F}} |R_{\text{q,emp}}[\gamma] - R_{\text{q}}[\gamma]| > \eta\right\} \leq 2\mathcal{N}\left(\tfrac{\epsilon}{8r}, \mathcal{F}, L_\infty(\ell_2^d)\right) e^{-\frac{m(\eta-\epsilon)^2}{2r^2}} \tag{9.35}$$

$$\Pr\left\{\sup_{\gamma \in \mathcal{F}} \left|R_{\text{q}}[\gamma^*_{\text{emp}}] - R_{\text{q}}[\gamma^*]\right| > \eta\right\} \leq 2\left(\mathcal{N}\left(\tfrac{\epsilon}{4r}, \mathcal{F}, L_\infty(\ell_2^d)\right) + 1\right) e^{-\frac{m(\eta-\epsilon)^2}{8r^2}} \tag{9.36}$$

For a proof see [Smola et al., 1999]. The first bound (9.35) is relevant in practice, as one wants to bound the expected quantization error in terms of the empirically measured one. The second bound (9.36) is more useful for theoretical statements — it allows to assess the proximity of an empirically found solution to the optimal solution (provided the empirical quantization functional is minimized).

9.3.2 Bounds on $L_\infty(\ell_2^d)$ covers

Before going into details it is convenient to review briefly what already exists in terms of bounds on the covering number $\mathcal{N}$ for $L_\infty(\ell_2^d)$ metrics. Kegl et al. show:

Lemma 9.10 Kégl, Krzyżak, Linder, and Zeger [1999]
For any $\varepsilon > 0$ there exists a finite collection of polygonal curves $\gamma(\cdot)$ of length L with M nodes in a sphere U_r of radius r in $\mathcal{X}$ such that

$$\sup_{x \in U_r} |\Delta(x, \gamma) - \Delta(x, \gamma')| \leq \varepsilon \tag{9.37}$$

and the number of such curves $\mathcal{N}_\varepsilon$ is bounded as

$$\mathcal{N}_\varepsilon \leq 2^{\frac{2Lr}{\varepsilon}} (S_d)^{M+1} \left(\frac{4r^2\sqrt{d}}{\varepsilon} + \sqrt{d}\right)^d \left(\frac{2rL\sqrt{d}}{M\varepsilon} + 3\sqrt{d}\right)^{Md}. \tag{9.38}$$

Here $\Delta(x, \gamma)$ is defined as the minimum distance between a curve $\gamma(\cdot)$ and $x \in U_r$.

Hence essentially lemma 9.10 means that $\log \mathcal{N}(\varepsilon, \mathcal{F}) = O(\frac{1}{\varepsilon})$. By using tools from chapter 8 one can, depending on the kernel expansion, obtain stronger results, which then, in turn, can replace lemma 9.10 to obtain better bounds on the expected quantization error.

First the distance measure of (9.37) is replaced by an $L_\infty(\ell_2^d)$ norm as defined in (9.34). It follows immediately from the definition that $\|\gamma - \gamma'\|_{L_\infty(\ell_2^d)} \leq \varepsilon$ implies (9.37). Moreover this generates a Banach space of parametrized curves/manifolds mapping $\mathcal{B} \to \mathcal{X}$.

What one has to do in the following is to bound the entropy/covering number of parametrized curves in $L_\infty(\ell_2^d)$ satisfying the constraint $\|P\gamma(\cdot)\|^2 \leq \Lambda$, i.e. the hypothesis class $\mathcal{F}_\Lambda$. Recall the results of section 8.3. As one is dealing with the multi output case, it handy to view $\gamma(\cdot)$ as generated by a linear $d = \dim \mathcal{X}$ dimensional operator in feature space, i.e.

$$\gamma(b) = W\Phi(b) = (\langle w_1, \Phi(b)\rangle, \ldots, \langle w_d, \Phi(b)\rangle) \tag{9.39}$$

with $\|W\|^2 := \sum_{i=1}^d \|w\|^2$. Moreover assume the coordinate system induced by $k(\cdot,\cdot)$. Again, the evaluation operator S (cf. (8.21)) plays a crucial role. One has to adapt it in the following way.

$$\begin{array}{lcl} S_{\Phi(\mathcal{B})} : (\ell_2)^d & \to & L_\infty(\ell_2^d) \\ S_{\Phi(\mathcal{B})} : W & \mapsto & (\langle w_1, \Phi(\mathcal{B})\rangle, \ldots, \langle w_d, \Phi(\mathcal{B})\rangle)\,. \end{array} \tag{9.40}$$

By a technical argument one can see that it is possible to replace $(\ell_2)^d$ by ℓ_2 without further worry — simply reindex the coefficients by

$$\begin{array}{lcl} I_d & : & (\ell_2)^d \to \ell_2 \\ I_d & : & \begin{array}{c} ((w_{11}, w_{12}, \ldots), (w_{21}, w_{22}, \ldots), \ldots, (w_{d1}, w_{d2}, \ldots)) \quad \to \\ (w_{11}, w_{21}, \ldots, w_{d1}, w_{12}, w_{22}, \ldots, w_{d2}, w_{13}, \ldots) \end{array} \end{array} \tag{9.41}$$

By construction $I_d U_{(\ell_2)^d} = U_{\ell_2}$ and vice versa, thus also $\|I_d\| = \|I_d^{-1}\| = 1$. Before proceeding to the actual theorem one has to define a scaling operator A_d for the multi output case in analogy to (8.16). It is the d times tensor product of A, i.e.

$$\begin{array}{lcl} A_d & : & (\ell_2)^d \to (\ell_2)^d \\ A_d & := & \underbrace{A \times A \times \ldots \times A}_{d\text{–times}} \end{array} \tag{9.42}$$

Theorem 9.11 Bounds for Principal Curves Classes
Let k be a Mercer kernel, let Φ be induced via (8.6) and let $T := S_{\Phi(\mathcal{B})}\Lambda$ where $S_{\Phi(\mathcal{B})}$ is given by (9.40) and $\Lambda \in \mathbb{R}^+$. Let A be defined by (8.16) and A_d by (9.42). Then the entropy numbers of T satisfy

$$\epsilon_n(T) \leq \Lambda \epsilon_n(A_d). \tag{9.43}$$

Proof Again one has to use a factorization argument just like in theorems 8.6 and 9.6. In particular one uses the following property.

$$\begin{tikzcd}
U_{\ell_2} \subset \ell_2 \arrow[rr, "T"] \arrow[d, "I_d^{-1}"'] & & L_\infty(\ell_2^d) \\
U_{(\ell_2)^d} \subset (\ell_2)^d \arrow[r, "\Lambda"'] & \Lambda U_{(\ell_2)^d} \subset (\ell_2)^d \arrow[r, "A_d"] \arrow[ur, "S_{\Phi(\mathcal{B})}"] & \Lambda \mathcal{E}_d \subset (\ell_2)^d \arrow[u, "S_{(A^{-1}\Phi(\mathcal{B}))}"']
\end{tikzcd} \tag{9.44}$$

In other words one exploits

$$\epsilon_n\left(S_{\Phi(\mathcal{B})}\left(\Lambda U_{(\ell_2)^d}\right)\right) = \epsilon_n\left(S_{(A^{-1}\Phi(\mathcal{B}))} A_d \Lambda I_d^{-1}\right) \tag{9.45}$$

$$\leq \left\|(A^{-1}\Phi(\mathcal{B}))\right\| \epsilon_n(A_d)\Lambda \left\|I_d^{-1}\right\| \tag{9.46}$$

$$\leq \Lambda \epsilon_n(A_d) \tag{9.47}$$

which proves the theorem. ■

Note that the price to pay for the multi output case (i.e. vector valued functions) is a degeneracy in the eigenvalues of A_d — scaling factors appear d times, instead of only once in the single output situation. From Theorem 8.12 one immediately obtains the following corollary to bound $e_n(A_d)$.

Corollary 9.12 Entropy numbers for $\Phi(\mathcal{B})$ — ***The Multi Output Case*** Let $k: \mathcal{B} \times \mathcal{B} \to \mathbb{R}$ be a Mercer kernel, let A_d be defined by (8.16) and A_d by (9.42). Then there exists an operator A_d such that

$$\epsilon_n(A_d: \ell_2 \to \ell_2) \leq \inf_{(a_s)_s: \left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \in \ell_2} \sup_{j \in \mathbb{N}} 6C_k\sqrt{d} \left\| \left(\frac{\sqrt{\lambda_s}}{a_s}\right)_s \right\|_{\ell_2} n^{-\frac{1}{j \cdot d}} (a_1 a_2 \cdots a_j)^{\frac{1}{j}}. \tag{9.48}$$

The proof exploits the fact that the radius of the sphere corresponding to A_d, as determined by $\|(C_k\sqrt{\lambda_s'}/a_s')_s\|_{\ell_2}$ is $\sqrt{d}$ times the radius of the corresponding sphere for A (due to the ℓ_2^d norm). The $\sup_{j' \in \mathbb{N}/d}$ is replaced by $\sup_{j \in \mathbb{N}}$ and $j' = jd$. Substitution of the variables, also in the geometric norm, yields the desired result.

Note that the dimensionality of $\mathcal{B}$ does not affect these considerations immediately, however it has to be taken into account implicitly by the decay of the eigenvalues of the integral operator induced by k. However d appears twice — once due to the increased operator norm (the $\sqrt{d}$ term) for the scaling operator A_d, and secondly due to the slower decay properties (each scaling factor a_i appears d times).

The same techniques that led to the propositions 8.10, 8.11, and 8.16 can be applied here, too. As that would be mainly technical, only one example of such a bound is given below.

Proposition 9.13 Exponential–Polynomial Decay
Suppose k is a Mercer kernel with $\lambda_j = O(e^{-\alpha j^p})$ for some $\alpha, p > 0$. Then

$$\ln \epsilon_n^{-1}(A_d\colon \ell_2 \to \ell_2) = O(\ln^{\frac{p}{p+1}} n) \tag{9.49}$$

Proof Since $\lambda_j = O(e^{-\alpha j^p})$ there exist some $\beta \in \mathbb{R}+$ such that $\lambda_j \le \beta^2 e^{-\alpha j^p}$. Now assume that the coefficients a_i are described by $(a_j)_j = e^{-\tau/2 j^p}$. As in the proof of theorem 8.11 we bound

$$\sqrt{d}\left\|\left(\frac{\sqrt{\lambda_j}}{a_j}\right)_j\right\|_{\ell_2} = \sqrt{d}\beta\left(\sum_{j=0}^{\infty} e^{(\tau-\alpha)j^p}\right)^{\frac{1}{2}} \begin{cases} \le \sqrt{d}\beta\sqrt{1+\frac{\Gamma(1/p)}{p(\alpha-\tau)^{1/p}}} \\ \ge \sqrt{d}\beta\sqrt{\frac{\Gamma(1/p)}{p(\alpha-\tau)^{1/p}}} \end{cases} \tag{9.50}$$

and

$$(a_1 a_2 \dots a_j)^{\frac{1}{j}} = e^{-\frac{1}{2j}\tau \sum_{s=1}^{j} s^p} \begin{cases} \ge e^{-\frac{\tau}{2(p+1)} j^p} \\ \le e^{-\frac{\tau}{2(p+1)} j^p + \frac{\tau}{2j(p+1)}} \le e^{-\frac{\tau}{2(p+1)} j^p + \frac{\tau}{2(p+1)}} \end{cases} \tag{9.51}$$

For the purpose of finding an upper bound, $\sup_{j \in \mathbb{N}}$ can be replaced by $\sup_{j \in [1,\infty]}$. In particular the latter is obtained for

$$j = \left(\frac{2(p+1)\ln dn}{\tau p}\right)^{\frac{1}{p+1}} \tag{9.52}$$

Resubstitution yields the claimed rate of convergence for any $\tau \in (0, \alpha)$ which proves the theorem. Again, as in theorem 8.11, one could show that the obtained rate is tight. ■

Observe that one may obtain bounds with the same asymptotic rate for ϵ_n as in 8.16[3] for the case of kernels like Gaussian radial basis functions.

9.3.3 Rates of Convergence

It is of theoretical interest how well Principal Manifolds can be learned. The $O(m^{-1/3})$ result of Kégl et al. [1999] is improved and it is shown that, depending on the kernel (i.e. regularization operator used), the learning rate can be bounded by $O(m^{-\frac{\alpha}{2(\alpha+1)}})$ for polynomial rates of decay ($\alpha + 1$ is the rate of decay), or $O(m^{-1/2+\alpha})$ for exponential rates of decay (α is an arbitrary positive constant). The latter is nearly optimal, as supervised learning rates are bounded by $O(m^{-1/2})$.

Proposition 9.14 Learning Rates for Principal Manifolds

For any fixed $\mathcal{F}_\Lambda$ the learning rate of principal manifolds can be lower bounded by $O(m^{-1/2+\alpha})$ where α is an arbitrary positive constant, i.e.

$$R_{\mathrm{q}}[\gamma^*_{\mathrm{emp}}] - R_{\mathrm{q}}[\gamma^*] \le O(m^{-1/2+\alpha}) \text{ for } \gamma^*_{\mathrm{emp}}, \gamma^* \in \mathcal{F}_\Lambda \tag{9.53}$$

3. Note that in proposition 8.16 d is the dimension of the input space, i.e. one has to use $\dim \mathcal{B}$ instead.

if the eigenvalues of k decay exponentially. Moreover the learning rate can be lower bounded by $O(m^{\frac{\alpha}{2(\alpha+1)}})$ in the case of polynomially decaying eigenvalues,

$$R_{\mathrm{q}}[\gamma^*_{\mathrm{emp}}] - R_{\mathrm{q}}[\gamma^*] \leq O(m^{\frac{\alpha}{2(\alpha+1)}}) \text{ for } \gamma^*_{\mathrm{emp}}, \gamma^* \in \mathcal{F}_\Lambda \tag{9.54}$$

where $\alpha + 1$ is the rate of decay.

Proof The proof exploits a clever trick from [Kégl et al., 1999], however without the difficulty of also having to bound the approximation error. Proposition 9.9 will be useful.

$$R_{\mathrm{q}}[\gamma^*_{\mathrm{emp}}] - R_{\mathrm{q}}[\gamma^*] = \int_0^\infty Pr\left\{R_{\mathrm{q}}[\gamma^*_{\mathrm{emp}}] - R_{\mathrm{q}}[\gamma^*] > \eta\right\} d\eta \tag{9.55}$$

$$\leq u + \epsilon + 2(\mathcal{N}(\epsilon/4r) + 1)\int_{u+\varepsilon}^\infty e^{-\frac{m(\eta-\epsilon)^2}{8r^2}} d\eta \tag{9.56}$$

$$\leq u + \epsilon + \frac{8r^2}{um}(\mathcal{N}(\epsilon/4r) + 1)e^{-\frac{mu^2}{8r^2}} \tag{9.57}$$

$$\leq \sqrt{\frac{8r^2 \ln(\mathcal{N}(\epsilon/4r) + 1)}{m}} + \epsilon + \sqrt{\frac{8r^2}{m \ln(\mathcal{N}(\epsilon/4r) + 1)}} \tag{9.58}$$

Here (9.57) was obtained by bounding $\int_x^\infty \exp(-t^2/2)dt$ by $\exp(-x^2/2)/x$, and (9.58) by substituting $u^2 = \frac{8r^2}{m}\log(\mathcal{N}(\varepsilon/4r)+1)$. Finally set $\epsilon = \sqrt{1/m}$ and exploit proposition 9.13 to obtain

$$R_{\mathrm{q}}[\gamma^*_{\mathrm{emp}}] - R_{\mathrm{q}}[\gamma^*] = O\left(\sqrt{\ln^{\frac{p+1}{p}} m/m}\right) + O(m^{-\frac{1}{2}}). \tag{9.59}$$

As $\ln^{\frac{p+1}{p}} m$ can be bounded by any $c_\alpha m^\alpha$ for suitably large c_α and $\alpha > 0$ one obtains the desired result. For polynomially decaying eigenvalues one obtains from proposition 8.10 that for a sufficiently large constant c

$$\ln \mathcal{N}\left(\tfrac{\epsilon}{\Lambda}, \mathcal{F}_\Lambda, L_\infty(\ell_2^d)\right) \leq c\varepsilon^{-\frac{2}{\alpha}}. \tag{9.60}$$

Substituting (9.60) into (9.58) yields

$$R_{\mathrm{q}}[\gamma^*_{\mathrm{emp}}] - R_{\mathrm{q}}[\gamma^*] \leq \sqrt{\frac{2^{3-\frac{4}{\alpha}} r^{2-\frac{2}{\alpha}} c}{m}} \varepsilon^{-\frac{1}{\alpha}} + 2\epsilon + O(m^{-\frac{1}{2}}). \tag{9.61}$$

The minimum is obtained for

$$\epsilon = c' m^{-\frac{\alpha}{2(\alpha+1)}} \tag{9.62}$$

for some $c' > 0$. Hence $m^{-\frac{1}{2}}\epsilon^{-\frac{1}{\alpha}}$ is of order $O(m^{-\frac{\alpha}{2(\alpha+1)}})$, which proves the theorem. ■

Interestingly the above result is slightly weaker than the result in Kégl et al. [1999] for the case of length constraints, as the latter corresponds to the differentiation operator, thus polynomial eigenvalue decay of order 2, i.e. $\alpha = 1$ and therefore to a rate $\frac{\alpha}{2(\alpha+1)} = \frac{1}{4}$ (Kégl et al. Kégl et al. [1999] obtain $\frac{1}{3}$). It is unclear, whether this is due to a (possibly) not optimal bound on the entropy numbers induced by k, or the fact that our results were stated in terms of the (stronger) $L_\infty(\ell_2^d)$ metric. This yet to be fully understood weakness should not detract from the fact that we *can* get better rates by using stronger regularizers, *and* our algorithm can utilize such regularizers.

9.4 Summing Up

It should have become clear by now that the functional analytic tools are a very powerful means of directly bounding the entropy numbers of function classes.

In particular, the results obtained for the case of SV regularization can be directly ported to the setting of regularization networks thus making it possible to provide a "drop in" replacement for uniform convergence bounds in the case of existing algorithms. Moreover, the new viewpoint allows to give good rates of approximation in reproducing kernel Hilbert spaces, significantly improving on previous results (e.g. [Corradi and White, 1995]).

Carrying the idea of a "drop-in" replacement further, one can see that linear programming machines and similar approaches requiring sparsity of the expansion via a suitable ℓ_1 regularizer can readily be analyzed in terms of the functional analytic framework stated in this thesis. Viewing the hypothesis of linear programming machines as a bilinear form in some feature space allowed the "translation" of the results obtained for SV machines into statements about convex combinations of hypotheses. Astonishingly, the resulting hypothesis classes seem to be even better behaved (in terms of their entropy numbers) than the SV counterparts. Whilst the "technical innovation" in this case may not be too important by itself, the results are very useful. They allow, for the first time ever, to control the capacity of convex combinations of kernels in a systematic way.

The last application of the new techniques was devoted to regularized principal manifolds, an example of unsupervised learning of vector valued functions. It was shown, how much easier it is (and more straightforward, too), to use entropy/covering numbers. Without the functional analytic tools one would have had to apply a combinatorial reasoning to bound the VC dimension for vector valued functions — a goal that is not easily achieved.

Still much work remains to be done to derive good bounds capacity bounds for unsupervised learning, an area still lacking some fundamental uniform convergence results of the type given for supervised learning. One of the first steps in this direction was taken by Buhmann [1998]. See his work for a detailed discussion of the issues in this type of problems.

Summing Up

Learning with Kernels, as proposed in this thesis, has been shown to be a flexible and effective way, to tackle problems of statistical learning theory. Whilst the foundations have been laid in the 60's with groundbreaking work on classification, clustering, capacity control, and feature space methods, it took thirty years until sufficient computational resources were available, to exploit these advances in practice. Much of the recent success of methods like Support Vector Machines, Gaussian Processes, Kernel Principal Component Analysis, or similar feature space methods can be explained in this way.

The main contribution of this thesis can be found in two aspects. The part on **Algorithms** shows how the rather closely knit initial Support Vector algorithm, as devised by Boser, Cortes, Guyon, and Vapnik, can be decomposed into its basic building blocks. This allows both a deeper understanding of the separate functional elements, and the construction of new algorithms.

One module is the choice of the cost function. While softmargin or ε–insensitive loss functions may be advantageous for many cases, there exist quite a few situations where these standard settings are not optimal. It has been pointed out, how this restriction can be overcome, thus offering a closer match of what might be desired by the user. This additional liberty leads to the imminent question, how it should be exploited in the most effective way. A principled answer to this problem was found, provided some basic properties of the noise model are known. A slight modification of the algorithm, finally, allows an automatic choice of the parameters, thus making the algorithm asymptotically optimal for a given parametric class of noise models.

The second module in Support Vector machines are kernels and regularization. This thesis builds the connection between the former and the latter, using the theory of reproducing kernel Hilbert spaces and Green's functions. This allows to understand the good performance of SV machines from a regularization theoretic point of view, thus resolving the problem why SVMs seemed to defy the curse of dimensionality, although operating in very high dimensional spaces. The regularization theoretic viewpoint allowed to use methods from interpolation theory to significantly increase the number of admissible kernel functions. Moreover, the problem of having to eliminate a subspace of functions from the kernel expansion led in a straightforward way to semiparametric modelling, thus further increase of the flexibility of SV machines.

One might ask why these two building blocks were not integrated into one big picture. The reason is simply that the combinatorial multitude of different algorithms arising from the combination of all these different extensions would have seriously confused the reader, rather than making the fundamental mechanisms of SVMs clear and thus controllable.

Being concerned about algorithms, one has to provide the basic strategies for solving the optimization problems posed in SV learning. A set of three different methods is provided for this purpose. Interior point algorithms, which provide reliable fast strategies for solving medium sized problems, are the most flexible ones in this regard. For large sample sizes, however, iterative approximation methods have to be devised. A detailed description of working set methods and sequential minimal optimization (SMO) is given, including pseudocode, to allow further proliferation of the proposed techniques. The use of concepts from interior point programming allows a deeper understanding (and control) of these approaches.

Having presented the building blocks, one may attempt to apply the new techniques to new problems like unsupervised learning. It is shown how principal component analysis can be extended in two directions: extraction of reliable features, or construction of reliable descriptions of the underlying density (without modelling the latter itself). For both approaches algorithms are presented. Crucial to this advance was the concept of regularized risk functionals and hence the idea of regularization via kernel methods. Practical applications of the algorithms to real world problems (regression and classification) show the versatility of kernel methods.

Whilst the first part was concerned about how to construct algorithms, the second part of the thesis aims at assessing the quality of these algorithms, i.e. to derive **Bounds** on the latter. Whilst conventional reasoning of statistical learning theory is mainly focused on the VC dimension and its extensions or modifications, this part takes a more radical approach.

The attempt to compute tight bounds on a more basic quantity, the covering numbers of a class of functions, is successful, due to the use of special properties of kernel expansions and tools from the theory of Banach spaces. This gives both a theoretical and an empirical handle on deriving uniform convergence bounds. Several methods are proposed for this purpose, and explicit rates and constants for bounding entropy numbers are provided in the SV case. In particular, the results obtained are significantly better than most previous bounds, in some cases improving the statements from a polynomial to an exponential rate.

The last chapter shows that these new methods are not only restricted to SVMs but can be applied to a wide range of similar learning problems. A first example is given with regularization networks, proving significantly tighter rates of approximation than previous results. Next capacity bounds for linear programming machines are given. There it is shown how much can be gained by using kernel expansions in comparison to general smoothness constraints. Finally, also unsupervised learning can be successfully dealt with, stating nearly optimal rates of convergence.

Summing up, new algorithms have been presented, together with new methods how to control them. A final integration was omitted, as it is the new *techniques* and the possibility of combining them quite freely, that is the main contribution of this thesis and will prove useful in the future.

References

M. A. Aizerman, E. M. Braverman, and L. I. Rozonoér. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

H. Akaike. A new look at the statistical model identification. *IEEE Trans. Automat. Control*, 19(6):716–723, 1974.

N. Alon, S. Ben-David, N. Cesa-Bianchi, and D. Haussler. Scale–sensitive Dimensions, Uniform Convergence, and Learnability. *J. of the ACM*, 44(4):615–631, 1997.

S. Amari, N. Murata, K.-R. Müller, M. Finke, and H. Yang. Aymptotic statistical theory of overtraining and cross-validation. *IEEE Trans. on Neural Networks*, 8 (5):985–996, 1997.

E. Anderson, Z. Bai, C. Bischof, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, S. Ostrouchov, and D. Sorensen. *LAPACK Users' Guide*. SIAM, Philadelphia, second edition, 1995.

M. Anthony. Probabilistic analysis of learning in artificial neural networks: The PAC model and its variants. *Neural Computing Surveys*, 1:1–47, 1997. http://www.icsi.berkeley.edu/~jagota/NCS.

M. Anthony and P. Bartlett. Function learning from interpolation. Technical Report 94–013, NeuroCOLT, 1994. An extended abstract appeared in EuroCOLT'95, Paul Vitanyi, ed., LNAI 904, Springer Verlag, 1995, 211–221.

M. Anthony and P. Bartlett. *A Theory of Learning in Artificial Neural Networks*. Cambridge University Press, 1999.

N. Aronszajn. La théorie générale des noyaux réproduisants et ses applications. *Proc. Cambridge Philos. Soc.*, 39:133–153, 1944.

N. Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 68:337–404, 1950.

R. Ash. *Information Theory*. Interscience Publishers, New York, 1965.

A.R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transaction on Information Theory*, 39(3):930–945, 1993.

P. Bartlett, T. Linder, and G. Lugosi. The minimax distortion redundancy in empirical quantizer design. *IEEE Transactions on Information Theory*, 44(5): 1802–1813, 1998.

P. Bartlett, P. Long, and R. Williamson. Fat–Shattering and the Learnability of Real–Valued Functions. *Journal of Computer and System Sciences*, 52(3):434–452, 1996.

P. 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, Cambridge, MA, 1999. MIT Press.

P.L. Bartlett. The sample complexity of pattern classsification with neural networks: the size of the weights is more important than the size of the network. *IEEE Trans. Information theory*, 44(2):525–536, 1998.

P.L. Bartlett, S.R. Kulkarni, and S.E. Posner. Covering numbers for real-valued function classes. *IEEE Transactions on Information Theory*, 1997. (to appear).

R.E. Bellman. *Adaptive Control Processes.* Princeton University Press, Princeton, NJ, 1961.

K. Bennett. Combining support vector and mathematical programming methods for induction. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods - SV Learning*, pages 307–326, Cambridge, MA, 1999. MIT Press.

P.J. Bickel, C.A.J. Klaassen, Y. Ritov, and J.A. Wellner. *Efficient and adaptive estimation for semiparametric models.* J. Hopkins Press, Baltimore, ML, 1994.

C. M. Bishop. *Neural Networks for Pattern Recognition.* Clarendon Press, Oxford, 1995.

C.M. Bishop, M. Svensén, and C.K.I. Williams. GTM: The generative topographic mapping. *Neural Computation*, 10(1):215–234, 1998.

S. Bochner. *Lectures on Fourier integral.* Princeton Univ. Press, Princeton, New Jersey, 1959.

B. E. Boser, I. M. Guyon, and V. N. Vapnik. A training algorithm for optimal margin classifiers. In D. Haussler, editor, *5th Annual ACM Workshop on COLT*, pages 144–152, Pittsburgh, PA, 1992. ACM Press.

P. Bradley and O. Mangasarian. Massive data discrimination via linear suppport vector machines. Mathematical Programming Technical Report 98-05, University of Wisconsin Madison, 1998.

P. S. Bradley, U. M. Fayyad, and O. L. Mangasarian. Data mining: Overview and optimization opportunities. Technical Report 98-01, University of Wisconsin, Computer Sciences Department, Madison, 1998. *INFORMS Journal on Computing*, submitted.

J.M. Buhmann. Empirical risk approximation: An induction principle for unsupervised learning. Technical Report IAI-TR-98-3, Institut fr Informatik III, Universitt Bonn, 1998.

J. R. Bunch and L. Kaufman. Some stable methods for calculating inertia and solving symmetric linear systems. *Mathematics of Computation*, 31:163–179, 1977.

C. J. C. Burges. Simplified support vector decision rules. In *Proc. 13th International Conference on Machine Learning*, pages 71–77, San Mateo, CA, 1996. Morgan Kaufmann.

C. J. C. Burges and B. Schölkopf. Improving the accuracy and speed of support vector learning machines. In M. Mozer, M. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 375–381, Cambridge, MA, 1997. MIT Press.

C.J.C. Burges. The geometry of support vector machines. *Neural Computation*, 1998a. submitted.

C.J.C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Knowledge Discovery and Data Mining*, 1998b. in press.

B. Carl. Inequalities of Bernstein-Jackson-type and the degree of compactness of operators in Banach spaces. *Annales de l'Institut Fourier*, 35(3):79–118, 1985.

B. Carl, I. Kyrezi, and A. Pajor. Metric entropy of convex hulls in Banach spaces. preprint, 1997.

B. Carl and I. Stephani. *Entropy, compactness, and the approximation of operators*. Cambridge University Press, Cambridge, UK, 1990.

S. Chen. *Basis Pursuit*. PhD thesis, Department of Statistics, Stanford University, 1995.

S. Chen, D. Donoho, and M. Saunders. Atomic decomposition by basis pursuit. Technical Report 479, Department of Statistics, Stanford University, 1995.

H. Chernoff. A measure of asymptotic efficiency of tests of a hypothesis based on the sum of observations. *Annals of Mathematical Statistics*, 23:493–507, 1952.

D. Cohn and G. Tesauro. Can neural networks do better than the VC bounds. In R. Lippmann, J. Moody, and D. Touretzky, editors, *Advances in Neural information processings systems 3*, pages 911–917, San Mateo, CA, 1991. Morgan Kaufmann Publishers.

V. Corradi and H. White. Regularized neural networks: Some convergence rate results. *Neural Computation*, 7:1225–1244, 1995.

C. Cortes. *Prediction of Generalization Ability in Learning Machines*. PhD thesis, Department of Computer Science, University of Rochester, 1995.

C. Cortes and V. Vapnik. Support vector networks. *M. Learning*, 20:273 – 297, 1995.

D. Cox and F. O'Sullivan. Asymptotic analysis of penalized likelihood and related estimators. *The Annals of Statistics*, 18:1676–1695, 1990.

N. Cristianini, C. Campbell, and J. Shawe-Taylor. Dynamically adapting kernels in support vector machines. Technical Report NC-TR-98-017, Royal Holloway College, University of London, UK, 1998.

A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *J. Roy. Stat. Soc. B*, 39:1–38, 1977.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition.* Number 31 in Applications of mathematics. Springer, New York, 1996.

K. I. Diamantaras and S. Y. Kung. *Principal Component Neural Networks.* Wiley, New York, 1996.

J.J. Dongarra, J. Du Croz, S. Duff, and S. Hammarling. A set of level 3 basic linear algebra subprograms. *ACM Transactions on Mathematical Software*, 16: 1–17, 1990.

J.J. Dongarra, J. Du Croz, S. Hammarling, and R.J. Hanson. An extended set of fortran basic linear algebra subprograms. *ACM Transactions on Mathematical Software*, 14:18–32, 1988.

R. O. Duda and P. E. Hart. *Pattern Classification and Scene Analysis.* John Wiley and Sons, New York, 1974.

N. Dyn. Interpolation and approximation by radial and related functions. In C.K. Chui, L.L. Schumaker, and D.J. Ward, editors, *Approximation Theory, VI*, pages 211–234. Academic Press, New York, 1991.

A. El-Bakry, R. Tapia, R. Tsuchiya, and Y. Zhang. On the formulation and theory of the Newton interior–point method for nonlinear programming. *J. Optimization Theory and Applications*, 89:507–541, 1996.

M. Ferraro and T. M. Caelli. Lie transformation groups, integral transforms, and invariant pattern recognition. *Spatial Vision*, 8:33 – 44, 1994.

R. Fletcher. *Practical Methods of Optimization.* John Wiley & Sons, New York, 1989.

J. H. Friedman. Exploratory projection pursuit. *Journal of the American Statistical Association*, 82:249–266, 1987.

J. H. Friedman and W. Stuetzle. Projection pursuit regression. *J. American Statistical Association*, 76(376):817–823, 1981.

J.H. Friedman and J.W. Tukey. A projection pursuit algorithm for exploratory data analysis. *IEEE Transactions on Computers*, C-23(9):881–890, 1974.

T.T. Frieß and R.F. Harrison. Linear programming support vector machiens for pattern classification and regression estimation and the set reduction algorithm. TR RR-706, University of Sheffield, Sheffield, UK, 1998.

F. Girosi. Approximation error bounds that use VC-bounds. In *Proceedings of the International Conference on Neural Networks*, pages 295–302, Paris, 1995.

F. Girosi. An equivalence between sparse approximation and support vector machines. *Neural Computation*, 10(6):1455–1480, 1998.

F. Girosi, M. Jones, and T. Poggio. Priors, stabilizers and basis functions: From regularization to radial, tensor and additive splines. A.I. Memo No. 1430, MIT, 1993.

H. Goldstein. *Classical Mechanics.* Addison-Wesley, Reading, MA, 1986.

Y. Gordon, H. König, and C. Schütt. Geometric and probabilistic estimates for entropy and approximation numbers of operators. *Journal of Approximation Theory*, 49:219–239, 1987.

I.S. Gradshteyn and I.M. Ryzhik. *Table of integrals, series, and products.* Academic Press, New York, 1981.

E. Grosswald. *Representation of Integers as Sums of Squares.* Springer, N. Y., 1985.

Y.G. Guo, P.L. Bartlett, J. Shawe-Taylor, and R.C. Williamson. Covering numbers for support vector machines. Submitted to COLT99, 1999.

L. Gurvits. A note on a scale-sensitive dimension of linear bounded functionals in Banach spaces. In M. Li and A. Maruoka, editors, *Algorithmic Learning Theory ALT-97*, LNAI-1316, pages 352–363, Berlin, 1997. Springer.

I. Guyon, B. Boser, and V. Vapnik. Automatic capacity tuning of very large VC-dimension classifiers. In Stephen José Hanson, Jack D. Cowan, and C. Lee Giles, editors, *Advances in Neural Information Processing Systems*, volume 5, pages 147–155. Morgan Kaufmann, San Mateo, CA, 1993.

I. Guyon, J. Makhoul, R. Schwartz, and V. Vapnik. What size test set gives good error rate estimates. *IEEE Pattern Analysis and Machine Intelligence*, 20(1): 52–64, 1998.

M. Hamermesh. *Group theory and its applications to physical problems.* Addison Wesley, Reading, MA, 2 edition, 1962. Reprint by Dover, New York, NY.

F. R. Hampel, E. Ronchetti, P. J. Rousseeuw, and W. Stahel. *Robust Statistics - The approach Based on Influence Function.* Wiley, New York, 1986.

P.C. Hansen. Analysis of discrete ill–posed problems by means of the l–curve. *SIAM Review*, 34(4):561–580, 1992.

W. Härdle. *Smoothing Techniques, With Implementations in S.* Springer, New York, 1991.

T. Hastie and W. Stuetzle. Principal curves. *Journal of the American Statistical Association*, 84(406):502–516, 1989.

G. E. Hinton. Connectionist learning procedures. *Artificial Intelligence*, 40 (1): 143 – 150, 1989.

W. Hoeffding. Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58:13–30, 1963.

A. Hole. Vapnik-chervonenkis generalization bounds for real valued neural networks. *Neural Computation*, 8:6:1277–1299, 1996.

H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441 and 498–520, 1933.

P. J. Huber. Robust statistics: a review. *Ann. Statist.*, 43:1041, 1972.

P. J. Huber. *Robust Statistics.* John Wiley and Sons, New York, 1981.

P. J. Huber. Projection pursuit. *The Annals of Statistics*, 13(2):435–475, 1985.

C. Ji and D. Psaltis. The VC dimension versus the statistical capacity of multilayer networks. In S. J. Hanson J. Moody and R. P. Lippman, editors, *Advances in Neural Information Processing Systems 4*, pages 928–935, San Mateo, CA, 1992. Morgan Kauffmann.

T. Joachims. Making large–scale svm learning practical. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 169–184, Cambridge, MA, 1999. MIT Press.

I. T. Jolliffe. *Principal Component Analysis.* Springer-Verlag, New York, New York, 1986.

L.K. Jones. A simple lemma on greedy approximation in Hilbert space and convergence rates for projection pursuit regression and neural network training. *The Annals of Statistic*, 1990. (to appear).

L.K. Jones. A simple lemma on greedy approximation in Hilbert space and convergence rates for Projection Pursuit Regression and neural network training. *The Annals of Statistics*, 20(1):608–613, 1992.

M.C. Jones and R. Sibson. What is projection pursuit? *Journal of the Royal Statistical Society, Series A*, 150(1):1–36, 1987.

K. Karhunen. Zur Spektraltheorie stochastischer Prozesse. *Ann. Acad. Sci. Fenn.*, 34, 1946.

M. Karpinski and A. Macintyre. Polynomial bounds for VC dimension of sigmoidal neural networks. Technical Report TR-95-001, International Computer Science Institute, Berkeley, CA, 1995.

W. Karush. Minima of functions of several variables with inequalities as side constraints. Master's thesis, Dept. of Mathematics, Univ. of Chicago, 1939.

L. Kaufmann. Solving the quadratic programming problem arising in support vector classification. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 147–168, Cambridge, MA, 1999. MIT Press.

M. Kearns. A bound on the error of cross validation using the approximation and estimation rates, with consequences for the training-test split. *Neural Computation*, 9(5):1143–1161, 1997.

M. Kearns, Y. Mansour, A.Y. Ng, and D. Ron. An experimental and theoretical comparison of model selection methods. *Machine Learning*, 27:7–50, 1997.

M. J. Kearns, R. E. Schapire, and L. M. Sellie. Toward efficient agnostic learning. *Machine Learning*, 17(2):115–141, 1994.

M.J. Kearns and R.E. Schapire. Efficient distribution–free learning of probabilistic concepts. *Journal of Computer and System Sciences*, 48(3):464–497, 1994.

B. Kégl, A. Krzyżak, T. Linder, and K. Zeger. Learning and design of principal curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1999.

G.S. Kimeldorf and G. Wahba. A correspondence between Bayesan estimation on stochastic processes and smoothing by splines. *Ann. Math. Statist.*, 2:495–502, 1971.

M. Kirby and L. Sirovich. Application of the karhunen-loeve procedure for the characterization of human faces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-12(1):103–108, 1990.

T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

H. König. *Eigenvalue Distribution of Compact Operators*. Birkhäuser, Basel, 1986.

H. W. Kuhn and A. W. Tucker. Nonlinear programming. In *Proc. 2^{nd} Berkeley Symposium on Mathematical Statistics and Probabilistics*, pages 481–492, Berkeley, 1951. University of California Press.

C.L. Lawson, R.J. Hanson, D. Kincaid, and F.T. Krogh. Basic linear algebra subprograms for FORTRAN usage. *ACM Transactions on Mathematical Software*, 5:308–323, 1979.

W.S. Lee. *Agnostic Learning and Single Hidden Layer Neural Networks*. PhD thesis, Australian National University, 1996. http://routh.ee.adfa.oz.au/~weesun/thesis.ps.Z.

W.S. Lee, P.L. Bartlett, and R.C. Williamson. The importance of convexity in learning with squared loss. *IEEE Transactions on Information Theory*, 1998.

W. Maass. Neural nets with superlinear VC-dimension. *Neural Computation*, 6: 877–884, 1994.

David J. C. MacKay. *Bayesian Modelling and Neural Networks*. PhD thesis, Computation and Neural Systems, California Institute of Technology, Pasadena, CA, 1991.

M. C. Mackey and L. Glass. Oscillation and chaos in physiological control systems. *Science*, 197:287–289, 1977.

W.R. Madych and S.A. Nelson. Multivariate interpolation and conditionally positive definite functions. II. *Mathematics of Computation*, 54(189):211–230, 1990.

Y. Makovoz. Random approximants and neural networks. *Journal of Approximation Theory*, 85:98–109, 1996.

O.L. Mangasarian. Linear and nonlinear separation of patterns by linear programming. *Operations Research*, 13:444–452, 1964.

O.L. Mangasarian. *Nonlinear Programming*. McGraw-Hill, New York, NY, 1969.

B. Maurey. In: "Remarques sur un resultat non publiè de B. Maurey" by G. Pisier. In Centre de Mathematique, editor, *Seminarie d'analyse fonctionelle 1980–1981*, Palaiseau, 1981.

G.P. McCormick. *Nonlinear Programming: Theory, Algorithms, and Applications*. Wiley-Interscience, New York, NY, 1983.

R. Meir. Structural risk minimization for nonparametric time series prediction.

In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10. The MIT Press, 1998.

J. Mercer. Functions of positive and negative type and their connection with the theory of integral equations. *Philos. Trans. Roy. Soc. London*, A 209:415–446, 1909.

C.A. Micchelli. Interpolation of scattered data: distance matrices and conditionally positive definite functions. *Constructive Approximation*, 2:11–22, 1986.

M. L. Minsky and S. Papert. *Perceptrons*. MIT Press, Cambridge, MA, 1969.

J. Moody and C. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, *1*(2):281–294, 1989.

Jorge J. More and Gerardo Toraldo. On the solution of large quadratic programming problems with bound counstraints. *SIAM Journal on Optimization*, 1(1):93–113, 1991.

V. A. Morozov. *Methods for Solving Incorrectly Posed Problems*. Springer Verlag, 1984.

C. Müller. *Analysis of Spherical Symmetries in Euclidean Spaces*, volume 129 of *Applied Mathematical Sciences*. Springer, New York, 1997.

K.-R. Müller, M. Finke, K. Schulten, N. Murata, and S. Amari. A numerical study on learning curves in stochastic multi-layer feed-forward networks. *Neural Computation*, 8:1085–1106, 1996.

K.-R. Müller, J. Kohlmorgen, and K. Pawelzik. Analysis of switching dynamics with competing neural networks. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E78–A(10):1306–1315, 1995.

K.-R. Müller, A. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik. Predicting time series with support vector machines. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 243–254, Cambridge, MA, 1999. MIT Press.

K.-R. Müller, A. J. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik. Predicting time series with support vector machines. In *Artificial Neural Networks — ICANN'97*, 1997.

N. Murata, S. Yoshizawa, and S. Amari. Network information criterion—determining the number of hidden units for artificial neural network models. *IEEE Transactions on Neural Networks*, 5:865–872, 1994.

N.J. Nilsson. *Learning machines: Foundations of Trainable Pattern Classifying Systems*. McGraw–Hill, 1965.

M. Okamoto. Some inequalities relating to the partial sum of binomial probabilities. *Annals of the Institue of Statistical Mathematics*, 10:29–35, 1958.

B.A. Olshausen and D.J. Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381:607–609, 1996.

M. Opper. On the annealed vc entropy for margin classifiers: A statistical mechanics study. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 117–126, Cambridge, MA, 1999. MIT Press.

E. Osuna, R. Freund, and F. Girosi. An improved training algorithm for support vector machines. In J. Principe, L. Gile, N. Morgan, and E. Wilson, editors, *Neural Networks for Signal Processing VII — Proceedings of the 1997 IEEE Workshop*, pages 276 – 285, New York, 1997. IEEE.

E. Osuna and F. Girosi. Reducing the run-time complexity in support vector regression. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 271–284, Cambridge, MA, 1999. MIT Press.

K. Pawelzik, J. Kohlmorgen, and K.-R. Müller. Annealed competition of experts for a segmentation and classification of switching dynamics. *Neural Computation*, 8 (2):342–358, 1996.

K. Pearson. On lines and planes of closest fit to points in space. *Philosophical Magazine*, 2 (sixth series):559–572, 1901.

J. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 185–208, Cambridge, MA, 1999. MIT Press.

T. Poggio. On optimal nonlinear associative recall. *Biological Cybernetics*, 19: 201–209, 1975.

D. Pollard. *Convergence of stochastic processes.* Springer-Verlag, Berlin, 1984.

W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing (2nd ed.).* Cambridge University Press, Cambridge, 1992. ISBN 0-521-43108-5.

R.T. Prosser. The ε–Entropy and ε–Capacity of Certain Time–Varying Channels. *Journal of Mathematical Analysis and Applications*, 16:553–573, 1966.

C. R. Rao. *Linear Statistical Inference and its Applications.* Wiley, New York, 1973.

G. Rätsch. Ensemble learning for classification. Master's thesis, University of Potsdam, 1998. in German.

F. Riesz and B.S. Nagy. *Functional Analysis.* Frederick Ungar Publishing Co., 1955.

B.D. Ripley. Neural networks and related methods for classification. *Proc. Royal Soc. London*, 1994.

J. Rissanen. Minimum-description-length principle. *Ann. Statist.*, 6:461–464, 1985.

S. Roweis and Z. Ghahramani. A unifying review of linear gaussian models. *Neural Computation*, 8, 1998. to appear.

D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(9):533–536, 1986.

S. Saitoh. *Theory of Reproducing Kernels and its Applications.* Longman Scientific & Technical, Harlow, England, 1988.

C. Saunders, M.O. Stitson, J. Weston, L. Bottou, B. Schölkopf, and A. Smola. Support vector machine — reference manual. Technical Report CSD-TR-98-03, Department of Computer Science, Royal Holloway, University of London, Egham, UK, 1998.

R. E. Schapire, Y. Freund, P. Bartlett, and W. S. Lee. Boosting the margin: a new explanation for the effectiveness of voting methods. In *Proc. 14th International Conference on Machine Learning*, pages 322–330. Morgan Kaufmann, 1997.

M. Schmidt and H. Gish. Speaker identification via support vector classifiers. In *Proc. ICASSP '96*, pages 105–108, Atlanta, GA, 1996.

I.J. Schoenberg. Metric spaces and completely monotone functions. *Ann. of Math.*, 39:811–841, 1938a.

I.J. Schoenberg. Metric spaces and positive definite functions. *Trans. Amer. Math. Soc.*, 44:522–536, 1938b.

B. Schölkopf. *Support Vector Learning.* R. Oldenbourg Verlag, Munich, 1997.

B. Schölkopf, P. Bartlett, A. Smola, and R. Williamson. Support vector regression with automatic accuracy control. In L. Niklasson, M. Bodén, and T. Ziemke, editors, *Proceedings of ICANN'98*, Perspectives in Neural Computing, pages 111 – 116, Berlin, 1998a. Springer Verlag.

B. Schölkopf, C. Burges, and V. Vapnik. Extracting support data for a given task. In U. M. Fayyad and R. Uthurusamy, editors, *Proceedings, First International Conference on Knowledge Discovery & Data Mining*, Menlo Park, 1995. AAAI Press.

B. Schölkopf, C. Burges, and V. Vapnik. Incorporating invariances in support vector learning machines. In C. von der Malsburg, W. von Seelen, J. C. Vorbrüggen, and B. Sendhoff, editors, *Artificial Neural Networks — ICANN'96*, pages 47 – 52, Berlin, 1996. Springer Lecture Notes in Computer Science, Vol. 1112.

B. Schölkopf, P. Knirsch, A. Smola, and C. Burges. Fast approximation of support vector kernel expansions, and an interpretation of clustering as approximation in feature spaces. In P. Levi, M. Schanz, R.-J. Ahlers, and F. May, editors, *Mustererkennung 1998 — 20. DAGM-Symposium*, Informatik aktuell, pages 124 – 132, Berlin, 1998b. Springer.

B. Schölkopf, S. Mika, A. Smola, G. Rätsch, and K.-R. Müller. Kernel PCA pattern reconstruction *via* approximate pre-images. In L. Niklasson, M. Bodén, and T. Ziemke, editors, *Proceedings of the 8th International Conference on Artificial Neural Networks*, Perspectives in Neural Computing, pages 147 – 152, Berlin, 1998c. Springer Verlag.

B. Schölkopf, J. Shawe-Taylor, A.J. Smola, and R.C. Williamson. Generalization bounds via eigenvalues of the gram matrix. Submitted to COLT99, 1999a.

B. Schölkopf, P. Y. Simard, A. J. Smola, and V. N. Vapnik. Prior knowledge in support vector kernels. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *Advances in Neural information processings systems*, volume 10, pages 640–646, Cambridge, MA, 1998. MIT Press.

B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299 – 1319, 1998a.

B. Schölkopf, A. Smola, R. Williamson, and P. Bartlett. New support vector algorithms. Technical Report NC-TR-98-027, NeuroColt2, University of London, UK, 1998b. submitted to Neural Computation.

B. Schölkopf, A.J. Smola, P. Bartlett, and R. Williamson. Shrinking the tube — a new support vector regression algorithm. In *Neural Information Processing Systems 1998*, Boston, MA, 1999b. MIT Press. forthcoming.

B. Schölkopf, K. Sung, C. Burges, F. Girosi, P. Niyogi, T. Poggio, and V. Vapnik. Comparing support vector machines with gaussian kernels to radial basis function classifiers. *IEEE Trans. Sign. Processing*, 45:2758 – 2765, 1997.

J. Segman, J. Rubinstein, and Y. Y. Zeevi. The canonical coordinates method for pattern deformation: Theoretical and computational considerations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:1171–1183, 1992.

J. Shawe-Taylor, P. Bartlett, R. Williamson, and M. Anthony. A framework for structural risk minimization. In *Proceedings of the 9th Annual Conference on Computational Learning Theory*, pages 68–76, New York, 1996a. Association for Computing Machinery.

J. Shawe-Taylor, P. Bartlett, R. Williamson, and M. Anthony. A framework for structural risk minimization. Technical Report NC-TR-96-032, Royal Holloway College, University of London, UK, 1996b.

J. Shawe-Taylor, P. Bartlett, R. Williamson, and M. Anthony. Structural risk minimization over data–dependent hierarchies. Technical Report NC-TR-96-053, Royal Holloway College, University of London, UK, 1996c.

J. Shawe-Taylor, P.L. Bartlett, R.C. Williamson, and M. Anthony. Structural risk minimization over data-dependent hierarchies. *IEEE Transactions on Information Theory*, 44(5):1926–1940, 1998.

J. Shawe-Taylor and R.C. Williamson. Generalization performance of classifiers in terms of observed covering numbers. In *Proceedings of EUROCOLT'99*, 1999.

A. Smola, N. Murata, B. Schölkopf, and K.-R. Müller. Asymptotically optimal choice of ε-loss for support vector machines. In L. Niklasson, M. Bodén, and T. Ziemke, editors, *Proceedings of ICANN'98*, Perspectives in Neural Computing, pages 105 – 110, Berlin, 1998a. Springer Verlag.

A. Smola, B. Schölkopf, and K.-R. Müller. General cost functions for support vector regression. In T. Downs, M. Frean, and M. Gallagher, editors, *Proc. of the Ninth Australian Conf. on Neural Networks*, pages 79 – 83, Brisbane, Australia, 1998b. University of Queensland.

A. J. Smola and B. Schölkopf. On a kernel–based method for pattern recognition, regression, approximation and operator inversion. *Algorithmica*, 22:211–231, 1998a.

A.J. Smola, T. Frieß, and B. Schölkopf. Semiparametric support vector and linear programming machines. In *Advances in Neural Information Processing Systems, 11*. MIT Press, 1998c. in press.

A.J. Smola, S. Mika, and B. Schölkopf. Quantization functionals and regularized principal manifolds. Technical Report NC-TR-98-028, Royal Holloway College, University of London, UK, 1998d. submitted to EUROCOLT 99.

A.J. Smola and B. Schölkopf. From regularization operators to support vector kernels. In *Advances in Neural information processings systems 10*, pages 343–349, San Mateo, CA, 1998b.

A.J. Smola, B. Schölkopf, and K.-R. Müller. The connection between regularization operators and support vector kernels. *Neural Networks*, 11:637–649, 1998e.

A.J. Smola, R.C. Williamson, S. Mika, and B. Schölkopf. Regularized principal manifolds. In *Proceedings of EUROCOLT'99*, 1999.

A.J. Smola, R.C. Williamson, and B. Schölkopf. Generalization bounds for convex combinations of kernel functions. Technical Report NC-TR-98-022, Royal Holloway College, University of London, UK, 1998f.

I. H. Sneddon. *The Use of Integral Transforms*. McGraw–Hill, New York, 1972.

S. A. Solla and E. Levin. Learning in linear neural networks: The validity of the annealed approximation. *Physical Review A*, 46(4):2124–2130, 1992.

M. Stone. Cross-validatory choice and assessment of statistical predictors(with discussion). *J. R. Statist. Soc.*, B36:111–147, 1974.

D. L. Swets and J. J. Weng. Using discriminant eigenfeatures for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):831–836, 1996.

F. Takens. Detecting strange attractors in fluid turbulence. In D. Rand and L.S. Young, editors, *Dynamical Systems and Turbulence*, pages 366–381. Springer-Verlag, Berlin, 1981.

M. Talagrand. The Glivenko–Cantelli problem, ten years later. *Journal of Theoretical Probability*, 9(2):371–384, 1996.

V.M. Tikhomirov. Diameters of sets in function spaces and the theory of best approximations. *Russian Mathematical Surveys*, 15(3):75–111, 1960.

A. N. Tikhonov and V. Y. Arsenin. *Solution of Ill–Posed Problems.* Winston, Washington, DC, 1977.

M. Unser, A. Aldroubi, and M. Eden. Fast B-spline transforms for continuous image representation and interpolation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-13(3):277–285, 1991.

R.J. Vanderbei. LOQO: An interior point code for quadratic programming. TR

SOR-94-15, Statistics and Operations Research, Princeton Univ., NJ, 1994.

R.J. Vanderbei. *Linear Programming: Foundations and Extensions.* Kluwer Academic Publishers, Hingham, MA, 1997.

V. Vapnik. *The Nature of Statistical Learning Theory.* Springer, N.Y., 1995.

V. Vapnik. *Statistical Learning Theory.* Wiley, N.Y., 1998.

V. Vapnik and A. Chervonenkis. *Theory of Pattern Recognition [in Russian].* Nauka, Moscow, 1974. (German Translation: W. Wapnik & A. Tscherwonenkis, *Theorie der Zeichenerkennung*, Akademie–Verlag, Berlin, 1979).

V. Vapnik, S. Golowich, and A. Smola. Support vector method for function approximation, regression estimation, and signal processing. In M. Mozer, M. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 281–287, Cambridge, MA, 1997. MIT Press.

V. Vapnik and A. Lerner. Pattern recognition using generalized portrait method. *Automation and Remote Control*, 24, 1963.

V. N. Vapnik. *Estimation of Dependences Based on Empirical Data.* Springer-Verlag, Berlin, 1982.

V. N. Vapnik and A. Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probab. and its Applications*, 16(2):264–280, 1971.

V.N. Vapnik and A. Ya. Chervonenkis. The necessary and sufficient conditions for the uniform convergence of averages to their expected values. *Teoriya Veroyatnostei i Ee Primeneniya*, 26(3):543–564, 1981.

V.N. Vapnik and A. Ya. Chervonenkis. The necessary and sufficient conditions for consistency in the empirical risk minimization method. *Pattern Recognition and Image Analysis*, 1(3):283–305, 1991.

M. Vidyasagar. *A Theory of Learning and Generalization.* Springer, New York, 1997.

N. Ya. Vilenkin. *Special Functions and the Theory of Group Representations*, volume 22 of *Translations of Mathematical Monographs.* American Mathematical Society Press, Providence, NY, 1968.

G. Wahba. *Splines Models for Observational Data.* Series in Applied Mathematics, Vol. 59, SIAM, Philadelphia, 1990.

G. Wahba. Support vector machines, reproducing kernel hilbert spaces and the randomized GACV. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 69–88, Cambridge, MA, 1999. MIT Press.

G.N. Watson. *A Treatise on the Theory of Bessel Functions.* Cambridge University Press, Cambridge, UK, 2 edition, 1958.

A. S. Weigend and N. A. Gershenfeld. *Time Series Prediction: Forecasting the Future and Understanding the Past.* Santa Fe Institute. Addison-Wesley, 1994.

J. Weston, A Gammerman, M. O. Stitson, V. Vapnik, V.Vovk, and C. Watkins. Density estimation using sv machines. In B. Schölkopf, C.J.C. Burges, and A.J. Smola, editors, *Advances in Kernel Methods — SV Learning*, Cambridge, MA, 1999. MIT Press. forthcoming.

E. Weyer. *System Identification in the Behavioural Framework.* PhD thesis, The Norwegian Institute of Technology, 1992.

E. Weyer, I.M.Y. Mareels, , and R.C. Williamson. Finite sample complexity of linear system model identification. Submitted to IEEE Transactions on Automatic Control, 1995.

E. Weyer, I.M.Y. Mareels, and R.C. Williamson. Sample complexity of least squares identification of FIR and ARX models. In *Proceedings of the 13th IFAC World Congress*, volume J, pages 239–244, 1996.

H. Widom. Asymptotic behaviour of eigenvalues of certain integral operators. *Archive for Rational Mechanics and Analysis*, 17:215–229, 1964.

C.K.I. Williams. Prediction with gaussian processes: From linear regression to linear prediction and beyond. *Learning and Inference in Graphical Models*, 1998.

R.C. Williamson. Some results in statistical learning theory with relevance to nonlinear system identification. In *Nonlinear Control Systems Design Symposium 1998 (NOLCOS98)*, volume 2, pages 443–448. IFAC, Elsevier, 1998.

R.C. Williamson, B. Schölkopf, and A.J. Smola. A Maximum Margin Miscellany. Typescript, 1998a.

R.C. Williamson, A.J. Smola, and B. Schölkopf. Generalization performance of regularization networks and support vector machines via entropy numbers of compact operators. NeuroCOLT NC-TR-98-019, Royal Holloway College, 1998b.

A. Yuille and N. Grzywacz. The motion coherence theory. In *Proceedings of the International Conference on Computer Vision*, pages 344–354, Washington, D.C., 1988. IEEE Computer Society Press.

X. Zhang and J. Hutchinson. Simple architectures on fast machines: practical issues in nonlinear time series prediction. In A. S. Weigend and N. A. Gershenfeld, editors, *Time Series Prediction: Forecasting the Future and Understanding the Past.* Santa Fe Institute, Addison-Wesley, 1994.