Dipartimento di Fisica

Multi-Output Learning with Spectral Filters

by

Luca Baldassarre

DIFI, Università di Genova
Via Dodecaneso 33, 16146 Genova, Italy
http://www.fisica.unige.it/

Dottorato di Ricerca in Fisica
XXII Ciclo
Dipartimento di Fisica
Università degli Studi di Genova

DIFI, Università di Genova
Via Dodecaneso 33
I-16146 Genova, Italy
http://www.fisica.unige.it/

Ph.D. Thesis in Physics

Submitted by Luca Baldassarre
Dipartimento di Fisica
Università degli Studi di Genova
luca.baldassarre@unige.it

Date of submission: February 2010

Title: Multi-Output Learning with Spectral Filters

Advisor: Prof. Alessandro Verri
Dipartimento di Informatica e Scienze dell'Informazione
Università degli Studi di Genova
verri@disi.unige.it

External Reviewer: Dott. Massimiliano Pontil
Department of Computer Science
University College London
m.pontil@cs.ucl.ac.uk

Abstract

Multi-output learning deals with the challenge of estimating several related functions from a set of training data, avoiding overfitting. Kernel methods exploit the properties of Reproducing Kernel Hilbert Spaces that allow to reduce an infinite-dimensional problem into a finite-dimensional one, while encoding some sort of prior information of the data generation process via kernel functions. We study a class of regularized kernel methods for vector valued learning which are based on filtering the spectrum of the kernel matrix. The proposed methods include Tikhonov regularization as a special case, as well as interesting alternatives such as vector valued extensions of L2 boosting. Computational properties are discussed for various examples of kernels for vector valued functions and the benefits of iterative techniques are illustrated. Generalizing previous results for the scalar case, we show a finite sample bound for the excess risk of the obtained estimator which allows to prove consistency both for vector valued regression and multi-category classification. Finally, we present some promising results of the proposed algorithms on synthetic examples and on two applications. The first tackles the problem of estimating the liver iron overload in patients suffering from blood related diseases. The second is a framework for reconstructing an active sensorial modality (grasping actions) from a passive one (images of objects to be grasped). We show how both applications can be naturally modeled as vector valued and multi-category problems that can be efficiently solved with the proposed spectral filters.

to Gaia

He would burn the essay after he had destroyed it by totally correcting it in the exact opposite of what he had started out to say. (Thomas Bernhard, *Correction*)

Table of Contents

Chapter 1

Introduction

This thesis presents a class of regularized kernel methods for estimating vector valued functions from a set of input-output examples, the *training set*. The spectral filters provide a powerful, and yet easy-to-use, framework for learning different kinds of multi-output functions with excellent generalization capabilities. In fact, they provide a stable and generalizing estimator by filtering-out unstable components that could lead to fit the noise in the data. We studied their theoretical properties that guarantee generalization and analyzed their algorithmic implementation showing that they can be much faster than currently used alternatives. Our experimental analysis on synthetic and real data confirms these results, although indicating the problematic issue of choosing a proper kernel function for leveraging the relationships among the outputs.

1.1 Learning Multi-Output Functions

The framework. The problem of estimating a function from a set of examples goes under the name of Supervised Learning and here we consider the approach of Statistical Learning Theory (Vapnik, 2000) that, in few words, focuses upon finding an estimator that *generalizes* (i.e. predicts correctly) as well as possible on new unseen examples, instead of trying to find a probabilistic model of the data generation process.

The Statistical Learning Theory literature abounds with analyses of methods that deal with scalar learning problems and in recent years many extensions have been proposed to allow these methods to be applied to several multi-output contexts (see Section 1.2 at the end of the chapter for a short review). However it is not sufficiently clear how the underlying theory extends to them. Despite some algorithms have been analyzed from a theoretical perspective, especially Regularized Least Squares (De Vito and Caponnetto, 2005; Caponnetto and De Vito, 2007), and spaces of vector valued functions have been introduced and studied (Schwartz, 1964; Carmeli et al., 2006), a general theory of multi-output supervised learning is still at its infancy.

Our goal is to show that a principled way to learn a multi-output function exploiting the relationships between its components exists, is consistent and efficient, and often leads to more accurate predictions. The proposed spectral filters rely on the theory of regularization and on matrix valued kernels.

Why learning? Because quite often an explicit modeling of the process underlying a certain experiment or measurement is unfeasible, due to the intrinsic complexity of the process itself or due to our degree of ignorance on its mechanisms. Most importantly, in many cases, we are more interested in finding a good predictive solution than in explaining the data generating process.

Consider for instance the following problem of face recognition. A video is recorded by a fixed camera that films the entrance of a building. We desire a system that automatically identifies the people allowed to enter the building and opens the door for them. From the video stream the system must identify the images that contain a person and within such images locate the region around the face. Then, from a pixel-based description of the face the system must be able to recognize the person. Instead of determining beforehand which are the facial features (expressed as functions of the pixels intensities) that differentiate a person from another, a supervised learning model will use whatever information is available to accurately identify each person. In order to do so, the face is described via general pixel-based features and the system is trained on a set of examples for which we know the exact associations. The system then *learns* the best classifier from a set of hypothesis that can be tuned to the given examples. A good learning model must be capable of accurately classify new instances of faces even if it was trained on a limited number of examples.

Why multi-output? Because most real-world applications, like the example above, have to deal with multi-output functions. Apart from the estimation of a vector field, we also consider multi-task problems where we have several scalar regression or classification problems (*tasks*) and we suppose the tasks to be highly correlated. Furthermore, multi-class problems, for which the goal is to categorize a given example into one of several classes, can be viewed as a multi-output problem, where we want to leverage the relations among the classes. The standard approach to deal with such situations is to learn each component or task independently and combine their outputs. In this thesis we present a framework and a set of algorithms to learn multi-output functions that leverage the structure of the outputs, in order to increase the accuracy of the estimator or impose on it certain properties.

We now present three examples of multi-output problems to clarify the distinctions between vector field estimation, multi-task learning and multi-class problems.

A straightforward example of vector field estimation is given by the common practical problem in experimental physics of estimating a velocity field from scattered spatial measurements — see Figure 1.1. We may know that the field has no sources or wells, hence its divergence is zero everywhere. We can enforce this property by using an appropriate set of hypothesis and suitable algorithms. Furthermore, we can estimate a general vector field and reconstruct its divergence-free part and its curl-free part separately. These solutions cannot be achieved if one estimates each component of the field independently (Macêdo and Castro, 2008). Expected advantages from the multi-output approach: learn a field with desired properties and be more robust against measurement noise.

An example of multi-task regression problem is given by modeling consumer preferences, for instance on a website like Amazon. We desire to train a model to predict the likelihood that a given consumer will buy a particular book. We could try to estimate it from its previous purchases only. A better approach is to leverage the similarities among buyers and correlate their

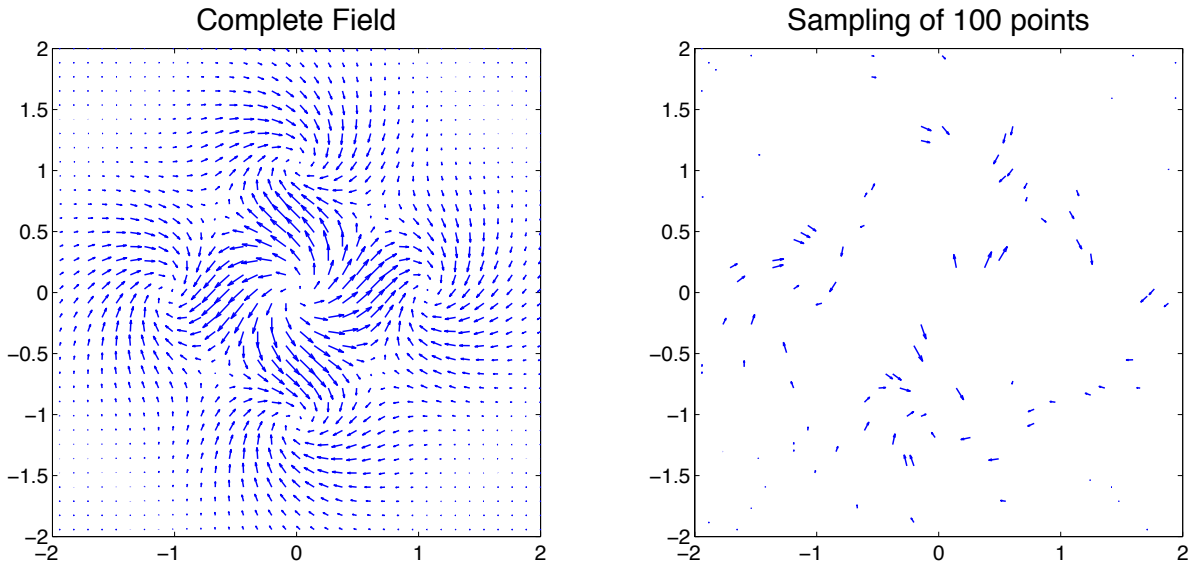


Figure 1.1: Example of a vector field given by the sum of a divergence-free part and a curl-free part. Complete field (left) and a random sampling of 100 examples, that may be used for training a model to estimate the entire field.

purchases. We expect buyers with similar purchasing histories, or tastes, to behave similarly also in relation to new books. We can then model each consumer as a component in a complicated vector valued function whose inputs are features describing the books. The model is then learned combining the buying histories of each consumer, in fact augmenting the training data available for each task and therefore yielding better predictions. Expected advantages from the multi-output approach: pool data across the tasks to augment the training sets available for each task in order to increase prediction accuracy.

An example of a multi-class problem is object recognition in images. The goal is to identify objects in images and classify them. A well-known dataset of images for object classification is the Caltech-256 (Griffin et al., 2007), for which a hand-made hierarchy or taxonomy of objects is provided and could be the key for designing better performing classifiers. In fact, it is reasonable to think that objects within the same category are related and easier to distinguish from objects in a different category. Despite multi-class problems could seemingly be viewed as scalar problems where the outputs are simply the labels (e.g. 1, 2, etc) for each class, this point of view would enforce an unnatural ordering among the classes. It could be the case that examples belonging, say, to class 3 are more similar to the examples in class 1 and in practice it is not feasible to find an appropriate ordering of the labels. In order to avoid this issue, in Chapter 5 we show how we can reformulate a multi-category problem as vector valued regression, assigning vector labels to each class, and then use the tools for enforcing correlation among the components of the vector valued function for exploiting the relations among the classes of objects. Expected advantages from the multi-output approach: be able to learn with fewer training examples and maintain relationships among the classes.

What is regularization theory? Usually, the examples we provide to a learning algorithm are affected by measurement noise or incorrect labeling. Furthermore, the relationship between

input and output could be probabilistic, so that the underlying data generation process does not associate a unique output to a given input. The straightforward approach to learning, also called *Empirical Risk Minimization*, is to tune the parameters describing a function to make it fit the training examples. However, this approach incurs in serious troubles if the candidate function is too "flexible": it will fit the noise in the data and lose the ability to generalize on new examples, a problem commonly known as *overfitting* — see Figure 1.2

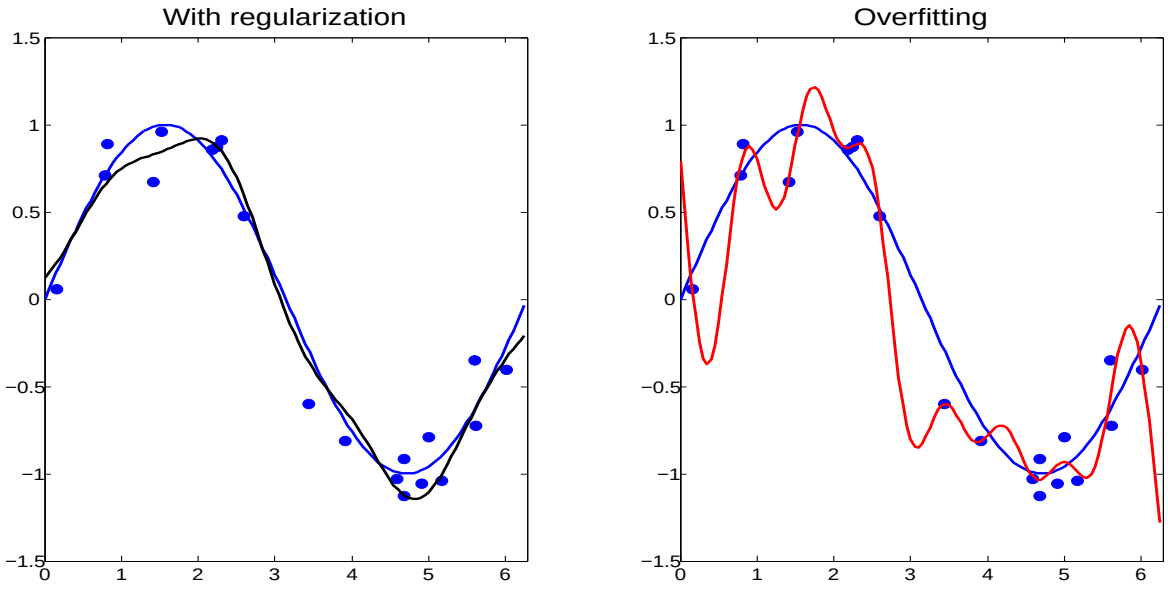


Figure 1.2: A simple example showing the importance of regularization to avoid overfitting. We consider the scalar regression problem of estimating a sine function (in blue) from randomly sampled and noisy points. The regularized solution (right, in black) is less sensitive to noise and better approximates the underlying function. The solution of empirical risk minimization (left, in red) fits the training data and also the noise, resulting in a very oscillatory function that less resembles the true function.

Regularization aims at achieving "noise-proof" estimators by *regularizing* (penalizing) functions that are prone to overfitting, for instance functions that present high-frequency oscillations. Another way of regularizing is to restrict the space of possible estimators, allowing only smooth functions. The two approaches are equivalent and go under the name of *Tikhonov* and *Ivanov* regularization respectively (see for example (Engl et al., 1996)), and both depend on one parameter, the *regularization* parameter, that controls the trade-off between fitting the data and choosing smoother estimates.

Spectral Filters. Instead of introducing a specific regularization term to penalize undesired functions, spectral filters act directly on the solution of the empirical risk minimization to change or eliminate the high-frequency contributions. These methods were firstly introduced in the context of the Theory of Inverse Problems (Groetsch, 1984) and then have been analyzed and applied for scalar learning problems (Caponnetto, 2006; Yao et al., 2007; Bauer et al., 2007; Lo Gerfo et al., 2008).

In this thesis we study from the theoretical, algorithmic and experimental perspectives their

extension to the multi-output setting.

In Chapter 3, we show that for spectral filters regularization is achieved by stabilizing the solution under small perturbations of the training set. In other words, if we slightly perturb the examples, the estimator returned by these algorithms is very similar to the one obtained from the original examples. Interestingly, some of the spectral filters can be implemented as iterative methods that are computationally very efficient, as we show in the experiments of Part II. These methods mimic gradient descent techniques that minimize the empirical risk, but early stopping the iterations ensures regularization is achieved. In other words, the number of iterations plays the role of the regularization parameter: fewer iterations yield smoother functions.

The main theoretical property we investigated is *consistency*.

Consistency. If a method was provided with an infinite amount of training points, we expect it to recover the best possible estimator, that is the estimator one could directly compute if one knew the underlying model that generates the data. This property of a learning algorithm is called *consistency* and it was shown that consistency can be achieved by controlling the complexity of the hypothesis space (Vapnik, 1998) or using regularization techniques (Bauer et al., 2007). We prove consistency for the spectral filters in the vector valued setting in a unified framework for all algorithms and also show a bound on the difference between the generalization error of the estimator obtained from a finite training set and the generalization error of the best possible estimator. This bound decays to zero as the inverse of the square root of the number of examples, ensuring consistency. In the multi-class setting, consistency is in relation to the Bayes classifier, that is the best classifier if one knew the conditional probabilities for each class. We prove Bayes consistency for the spectral filters using the vector valued reformulation of a multi-category problem. In the multi-task case the analysis is complicated by the fact that the training sets for each task might be different and of different cardinality and one could want to assess the risk of each task and not overall. We reserve to future work the investigation of these issues.

What are kernels? These algorithms perform non-parametric regression, that is they do not seek an estimator from a parametric family of functions, but from an infinite-dimensional hypothesis space of functions. The problem is made tractable due to the use of special spaces of functions called Reproducing Kernel Hilbert spaces. These spaces are defined by a function called kernel that takes as argument two input points and returns an operator that acts on the output space.

In the scalar case, it is well known (Kimeldorf and Wahba, 1971; Schölkopf and Smola, 2001) that the solution of Empirical Risk Minimization and its regularized versions can be written as a linear combination of the kernel function evaluated at the training points. A similar result was recently proved for vector valued functions (Micchelli and Pontil, 2005), where they show that the coefficients of the linear combination are elements of the output space and are given by solving a system of equations. Due to these results, an infinite-dimensional problem is reduced to a finite dimensional one that is possible to compute.

Many general-purpose kernels exist for scalar problems and among them the gaussian kernel is perhaps the most widely used since, given enough points, a linear combination of gaussian

kernels allows to well approximate any smooth function. In the case of vector valued functions, the kernels also encode for the relations among the components, but no general off-the-shelf kernel, like the gaussian for scalar problems, exists. We review several matrix valued kernels for vector valued functions that have been proposed in the literature and, in particular, consider a specific class of kernels that can be decomposed as the product of a scalar kernel and a positive semi-definite matrix. We prove a theoretical result that allows to compute this type of matrix valued kernels for several recently proposed regularization terms that directly penalize functions whose components are not related in a specific manner. The importance of this results lies in the fact that the spectral filters act only on the matrix composed of all pair-wise evaluations of the kernel function on the training points. Therefore, the spectral filters can now be used to compute the solution also for the learning problems that use these regularization terms.

1.2 Relevant literature on multi-output learning

Here we provide a generic review of the contributions upon which this thesis is build upon. More specific references are given in each chapter.

Regularized kernel methods (Evgeniou et al., 2000; Schölkopf and Smola, 2001) have been amply studied and the connection between regularization and stability of learning algorithms, introduced in (Bousquet and Elisseeff, 2002), is now widely used to assess consistency of learning algorithms (Mukherjee et al., 2006; Poggio et al., 2004). The connection between Inverse Problems and Learning Theory (De Vito et al., 2005b) has been exploited to extend spectral filters to the scalar learning setting (Yao et al., 2007; Bauer et al., 2007; Lo Gerfo et al., 2008)

Several recent works considered multi-output learning, especially multi-task, and proposed a variety of approaches. Starting from the work of Caruana (1997), related ideas have been developed in the context of regularization methods (Argyriou et al., 2008b; Jacob et al., 2008), Bayesian techniques - e.g. Gaussian processes (Boyle and Frean, 2005; Chai et al., 2009; Alvarez et al., 2009), collaborative filtering (Abernethy et al., 2009) and online sequential learning (Abernethy et al., 2007).

The specific problem of learning a vector valued function has received considerably less attention in machine learning. In statistics we mention the Curds & Whey method of Breiman and Friedman (1997), Reduced Rank Regression (Izenman, 1975), Filtered Canonical y-variate Regression (van der Merwe and Zidek, 1980) and Partial Least Squares (Wold et al., 1984). Interestingly, a literature on statistical techniques for vector field estimation exists in the context of geophysics and goes under the name of kriging (or co-kriging) (Stein, 1999). Few attempts to extend machine learning algorithms from the scalar to the vector setting have also been made. For example some extensions of Support Vector Machines can be found in (Brudnak, 2006) or (Vazquez and Walter, 2003). A study of vector valued learning with kernel methods is started in (Micchelli and Pontil, 2005), where regularized least squares are analyzed from the computational point of view. The error analysis of vector valued Tikhonov regularization is given in (De Vito and Caponnetto, 2005; Caponnetto and De Vito, 2007).

Finally, we note that the use of vector valued kernels for multi-category classification has not been analyzed yet, though we will see that it is implicit in methods such as multi-category Support Vector Machines (Lee et al., 2004). Algorithms for multi-category classification include so

called single machines methods, as well as techniques that reduce the multi-class problems to a family of binary problems, e.g. one versus all and all versus all (see (Tewari and Bartlett, 2005; Rifkin and Klautau, 2004) for discussion and references). In our study we consider the results of Tewari and Bartlett (2005) and Rifkin and Klautau (2004) as starting points for theoretical and practical considerations. The former work shows that naïve extensions of binary classification algorithms to multiple classes might lead to inconsistent methods, and provide sufficient conditions for a multi-class method to be Bayes consistent (see also (Zhang, 2004)). The latter work presents a thorough experimental analysis, supporting the fact that a finely tuned one versus all (OVA) scheme yields performances that are comparable or better than more complicated approaches in most practical situations.

1.3 Contributions

- Thorough understanding of the differences of kernel methods when applied to vector valued, multi-task and multi-class problems.
- Connection between certain regularizers on the components of the multi-output function and a class of matrix valued kernels.
- Extension of spectral filtering methods to the multi-output case.
- Decomposition of the optimization problem for a particular class of kernels, obtaining a great improvement in speed.
- Sound conversion of a multi-category problem into a vector valued one.
- Consistency results for vector valued learning and multi-class learning.
- Experimental applications and results.

1.4 Plan of the thesis

The thesis is dived in two parts. Part I formalizes the problem of vector valued learning, introduces the spectral filters and deals with theoretical aspects.

In Chapter 2, we present and formalize the problem of learning, defining the mathematical quantities required to evaluate the algorithms. We introduce our hypothesis spaces, Reproducing Kernel Hilbert spaces of vector valued functions and discuss the properties of several matrix valued kernels. Here we also present our first theoretical result, that connects the norm of a vector valued function to a weighted sum of the scalar products between its components and how this, in turn, allows to compute the kernels corresponding to several regularizers introduced in the literature. We conclude the chapter reviewing the extension from scalar to vector valued output for Tikhonov regularization.

In Chapter 3, we introduce the spectral algorithms subject of this thesis and discuss how they achieve a regularized solution. We present our main result, the finite bound on the excess risk of the estimators obtained with the spectral filters, and discuss how this bound leads to

consistency. In Section 3.5 we provide a decomposition scheme for a class of kernels that allows to dramatically reduce the computational complexity of the learning problem. We conclude with an analysis of the computational complexity of the proposed algorithms, taking into account the cost of computing the optimal regularization parameter value.

In Chapter 4, we show how the proposed framework for estimating vector valued functions can be applied to multi-task problems and discuss the differences between the two problems.

In Chapter 5, we present and analyze a scheme to represent and solve a multi-class problem as a vector valued regression problem. This allow us to prove Bayes consistency for the proposed methods exploiting the bound derived in Chapter 3.

In Part II, we present some applications of the spectral filters to synthetic data as well as to two real world problems that have been tackled during the course of the doctoral research.

In Chapter 6 ,we present a comparison between vector valued and multi-task learning and show how the latter can benefit from the different training sets available for each task. Successively, we apply the proposed methods to the estimation of a vector field with specific properties that can be encoded in a matrix valued kernel. The vector field is obtained as the sum of a curl-free part and a divergence-free part. We show how, using ad-hoc kernels, it is possible to improve upon regressing each component of the field independently. A computational comparison of the algorithms is also presented, offering a clear indication that iterative spectral filters are much faster than Tikhonov regularization. We end the chapter with a comparison on a real dataset of examination scores of students belonging to more than one hundred schools in the UK. Our methods perform better than the current state-of-the-art algorithms.

Chapter 7 presents and addresses the problem of estimating the iron overload in the liver of patients suffering from Thalassemia or Hereditary Hemochromatosis. We show how this problem can be formulated as vector valued regression and how prior information can be encoded in the design of a specific matrix valued kernel. We report the results and the comparison with a previous parametric model, currently used for diagnosis in parallel with our model.

Chapter 8 deals with the problem of estimating a proper hand configuration for grasping objects. We consider an experimental situation in which a person is grasping several types of objects, each of which can be grasped in different ways. A camera records an image of the object and a sensorized glove measures the angles of the fingers joints. We present three scenarios that differ on the approach to the problem, but they are all based on vector valued regression from visual features, describing the appearance of the object, to the 22 sensors that represent the hand configuration. A preliminary or subsequent multi-category classification is used either to classify the objects or the grasp types. We show that using data recorded from several people it is more difficult to enforce and exploit the correlation between the positions of each hand joint, due to the variability of the grasping action among the volunteers.

Part I

Theory

Chapter 2

Learning and Regularization

In this chapter we formalize the problem of learning a vector valued function from examples, set the notation and define the mathematical objects of interest for the subsequent analysis.

In Section 2.1, we set the notation, formalize the supervised learning problem and discuss some theoretical properties we require from learning algorithms. In Section 2.2, we define matrix valued kernels and the corresponding Reproducing Kernel Hilbert Spaces (RKHS), that will be used as hypothesis spaces of candidate estimators for the learning algorithms. In Section 2.3, we review several matrix valued kernels proposed in the literature and present our first theoretical result, a proposition that connects a particular class of kernels to some regularizers on the components of the vector valued function. This result allows to use spectral algorithms for finding the estimator subject to these regularizers. We conclude the chapter presenting and discussing the extension of Tikhonov regularization from the scalar to the vector valued case, which will help us in understanding the main ideas behind the spectral filters proposed in the next chapter.

2.1 Supervised Learning

Given only a finite number of examples $\mathbf{z} = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, where $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$ for all $i = 1, \ldots, n$, the problem of supervised learning amounts to inferring a function $f_{\mathbf{z}} : \mathcal{X} \to \mathcal{Y}$. We call $\mathbf{z}$ the *training set*, $\mathcal{X}$ the *input space* and $\mathcal{Y}$ the *output space*. Here we will assume that $\mathcal{X} \subseteq \mathbb{R}^p$ and $\mathcal{Y} = \mathbb{R}^d$ and that the data is identically and independently distributed (i.i.d.) according to a fixed, but unknown probability distribution $\rho(x, y) = \rho_{\mathcal{X}}(x)\rho(y|x)$ on $\mathcal{X} \times \mathcal{Y}$. It follows that for a given input point x, more than one output y can be associated to it, due to noise or other factors. The goal of a learning algorithm is to find, using only the data contained in the training set $\mathbf{z}$, the *estimator* $f_{\mathbf{z}}$ that best approximates the probabilistic relationship between $\mathcal{X}$ and $\mathcal{Y}$.

A good estimator, f, should generalize (i.e. predict correctly) on unseen examples and this translates in having a small *expected risk* (or error)

$$\mathrm{I}(f) = \int_{\mathcal{X}\times\mathcal{Y}} ||f(x) - y||_d^2 \rho(x, y) dx dy,$$

where we have chosen the square loss[1] to quantify the discrepancy between the predicted value $f(x)$ and the true value y. The expected risk can be seen as the average error obtained by the candidate solution of the learning problem.

In this framework, the ideal solution is the minimizer of the expected error, called the *regression function*

$$f_\rho(x) = \int_{\mathcal{Y}} y\rho(y|x)dy.$$

However, we cannot directly compute it since the probability distribution ρ is unknown. What we can do is to find a good approximation exploiting all we have, the training data. The (conceptually) simplest approach is the Least Squares (LS) method that consists in computing the minimizer of the *empirical error*

$$\mathrm{I_S}(f) = \frac{1}{n}\sum_{i=1}^{n} ||f(x_i) - y_i||_d^2, \tag{2.1}$$

in a given space $\mathcal{H}$ of functions called *hypothesis space*. This method is more generally called empirical risk minimization induction principle (ERM) when we allow other loss functions.

The choice of a proper hypothesis space is paramount to effectively address the problem of learning. In the next section, we will introduce the Reproducing Kernel Hilbert spaces and we will show why they have become de facto a natural choice in Statistical Learning Theory and its applications. By restricting the range of possible solutions, we are also limiting ourselves from finding the optimal regression function. Indeed, when we work in a particular hypothesis space, the best attainable error is $\inf_{f\in\mathcal{H}} \mathrm{I}(f)$. If the minimizer exists and belongs to $\mathcal{H}$, we denote it with $f_\mathcal{H}$.

We will focus on an important theoretical property of a learning algorithm, called *consistency*. Consistency essentially means that, if more data is available for training, then the algorithm will learn an estimator $f_\mathbf{z}$ that will better approximate the best possible estimator in the hypothesis space, $f_\mathcal{H}$. The quality of the approximation can be evaluated considering the distribution of the *excess risk*, $\mathrm{I}(f_\mathbf{z}) - \mathrm{I}(f_\mathcal{H})$. More precisely, we say that an estimator is *consistent* if, for all positive ε

$$\lim_{n\to\infty} \mathrm{P}\left[\,|\mathrm{I}(f_\mathbf{z}) - \mathrm{I}(f_\mathcal{H})| \geq \varepsilon\right] = 0$$

where $\mathrm{P}\left[\,A\right]$ is the probability of event A according to the probability distribution ρ. A more quantitative result is given by finite sample bounds,

$$\mathrm{P}\left[\,|\mathrm{I}(f_\mathbf{z}) - \mathrm{I}(f_\mathcal{H})| \geq \varepsilon\right] \leq \eta(\varepsilon, n), \quad 0 < \eta \leq 1$$

that quantify the probability that the excess risk is greater than a fixed quantity ε. This probability usually depends on the amount of excess risk that we allow, ε, and the number of examples, n. For an algorithm to be consistent it is necessary that $\lim_{n\to\infty} \eta(\varepsilon, n) = 0$, for any $\varepsilon \geq 0$. These bounds provide insightful rates of convergence and allow the comparison of different algorithms from a pure theoretical standpoint. That is, we prefer algorithms for which $\eta(\varepsilon, n)$ decays faster to zero as n increases.

[1] Other loss functions $V : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ have been introduced in the literature, giving rise to algorithms with different theoretical and practical properties (for reviews and comparisons see (Hastie et al., 2001; Rosasco et al., 2004)).

Since $\mathrm{I}(f_{\mathcal{H}})$ will always be smaller than $\mathrm{I}(f_{\mathbf{z}})$, we can use the probabilistic bound above to write a bound on the expected risk that holds with probability greater than $1-\eta$

$$\mathrm{I}(f_{\mathbf{z}}) \leq \mathrm{I}(f_{\mathcal{H}}) + \varepsilon(\eta, n)$$

where ε depends also on constants defined by properties of the algorithm under consideration. For the algorithm that yields $f_{\mathbf{z}}$ to be consistent, we must have $\lim_{n\to\infty} \varepsilon(\eta, n) = 0$ and $\lim_{n\to\infty} \eta(n) = 0$. We will show that the class of algorithms for multi-output learning that we propose is indeed consistent.

The main issue with ERM is that the minimization problem is *ill-posed*, which means that, according to the definition of Hadamard (Hadamard, 1902), the solution not always exists, it may not be unique and it does not depend continuously on the data. Within the theory of inverse problems, ill-posedness has been tackled via regularization, that is restricting the hypothesis space or favoring solutions with lower complexity. The norm of a function in $\mathcal{H}$ is usually related to its complexity or smoothness. The smaller the norm, the simpler the solution. In fact, one of the more standard approaches, Ivanov regularization (Ivanov, 1976), amounts to the following minimization problem

$$\min_{\substack{f\in\mathcal{H} \\ ||f||_{\mathcal{H}}\leq R}} \mathrm{I}_{\mathrm{S}}(f). \tag{2.2}$$

Through the Lagrangian formulation of the Ivanov problem, we obtain the Tikhonov regularization scheme (Tikhonov and Arsenin, 1977)

$$\min_{f\in\mathcal{H}} \left(\mathrm{I}_{\mathrm{S}}(f) + \lambda||f||_{\mathcal{H}}^2\right). \tag{2.3}$$

The second term is usually called *penalty* or *regularization* term and λ the *regularization parameter*, that must be chosen according to prior information on the difficulty of the learning problem or selected from the available data. For an appropriate choice of the parameters, λ and R, the two problems are equivalent. In his seminal work (Vapnik and Chervonenkis, 1974), Vapnik directly tackled the problem of learning and suggested to compute the minimizer of the empirical risk in a nested structure of hypothesis spaces of increasing complexity and called his approach *Structural Risk Minimization* (SRM). His main argument was a bound on the expected risk that depends on two terms. The first term is the empirical risk, while the second term depends on a measure of the complexity of the hypothesis space and on the number of training examples. If the training examples are scarce, the second term starts to dominate the two, so that simply minimizing the empirical risk does not guarantee to have a small expected risk and hence good generalization. Regularization and SRM are strongly connected. In fact, one can see Ivanov or Tikhonov regularization as defining a nested structure of hypothesis spaces $M = \{f \in \mathcal{H} : ||f||_{\mathcal{H}} \leq R\}$. Regularization can be extended by considering more general penalties $\Omega(f)$, besides $||f||_{\mathcal{H}}$, that favor estimators with different behaviors.

Depending on the hypothesis space we consider, the penalty term can favor different classes of functions. In the next section we will see that working in Reproducing Kernel Hilbert Spaces (RKHS), we can link the regularization term directly to the kernel function, allowing us to tailor the learning process to the type of problem we are dealing with. We will see how eventual prior information on the probability distribution generating the data can guide us in the design of the kernel function.

We conclude noting that the connection between stability of learning algorithms and their generalization abilities has allowed us to introduce learning schemes not directly based on regularized empirical risk minimization. These methods find their motivation in the effort to offer a solution to the learning problem that is stable with respect to small perturbations of the training data. We will see that the spectral filters, topic of this thesis, exactly follow this principle.

2.2 Kernels and RKHS

In this section we recall some basic facts about Reproducing Kernel Hilbert Spaces and discuss the differences between the scalar and the vector valued case. For further details we refer to (Aronszajn, 1950; Schwartz, 1964; Hein and Bousquet, 2004; Micchelli and Pontil, 2005; Carmeli et al., 2006).

We consider a compact input space $\mathcal{X}$ and a finite dimensional Euclidean output space $\mathcal{Y} = \mathbb{R}^d$ with norm $||\cdot||_d$ and scalar product $\langle \cdot, \cdot \rangle_d$.

A *Reproducing Kernel Hilbert Space* (RKHS) is a Hilbert space $\mathcal{H}$ of functions, $f : \mathcal{X} \to \mathbb{R}^d$, such that the evaluation maps $ev_x : f \in \mathcal{H} \to f(x)$ are continuous and bounded. That is, for some $C_x > 0$, we have

$$\|f(x)\|_d = \|ev_x f\|_d \leq C_x \|f\|_{\mathcal{H}}. \tag{2.4}$$

Defining

$$\Gamma(x, s) = ev_x ev_s^*$$

we obtain a kernel $\Gamma : \mathcal{X} \times \mathcal{X} \to \mathcal{B}(\mathcal{Y})$, where $\mathcal{B}(\mathcal{Y})$ is the space of linear and bounded operators on $\mathcal{Y}$, that is the space of $d \times d$ matrices, and ev_x^* is the adjoint[2] of ev_x. Note that $ev_x^* c$ belongs to $\mathcal{H}$ for all $c \in \mathbb{R}^d$.

The kernel Γ has the reproducing property

$$\langle f(x), c \rangle_d = \langle f, ev_x^* c \rangle_\Gamma = \langle f, \Gamma_x c \rangle_\Gamma \,, \tag{2.5}$$

for all $c \in \mathbb{R}^d$ and $x \in \mathcal{X}$, where $ev_x^* c = \Gamma_x c := \Gamma(\cdot, x)c$.

In the special scalar case $\mathcal{Y} = \mathbb{R}$, $\mathcal{B}(\mathbb{R})$ is nothing more than $\mathbb{R}$ itself, so that the kernel, let us call it K in the scalar case, is simply a real valued function of two arguments. In this case ev_x^* and, by definition, the kernel K have the reproducing property, for all $x \in \mathcal{X}$:

$$f(x) = \langle f, ev_x^* \rangle_K = \langle f, K_x \rangle_K \,,$$

where $ev_x^* = K_x := K(\cdot, x)$.

The smallest constant C_x for which (2.4) holds, is $C_x \leq \sup_{x \in \mathcal{X}} \|\Gamma(x, x)\|_{\mathcal{Y},\mathcal{Y}}^{\frac{1}{2}}$, where $\|\cdot\|_{\mathcal{Y},\mathcal{Y}}$ is the operator norm (absolute value in the scalar case). For the rest of our discussion, we assume throughout that

$$\sup_{x \in \mathcal{X}} ||\Gamma(x, x)||_{\mathcal{Y},\mathcal{Y}} = \kappa < \infty. \tag{2.6}$$

[2]Recall that the adjoint of a linear bounded operator A from some Hilbert space $\mathcal{H}$ into itself, is the unique operator A^* such that $\langle A^* f, g \rangle_{\mathcal{H}} = \langle f, Ag \rangle_{\mathcal{H}}$ for all $f, g \in \mathcal{H}$

Conversely, given a kernel Γ, that is a positive semi-definite symmetric function $\Gamma : \mathcal{X} \times \mathcal{X} \to \mathcal{B}(\mathcal{Y})$, a unique RKHS $\mathcal{H}$ (Aronszajn, 1950) can be defined. Positive semi-definiteness means that for any integer $n > 0$ and any sets $\{x_1, \ldots, x_n\} \subseteq \mathcal{X}$, $\{y_1, \ldots, y_n\} \subseteq \mathcal{Y}$

$$\sum_{i,j=1}^{n} \langle y_i, \Gamma(x_i, x_j) y_j \rangle_{\mathcal{Y}} \geq 0. \tag{2.7}$$

The scalar product induced by the kernel Γ on the corresponding RKHS is

$$\langle f, g \rangle_{\Gamma} = \sum_{i=1}^{N} \sum_{j=1}^{M} \langle \Gamma(x_j, x_i) c_i, \beta_j \rangle_{\mathcal{Y}},$$

for any $f, g \in \mathcal{H}$ with $f = \sum_{i=1}^{N} \Gamma(\cdot, x_i) c_i$ and $g = \sum_{j=1}^{M} \Gamma(\cdot, x_j) \beta_j$.

Remark 1. It is interesting to note that, when $\mathcal{Y} = \mathbb{R}^d$, any matrix valued kernel Γ can be seen as a scalar kernel, $Q : (\mathcal{X}, \Pi) \times (\mathcal{X}, \Pi) \to \mathbb{R}$, where Π is the index set of the output components, i.e. $\Pi = \{1, \ldots, d\}$. More precisely, we can write $\Gamma(x, x')_{\ell q} = Q((x, \ell), (x', q))$, for all $l, q \in \Pi$ and $x, x' \in \mathcal{X}$. See (Hein and Bousquet, 2004) for more details.

2.3 Matrix Valued Kernels and Regularizers

Kernels and RKHS are tightly connected. For instance, a scalar valued linear kernel $K(x, x') = x \cdot x'$ defines a space of linear functions from $\mathcal{X}$ to $\mathbb{R}$, while the gaussian kernel $K_\sigma(x, x') = exp(-\frac{||x-x'||^2}{2\sigma^2})$ defines a RKHS which is dense in the space of square-integrable functions,

$$L^2(\mathcal{X}, \rho_{\mathcal{X}}(x)dx) = \left\{ f : \mathcal{X} \to \mathbb{R} \Big| \int_{\mathcal{X}} f^2(x) \rho_{\mathcal{X}}(x) dx < +\infty \right\},$$

with respect to any measure $\rho_{\mathcal{X}}(x)dx$. In other words, we can approximate arbitrarily well any function in $L^2(\mathcal{X}, \rho_{\mathcal{X}}(x)dx)$ with a linear combination of gaussian kernels. Apart from the linear and gaussian kernels, many other scalar kernel functions are known in the literature and their properties, both for theoretical analyses and for practical applications, amply discussed, see for example (Hastie et al., 2001; Schölkopf and Smola, 2001; Hein and Bousquet, 2004).

Unlike the scalar case, in the vector valued case there are no natural off-the-shelf kernels. There are no obvious extensions of gaussian or polynomial kernels and the choice of the kernel is considerably more difficult. In the context of scalar Tikhonov regularization, the penalty term or regularizer is usually a non-decreasing function of the norm of the estimator in the chosen RKHS. The regularizer therefore directly depends on the choice of the kernel and vice-versa. In other words, choosing a regularizer often defines the kernel (Smola et al., 1998). When dealing with multi-output functions, one could write a regularizer based on the norms of each component of the estimator. This approach will result in a diagonal matrix valued kernel, decomposable in as many scalar kernels as the number of output components. More sophisticated approaches design regularizes that couple together the norms of the components and in certain cases, which we will review in the following, these regularizers give rise to proper matrix valued kernels. This is the point of view that has been mainly considered for multi-output functions, especially in the context of multi-task learning. Couplings among the different outputs are explicitly incorporated

in the penalty. In the following, we review several regularizer choices from the perspective of matrix valued kernels. This allows to use algorithms other than Tikhonov regularization, like the spectral filters that manipulate the kernel matrix, which will be the topic of the next chapter. Also, the connection between kernels and regularizers shows a common structure among different regularizers.

Clearly, a matrix valued kernel can also be directly defined without passing through the definition of a regularizer and examples are given at the end of the section.

A straightforward example of matrix valued kernel was proposed in (Micchelli and Pontil, 2005). This kernel imposes a common similarity structure between all the output components and the strength of the similarity is controlled by a parameter ω,

$$\Gamma_\omega(x, x') = K(x, x')(\omega \mathbf{1} + (1 - \omega)\mathbf{I}) = K(x, x') \begin{pmatrix} 1 & \omega & \cdots & \omega \\ \omega & 1 & \cdots & \omega \\ \vdots & \vdots & \ddots & \vdots \\ \omega & \cdots & \omega & 1 \end{pmatrix} \tag{2.8}$$

where $\mathbf{1}$ is the $d \times d$ matrix whose entries are all equal to 1, $\mathbf{I}$ is the d-dimensional identity matrix and K is a scalar kernel on the input space $\mathcal{X}$. Setting $\omega = 0$ corresponds to treating all components independently and the possible similarity among them is not exploited. Conversely, $\omega = 1$ is equivalent to assuming that all components are identical and are described by the same function.

A more general class of matrix valued kernels, which includes the aforementioned kernel as a special case, is composed of kernels of the form:

$$\Gamma(x, x') = K(x, x')A \tag{2.9}$$

where K is a scalar kernel on $\mathcal{X}$ and A a positive semi-definite $d \times d$ matrix that encodes how the outputs are related. This class of kernels allows to decouple the role played by input and output spaces. The choice of the kernel K depends on the desired shape of the function with respect to the input variables, while the choice of the matrix A depends on the relations among the outputs. This information can be available in the form of a prior knowledge on the problem at hand or can be potentially estimated from the training set.

The role of A can be better understood by recalling that any vector valued function belonging to a RKHS can be expressed as $f(x) = \sum_i \Gamma(x, x_i)c_i = \sum_i K(x, x_i)Ac_i$ with $c_i \in \mathbb{R}^d$, so that the ℓ-th component is

$$f^\ell(x) = \sum_i K(x, x_i)\langle e_\ell, Ac_i \rangle = \sum_i \sum_{t=1}^{d} K(x, x_i)A_{\ell t}c_i^t,$$

where $\{e_\ell\}_{\ell=1}^d$ denotes the canonical basis of $\mathbb{R}^d$ and $c_i^t \in \mathbb{R}$ is the t-th component of c_i. Therefore, each f^ℓ belongs to $\mathcal{H}_K$ and it is a different linear combination of the same coefficients $\{c_i\}_{i=1}^n$ which depends on the corresponding row of the matrix A. If A is the d-dimensional identity matrix $\mathbf{I}$, the linear combinations depend on the corresponding components of the coefficients c_i and therefore each f^ℓ is independent to the others.

Also the norm of the vector valued function, f, can be expressed in terms of the coefficients c_i and the matrix A,

$$\begin{aligned} ||f||_\Gamma^2 &= \langle f, f\rangle_\Gamma = \sum_{i,j=1}^{n} \langle c_i, \Gamma(x_i, x_j) c_j\rangle_d \\ &= \sum_{i,j} \langle c_i, K(x_i, x_j) A c_j\rangle_d \\ &= \sum_{i,j} K(x_i, x_j) \langle c_i, A c_j\rangle_d. \end{aligned}$$

Expanding the matrix-vector product Ac_j and the scalar product $\langle\cdot,\cdot\rangle_d$, we obtain

$$||f||_\Gamma^2 = \sum_{i,j} \sum_{\ell,q=1}^{d} K(x_i, x_j) c_i^\ell A_{\ell q} c_j^q. \tag{2.10}$$

For the considered kernels (2.9), the similarity between the components can be evaluated by their pairwise scalar products

$$\begin{aligned} \langle f^\ell, f^q\rangle_K &= \sum_{i,j} K(x_i, x_j) \langle e_\ell, A c_i\rangle_d \langle e_q, A c_j\rangle_d \\ &= \sum_{i,j} \sum_{t,s=1}^{d} K(x_i, x_j) A_{\ell t} c_i^t A_{qs} c_j^s. \end{aligned} \tag{2.11}$$

Given the simple calculations above, we immediately have the following proposition – see (Sheldon, 2008).

Proposition 1. *Let Γ be a product kernel of the form in (2.9). Then the norm of any function, f, in the corresponding RKHS can be written as*

$$||f||_\Gamma^2 = \sum_{\ell,q=1}^{d} A_{\ell q}^\dagger \langle f^\ell, f^q\rangle_K, \tag{2.12}$$

where $A^\dagger$ is called the pseudo-inverse (Penrose, 1955) of A and satisfies $AA^\dagger A = A$.

Proof. We begin the proof by substituting in the right-hand-side of (2.12) the expression (2.11) for the scalar product between two components, obtaining

$$\sum_{\ell,q=1}^{d} A_{\ell q}^\dagger \langle f^\ell, f^q\rangle_K = \sum_{i,j=1}^{n} \sum_{t,s=1}^{d} \sum_{\ell,q=1}^{d} K(x_i, x_j) c_i^t c_j^s A_{\ell t} A_{\ell q}^\dagger A_{qs}.$$

Noting that $\sum_q A_{\ell q}^\dagger A_{qs} = (A^\dagger A)_{\ell s}$ and recalling that $A^\dagger A$ is always symmetric (hence $(A^\dagger A)_{ls} = (A^\dagger A)_{sl}$), we get

$$\sum_{\ell,q=1}^{d} A_{\ell q}^\dagger \langle f^\ell, f^q\rangle_K = \sum_{i,j=1}^{n} \sum_{t,s=1}^{d} \sum_{\ell}^{d} K(x_i, x_j) c_i^t c_j^s A_{\ell t} (A^\dagger A)_{s\ell},$$

From the above expression we can obtain the equation for the norm of the vector valued function, using the fact that $\sum_{\ell=1}^{d} A_{\ell t}(A^\dagger A)_{s\ell} = (AA^\dagger A)_{ts} = A_{ts}$ and remembering (2.10)

$$\sum_{\ell,q=1}^{d} A^\dagger_{\ell q}\langle f^\ell, f^q\rangle_K = \sum_{i,j=1}^{n}\sum_{t,s=1}^{d} K(x_i,x_j)c_i^t c_j^s A_{ts} = ||f||^2_\Gamma$$

□

The above result immediately leads to the interpretation of many regularizers as defining matrix valued kernels. We illustrate this recalling some examples.

In the following, we will consider a scalar kernel K, that defines a scalar RKHS $\mathcal{H}_K$. We will define a functional J on $\mathcal{H}_K^d$ and we will prove that there exists a positive semi-definite matrix A, such that $\mathcal{H}_K^d$ contains the RKHS associated to the matrix valued kernel $\Gamma(\cdot,\cdot) = K(\cdot,\cdot)A$ and $J(f) = ||f||^2_\Gamma$.

Graph regularization. Following (Sheldon, 2008; Micchelli and Pontil, 2004), we define a regularizer that, in addition to a standard regularization on the single components, forces stronger or weaker similarity between them through a $d \times d$ positive weight matrix M.

Given a scalar kernel K, we define a functional on $\mathcal{H}_K^d$, by setting

$$J(f) = \frac{1}{2}\sum_{\ell,q=1}^{d} ||f^\ell - f^q||^2_K M_{\ell q} + \sum_{\ell=1}^{d} ||f^\ell||^2_K M_{\ell\ell}, \tag{2.13}$$

where $f = (f^1,\ldots,f^d)$ and $f^\ell \in \mathcal{H}_K$ for any $\ell = 1,\ldots,d$. Since M is symmetric, the regularizer $J(f)$ can be rewritten as

$$\begin{aligned}\sum_{\ell,q=1}^{d}\left(||f^\ell||^2_K M_{\ell q} - \langle f^\ell, f^q\rangle_K M_{\ell q}\right) + \sum_{\ell=1}^{d}||f^\ell||^2_K M_{\ell\ell} &= \\ \sum_{\ell=1}^{d}||f^\ell||^2_K \sum_{q=1}^{d}(1+\delta_{\ell q})M_{\ell q} - \sum_{\ell,q=1}^{d}\langle f^\ell, f^q\rangle_K M_{\ell q} &= \\ \sum_{\ell,q=1}^{d}\langle f^\ell, f^q\rangle_K L_{\ell q} &\end{aligned} \tag{2.14}$$

where $L = D - M$, with $D_{\ell q} = \delta_{\ell q}\left(\sum_{h=1}^{d} M_{\ell h} + M_{\ell q}\right)$. L is positive semi-definite, in fact, for any vector $v \in \mathbb{R}^d$, we have $v^T L v = v^T(D-M)v = \frac{1}{n}\sum_{ij}(v_i - v_j)^2 M_{ij} + \sum_i v_i^2 M_{ii} \geq 0$, since by definition $M_{ij} \geq 0$. Because L is positive semi-definite, also its pseudo-inverse $L^\dagger$ is positive semi-definite, therefore we can define the matrix valued kernel $\Gamma(x,x') = K(x,x')L^\dagger$ such that $||f||^2_\Gamma = J(f)$ by Proposition 1.

Output components partitioning. The regularizer proposed in (Evgeniou et al., 2005) is based on the idea of grouping the components of the vector valued function into r disjoint clusters and enforcing the components in each cluster to be similar. Following (Jacob et al.,

2008), we consider a partition of the set $\{1, \ldots, d\}$ composed by r disjoint clusters, and define the matrix E as the $d \times r$ matrix, such that $E_{\ell c} = 1$ if the component ℓ belongs to cluster c and 0 otherwise, so that we can have only one 1 in each row. From E we can compute

$$(E^T E)_{cc'} = \sum_s E_{sc} E_{sc'} = \begin{cases} 0 & \text{if } c \neq c' \\ m_c & \text{if } c = c' \end{cases},$$

where m_c is the number of components assigned to the cluster c. Note that $(E^T E)_{cc'} = 0$ when $c \neq c'$, because each component can be assigned to only one cluster.

In particular, $E^T E$ is invertible and we have $(E^T E)^{-1}_{cc'} = \frac{1}{m_c}\delta_{cc'}$. We can also define the $d \times d$ matrix $M = E(E^T E)^{-1}E^T$ such that

$$\begin{aligned} M_{\ell q} &= \sum_{c,c'=1}^{r} E_{\ell c}(E^T E)^{-1}_{cc'} E_{qc'} \\ &= \sum_c E_{\ell c} \frac{1}{m_c} E_{qc} \\ &= \begin{cases} \frac{1}{m_c} & \text{if } \ell \text{ and } q \text{ belong to the same cluster} \\ 0 & \text{otherwise} \end{cases}. \end{aligned}$$

Furthermore, let $I(c)$ be the index set of the components that belong to cluster c. We can then consider the following regularizer that forces the components belonging to the same cluster c to be close to their mean $\overline{f}_c = \frac{1}{m_c}\sum_{q \in I(C)} f^q$,

$$J(f) = \epsilon_1 \sum_{c=1}^{r} \sum_{\ell \in I(c)} ||f^\ell - \overline{f}_c||_K^2 + \epsilon_2 \sum_{c=1}^{r} m_c ||\overline{f}_c||_K^2,$$

where $\epsilon_1, \epsilon_2 > 0$ are two constants that weight the two terms. Using the definition of $\overline{f}_c$ and exploiting the properties of the matrix M, we can rewrite the regularizer as

$$\begin{aligned} J(f) &= \epsilon_1 \sum_{c=1}^{r} \sum_{\ell \in I(c)} \left(||f^\ell||_K^2 - 2\langle f^\ell, \overline{f}_c \rangle + ||\overline{f}_c||_K^2 \right) + \epsilon_2 \sum_{c=1}^{r} m_c ||\overline{f}_c||_K^2 \\ &= \sum_{\ell=1}^{d} \epsilon_1 ||f^\ell||_K^2 + \sum_{c=1}^{r} \sum_{\ell,q \in I(C)} (\epsilon_2 - \epsilon_1) \frac{1}{m_c} \langle f^\ell, f^q \rangle \\ &= \sum_{\ell=1}^{d} \epsilon_1 ||f^\ell||_K^2 + \sum_{\ell,q=1}^{d} (\epsilon_2 - \epsilon_1) \langle f^\ell, f^q \rangle M_{\ell q} \\ &= \sum_{\ell,q=1}^{d} \langle f^\ell, f^q \rangle G_{\ell q} \end{aligned}$$

where $G_{lq} = \epsilon_1 \delta_{lq} + (\epsilon_2 - \epsilon_1) M_{lq}$. Before defining the corresponding matrix valued kernel, we must assess the positive definiteness of $G = \epsilon_1 \mathbf{I} + (\epsilon_2 - \epsilon_1) M$. The first term is simply the identity matrix and so it is positive definite with the eigenvalue 1 with multiplicity d. Rearranging the matrix M, permuting the indexes in order to place the indexes corresponding to the components assigned to the same partition close to each other, we obtain a block diagonal matrix, where

the elements of each block have the value $\frac{1}{m_c}$. It is now easy to see that such a matrix has r eigenvectors composed on m_c elements of value one in the positions corresponding to a partition and zero elsewhere. They all have the same eigenvalue equal to one. The rest of the eigenvectors are orthogonal to these ones and have the eigenvalue equal to zero. Therefore the matrix M is positive semi-definite. It follows that the matrix G is positive semi-definite if $\epsilon_2 > 0$. Therefore, defining $\Gamma(x, x') = K(x, x')G^\dagger$, we obtain a matrix valued kernel such that $||f||^2_\Gamma = J(f)$, by Proposition 1.

Common similarity. The simple matrix valued kernel (2.8), that imposes a common similarity between the output components, can also be viewed as a particular regularizer. In fact, letting $\gamma = \frac{1}{1-\omega+\omega d}$, a simple calculation shows that the corresponding regularizer is

$$J(f) = \gamma \sum_{\ell=1}^{d} ||f^\ell||^2_K + \gamma \frac{\omega d}{1-\omega} \sum_{\ell=1}^{d} ||f^\ell - \frac{1}{d} \sum_{q=1}^{d} f^q||^2_K. \tag{2.15}$$

It is composed of two terms: the first is a standard regularization term on the norm of each component of the estimator; the second forces each f^ℓ to be close to the mean estimator across the components, $\overline{f} = \frac{1}{d}\sum_{q=1}^{d} f^q$. To show that $J(f)$ is the regularizer that corresponds to the matrix valued kernel (2.8), we can rewrite it as

$$\begin{aligned}
J(f) &= \gamma \sum_{\ell,q=1}^{d} \langle f^\ell, f^q \rangle \delta_{\ell q} + \\
&+ \gamma \frac{\omega d}{1-\omega} \sum_{\ell=1}^{d} \left(\langle f^\ell, f^\ell \rangle + \frac{1}{d^2} \sum_{q,t=1}^{d} \langle f^q, f^t \rangle - \frac{2}{d} \sum_{q=1}^{d} \langle f^\ell, f^q \rangle \right) \\
&= \gamma \left(1 + \frac{\omega d}{1-\omega} \right) \sum_{\ell,q=1}^{d} \langle f^\ell, f^q \rangle \delta_{\ell q} - \gamma \frac{\omega}{1-\omega} \sum_{\ell,q=1}^{d} \langle f^\ell, f^q \rangle \\
&= \sum_{\ell,q=1}^{d} \langle f^\ell, f^q \rangle \frac{1}{1-\omega} \left(\delta_{\ell q} - \gamma\omega \right).
\end{aligned}$$

To recover the matrix valued kernel, according to Proposition 1, we must now compute the pseudo-inverse of the matrix $\frac{1}{1-\omega}(\mathbf{I} - \gamma\omega\mathbf{1})$, where $\mathbf{1}$ is the matrix whose elements are all equal to one. We can proceed by using the eigen-decomposition according to the eigen-system of the matrix $\mathbf{1} = U^T D U$, where the diagonal matrix of eigenvalues D has only one nonzero element equal to d. Since $\mathbf{I} = U^T \mathbf{I} U$ for any U such that $U^T U = U U^T = \mathbf{I}$, we have

$$\frac{1}{1-\omega}(\mathbf{I} - \gamma\omega\mathbf{1}) = U^T S U,$$

where $S = \frac{1}{1-\omega} diag(1 - \gamma\omega d, 1, \ldots, 1) = diag(\gamma, \frac{1}{1-\omega}, \ldots, \frac{1}{1-\omega})$. Its pseudo-inverse is computed by inverting the elements of S, obtaining

$$S^\dagger = diag(1 - \omega + \omega d, 1 - \omega, \ldots, 1 - \omega) = (1-\omega)\mathbf{I} + \omega D$$

Putting all together we have

$$\begin{aligned}\left(\frac{1}{1-\omega}(\mathbf{I}-\omega\mathbf{1})\right)^{\dagger} &= U^T S^{\dagger} U \\ &= U^T\big((1-\omega)\mathbf{I}+\omega D\big)U \\ &= (1-\omega)\mathbf{I}+\omega U^T D U \\ &= (1-\omega)\mathbf{I}+\omega\mathbf{1}\end{aligned}$$

which is exactly the matrix that encodes the similarity between the components of the kernel (2.8).

Divergence free and curl free fields. The following two matrix valued kernels apply only for vector fields whose input and output spaces have the same dimensions. In (Macêdo and Castro, 2008), the problem of reconstructing divergence-free or curl-free vector fields is tackled by means of the *Support Vector Regression* method, with ad-hoc matrix valued kernels based on matrix valued radial basis functions. These kernels induce a similarity between the vector field components that depends on the input points, and therefore cannot be reduced to the form $\Gamma(x,x') = K(x,x')A$, with A not depending on x and x'. In fact, these kernels can be written as $\Gamma(x,x') = K(x,x')A(x,x')$, where $K(x,x') = \frac{1}{\sigma^2}exp(-\frac{||x-x'||^2}{2\sigma^2})$ is the scalar gaussian kernel.

The divergence-free kernel is defined as

$$\Gamma_{df}(x,x') = \frac{1}{\sigma^2}e^{-\frac{||x-x'||^2}{2\sigma^2}}A_{x,x'} \tag{2.16}$$

where, letting $v = x - x'$,

$$A_{x,x'} = \frac{1}{\sigma^2}vv^T + \left((d-1) - \frac{||v||^2}{\sigma^2}\right)\mathbf{I},$$

with $\mathbf{I}$ the $d \times d$ identity matrix and vv^T is equivalent to $v \otimes v$, that is a $d \times d$ matrix that contains all pair-wise products between the components of v.

The curl-free kernel is defined as

$$\Gamma_{cf}(x,x') = \frac{1}{\sigma^2}e^{-\frac{||x-x'||^2}{2\sigma^2}}\left(\mathbf{I} - \left(\frac{x-x'}{\sigma}\right)\left(\frac{x-x'}{\sigma}\right)^T\right). \tag{2.17}$$

It is possible to consider a convex linear combination of these two kernels to obtain a kernel for learning any kind of vector field, while at the same time allowing to reconstruct its divergence-free and curl-free parts separately (see (Macêdo and Castro, 2008) and the experiments in Section 6.2 and Section 6.3 for more details).

Product of scalar kernels and operators. Another example of a class of kernels that cannot be decomposed into the simple form $\Gamma(x.x') = K(x,x')A$, is given by kernels defined as $\Gamma(x,x') = \sum_{i=1}^{m} K_i(x,x')B_i$, with $m > 1$ and B_i $d \times d$ positive semi-definite matrices (Micchelli and Pontil, 2005; Caponnetto et al., 2008). Contrary to the case $m = 1$, it is impossible to reduce the kernel Γ to a diagonal one, unless all the matrices B_j can be transformed in diagonal form by the same transformation.

Transformable kernels. In (Caponnetto et al., 2008), examples of several other operator valued kernels (which become matrix valued kernels when $\mathcal{Y} = \mathbb{R}^d$) are introduced. One such example is given by kernels defined by transformations. For the purpose of our discussion, let $\mathcal{Y} = \mathbb{R}^d$, $\mathcal{X}_0 = \mathbb{R}^p$ and T_p be a map (not necessarily linear) from $\mathcal{X}$ to $\mathcal{X}_0$ for $p \in \{1, \dots, d\}$. Then, given a continuous scalar kernel $K : \mathcal{X}_0 \times \mathcal{X}_0 \to \mathbb{R}$, it is possible to define the following matrix valued kernel for any $x, x' \in \mathcal{X}$

$$\Gamma(x, x') = \Big(K(T_p x, T_q x')\Big)_{p,q=1}^{d}.$$

2.4 Tikhonov Regularization from the Scalar to the Vector Case

In this section, we start from the Tikhonov regularization (Tikhonov and Arsenin, 1977; Wahba, 1990; Girosi et al., 1995) in the scalar setting to illustrate the extension to the general vector valued case. In particular we are interested in the role played by the kernel matrix.

Tikhonov regularization, also known as Regularized Least Squares (RLS) (Evgeniou et al., 2000) or Ridge Regression (Hoerl and Kennard, 1970), belongs to the class of spectral filters that we consider in this work. We devote a section to this algorithm because its derivation is straightforward and illustrates the main ingredients of spectral filtering. Furthermore, it allows us to present the extension to the vector valued case of the regularized empirical risk functional in a more detailed manner.

In the scalar case, Tikhonov regularization in a RKHS $\mathcal{H}$, with a scalar kernel K, corresponds to the minimization problem

$$\min_{f \in \mathcal{H}} \left\{ \frac{1}{n} \sum_{i=1}^{n} (y_i - f(x_i))^2 + \lambda \|f\|_K^2 \right\}. \tag{2.18}$$

Its minimizer $f_{\mathbf{z}}^{\lambda}$ in $\mathcal{H}$ is unique due to the strict convexity of the functional and, thanks to the Representer Theorem (Kimeldorf and Wahba, 1971; Girosi et al., 1995), it can be written as

$$f_{\mathbf{z}}^{\lambda}(\cdot) = \sum_{i=1}^{n} K(\cdot, x_i) c_i, \quad c_i \in \mathbb{R} \quad \forall i = 1, \dots, n \tag{2.19}$$

where the coefficients $\mathbf{c} = (c_1, \dots, c_n)$ remain to be determined. Substituting (2.19) in (2.18) and considering the output vector $\mathbf{y} = (y_1, \dots, y_n)$ and the $n \times n$ kernel (or Gram) matrix $\mathbf{K}_{ij} = K(x_i, x_j)$, we can rewrite the functional as

$$\frac{1}{n} ||K\mathbf{c} - \mathbf{y}||_n^2 + \lambda \mathbf{c}^T K \mathbf{c}, \tag{2.20}$$

where $|| \cdot ||_n$ is the Euclidean norm in $\mathbb{R}^n$. The minimizer is found by computing the derivative with respect to $\mathbf{c}$ and setting it equal to zero, obtaining the following condition

$$(\mathbf{K} + \lambda n \mathbf{I})\mathbf{c} = \mathbf{y}. \tag{2.21}$$

on the coefficients $\mathbf{c}$, so that

$$\mathbf{c} = (\mathbf{K} + \lambda n \mathbf{I})^{-1} \mathbf{y} \tag{2.22}$$

In the case of vector valued output, i.e. $\mathcal{Y} = \mathbb{R}^d$, the simplest idea is to consider a naïve extension of Tikhonov regularization, reducing the problem to learning each component independently. Namely, the solution is assumed to belong to

$$\mathcal{H} = \mathcal{H}_1 \oplus \mathcal{H}_2 \cdots \oplus \mathcal{H}_d, \tag{2.23}$$

where the spaces $\mathcal{H}_1, \mathcal{H}_2, \ldots, \mathcal{H}_T$ are endowed with norms $\|\cdot\|_{\mathcal{H}_1}, \ldots, \|\cdot\|_{\mathcal{H}_d}$. It follows that $f = (f^1, \ldots, f^d)$ and $\|f\|_{\mathcal{H}}^2 = \sum_{j=1}^d \|f^j\|_{\mathcal{H}_j}^2$. This naïve vector valued Tikhonov regularization amounts to solving the following problem

$$\min_{f \in \mathcal{H}} \left\{ \frac{1}{n} \sum_{i=1}^n \|y_i - f(x_i)\|_d^2 + \lambda \|f\|_{\mathcal{H}}^2 \right\}, \tag{2.24}$$

that can be rewritten as

$$\min_{f^1 \in \mathcal{H}_1, \ldots, f^d \in \mathcal{H}_d} \left\{ \frac{1}{n} \sum_{i=1}^n \sum_{j=1}^d (y_i^j - f^j(x_i))^2 + \lambda \sum_{j=1}^d \|f^j\|_{\mathcal{H}_j}^2 \right\}.$$

In the expression above we can swap the sum over the components with the minimum, obtaining

$$\sum_{j=1}^d \min_{f^j \in \mathcal{H}_j} \left\{ \frac{1}{n} \sum_{i=1}^n (y_i^j - f^j(x_i))^2 + \lambda \|f^j\|_{\mathcal{H}_j}^2 \right\}.$$

It is now clear that the solution of the problem (2.24) is equivalent to solving T *independent* scalar problems, one for each component of the output.

Within the framework of vector valued kernels, assumption (2.23) corresponds to a special choice of a matrix valued kernel Γ, namely a kernel of the form

$$\Gamma(x, x') = \begin{pmatrix} K_1(x, x') & 0 & \cdots & 0 \\ 0 & K_2(x, x') & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & K_d(x, x') \end{pmatrix}$$

Assuming each component to be independent to the others is a strong assumption and might not reflect the real functional dependence among the data. Recently, a regularization scheme of the form (2.24) has been studied in (Micchelli and Pontil, 2005) for general matrix valued kernels. In this case there is no straightforward decomposition of the problem and one of the main results in (Micchelli and Pontil, 2005) is an extension to the vector valued case of the Representer Theorem, which shows that the regularized solution can be written as

$$f_{\mathbf{z}}^{\lambda}(\cdot) = \sum_{i=1}^n \Gamma(\cdot, x_i) c_i, \quad c_i \in \mathbb{R}^d \quad \forall i = 1, \ldots, n. \tag{2.25}$$

The form of the estimator is essentially the same as in the scalar case, replacing the scalar valued kernel K with the matrix valued one Γ and using vector valued coefficients c_i instead of scalar

ones. The norm of the estimator can be computed as

$$\begin{aligned} ||f_{\mathbf{z}}^{\lambda}||_{\Gamma}^2 &= \langle f_{\mathbf{z}}^{\lambda}, f_{\mathbf{z}}^{\lambda} \rangle \\ &= \langle \sum_{i=1}^{n} \Gamma(\cdot, x_i) c_i, \sum_{i=1}^{n} \Gamma(\cdot, x_i) c_i \rangle \\ &= \sum_{ij} c_i \Gamma(x_i, x_j) c_j^T \\ &= \mathbf{C}^T \mathbf{\Gamma} \mathbf{C} \end{aligned} \tag{2.26}$$

where we have stacked the coefficients c_i in the nd vector $\mathbf{C} = (c_1, \ldots, c_n)$ and the kernel matrix $\mathbf{\Gamma} = \big(\Gamma(x_i, x_j)\big)_{i,j=1,\ldots,n}$ is a $n \times n$ block matrix, where each block $\Gamma(x_i, x_j)$ is a $d \times d$ scalar matrix, so that $\mathbf{\Gamma}$ is a $nd \times nd$ scalar matrix. Indeed the presence of off-diagonal terms reflects the dependence among the components. Similarly we can write the empirical risk as

$$\begin{aligned} \mathrm{I_S}(f_{\mathbf{z}}^{\lambda}) &= \frac{1}{n} \sum_{i=1}^{n} ||y_i - f_{\mathbf{z}}^{\lambda}(x_i)||_d^2 \\ &= \frac{1}{n} \sum_{i=1}^{n} ||y_i - \sum_{j=1}^{n} \Gamma(x_i, x_j) c_j||_d^2 \\ &= \frac{1}{n} ||\mathbf{Y} - \mathbf{\Gamma} \mathbf{C}||_{nd}^2 \end{aligned} \tag{2.27}$$

where $\mathbf{Y}$ is the nd vector where we concatenated the outputs $(y_1, \ldots, y_n)$. Summing the two terms (2.26) and (2.27) and setting their derivative with respect to $\mathbf{C}$ equal to zero, we obtain

$$\mathbf{C} = (\mathbf{\Gamma} + \lambda n \mathbf{I})^{-1} \mathbf{Y}. \tag{2.28}$$

The coefficients of the estimator $f_{\mathbf{z}}^{\lambda}$ are found by inverting the matrix $(\mathbf{\Gamma} + \lambda n \mathbf{I})$, which is invertible since the penalty term $||f||_{\Gamma}^2$ renders all eigenvalues of $\mathbf{\Gamma} + \lambda n \mathbf{I}$ strictly greater than zero and greater or equal than λn. Therefore the penalty term has the effect of stabilizing the inversion of the kernel matrix $\mathbf{\Gamma}$.

Chapter 3

Spectral Filtering

In this chapter, we present the class of regularized kernel methods under study, referring to (Caponnetto, 2006; Yao et al., 2007; Bauer et al., 2007; Lo Gerfo et al., 2008) for the scalar case. In Section 3.1, we present the methods and discuss their motivation recalling Tikhonov regularization, while in Section 3.2 we analyze the derivation of one iterative algorithm, namely the Landweber method. After presenting other examples of spectral filters, in Section 3.3, we discuss our main theorem, a finite sample bound on the excess risk of the estimator, that leads to consistency. The theorem is proved in Section 3.4.
The second main contribution, discussed in Section 3.5, is a decomposition scheme that allows, for a specific class of kernels, to reduce the vector valued learning problem into d scalar learning problems, greatly reducing the computational burden of the algorithms. In Section 3.6, we present bias-aware model selection procedures, in particular *cross-validation* and, finally, in Section 3.7, we discuss more in detail the computational complexity of the methods, taking into consideration the model selection procedure adopted.

3.1 Beyond Tikhonov: Regularization via Spectral Filtering

We call these methods *spectral regularization* because they achieve a stable, hence generalizing, solution by *filtering out* the unstable components of the kernel matrix, that is the directions associated to small eigenvalues. Each algorithm corresponds to a specific filter function and in general there is no natural interpretation in terms of penalized empirical risk minimization, that is we cannot relate the regularization achieved by these algorithms to particular penalty terms on the estimator.

More precisely, the solution of (unpenalized) empirical risk minimization can be written as in (2.25), that is

$$f_{\mathbf{z}}(\cdot) = \sum_{i=1}^{n} \Gamma(\cdot, x_i) c_i, \quad c_i \in \mathbb{R}^d \quad \forall i = 1, \dots, n, \tag{3.1}$$

where the coefficients c_i are given by the solution of

$$\mathbf{\Gamma C} = \mathbf{Y}. \tag{3.2}$$

Comparing the above expression to the one obtained for Tikhonov regularization,

$$(\mathbf{\Gamma} + \lambda n \mathbf{I})\mathbf{C} = \mathbf{Y},$$

we see that adding a penalty to the empirical risk has a stabilizing effect from a numerical point of view, since it suppresses the weights of the components associated to the small eigenvalues of the kernel matrix. This allows to look at Tikhonov regularization as performing a low pass filtering of the kernel matrix, where high frequencies correspond to the small eigenvalues.

The interpretation of regularization as a way to restore stability is classical in ill-posed inverse problems, where many algorithms, besides Tikhonov regularization, are used (Engl et al., 1996). The connection between learning and regularization theory of ill-posed problems (De Vito et al., 2005b) motivates considering spectral regularization techniques. In the scalar case this was done in (Lo Gerfo et al., 2008; Bauer et al., 2007; Caponnetto, 2006).

The idea is that other regularized matrices $g_\lambda(\mathbf{\Gamma})$ besides $(\mathbf{\Gamma} + \lambda n I)^{-1}$ can be defined. Here the matrix valued function $g_\lambda(\mathbf{\Gamma})$ is described by a scalar function g_λ using spectral calculus. More precisely, if

$$\mathbf{\Gamma} = \mathbf{U}\mathbf{S}\mathbf{U}^T$$

is the eigen-decomposition of $\mathbf{\Gamma}$, with $\mathbf{U}^T\mathbf{U} = \mathbf{U}\mathbf{U}^T = \mathbf{I}$ and $\mathbf{S} = diag(\sigma_1, \ldots, \sigma_{nd})$ is the diagonal matrix whose elements are the eigenvalues of $\mathbf{\Gamma}$, then

$$g_\lambda(\mathbf{S}) = diag(g_\lambda(\sigma_1), \ldots, g_\lambda(\sigma_{nd}))$$

and

$$g_\lambda(\mathbf{\Gamma}) = \mathbf{U} g_\lambda(\mathbf{S}) \mathbf{U}^T.$$

For example, in the case of Tikhonov regularization $g_\lambda(\sigma) = \frac{1}{\sigma + n\lambda}$.
Suitable choices of filter functions g_λ define estimators of the form (2.25) with coefficients given by

$$\mathbf{C} = g_\lambda(\mathbf{\Gamma})\mathbf{Y}. \tag{3.3}$$

From the computational perspective, a key point that we show in the following is that many filter functions allow to compute the coefficients $\mathbf{C}$ without explicitly computing the eigen-decomposition of $\mathbf{\Gamma}$.

Remark 2. Note that in the scalar case, manipulations of the kernel matrix have been extensively used to define (and learn) new kernels to be used in Tikhonov regularization - see for example (Smola and Kondor, 2003; Chapelle et al., 2003). In the approach we present, rather than defining a new kernel, each spectral filter g_λ defines an *algorithm* which is not based on empirical risk minimization.

Clearly not all filter functions are admissible. Roughly speaking, an admissible filter function should be such that $g_\lambda(\mathbf{\Gamma})$ approximates $\mathbf{\Gamma}^{-1}$ as λ decreases and its condition number should increase as λ decreases. In the next section we describe several examples and in Section 3.3 we provide a formal definition. The latter will be the key to give an error analysis for the different algorithms within a unified framework.

3.2 Examples of Spectral Regularization Algorithms

We now describe several examples of algorithms that can be cast in the above framework. Although the spectral algorithms have a similar flavor, they present different algorithmic and theoretical properties. This can be seen, for example, comparing the computational complexities of the algorithms, especially if we consider the cost of tuning the regularization parameter. Some considerations along this line are given in Section 3.7, whereas theoretical aspects are discussed in the next section.

3.2.1 L2 Boosting

We start describing in some details vector valued *L2 Boosting*. In the scalar setting, this method has been interpreted as a way to combine weak classifiers corresponding to splines functions at the training set points (Bühlmann and Yu, 2002) and is called *Landweber iteration* in inverse problem literature (Engl et al., 1996). The method can also be seen as the gradient descent minimization of the empirical risk on the whole RKHS, with no further constraint. Regularization is achieved by early stopping of the iterative procedure (Yao et al., 2007), hence the regularization parameter is the number of iterations.

The coefficients (3.3) can be found by setting $\mathbf{C}^0 = 0$ and considering for $i = 1, \dots, t$ the following iteration

$$\mathbf{C}^i = \mathbf{C}^{i-1} + \eta(\mathbf{Y} - \mathbf{\Gamma}\mathbf{C}^{i-1}), \tag{3.4}$$

where the step size η must be chosen so that

$$\|\mathbf{I} - \eta\mathbf{\Gamma}\| < 1. \tag{3.5}$$

Therefore we can set $\eta = 1/\sigma_{max}$, where σ_{max} is the maximum eigenvalue of the kernel matrix $\mathbf{\Gamma}$.

It is easy to see that (3.4) is simply gradient descent if we use (3.1) to write the empirical risk as (ignoring the constant $1/n$)

$$\|\mathbf{\Gamma}\mathbf{C} - \mathbf{Y}\|^2.$$

Its gradient with respect to $\mathbf{C}$ is $\mathbf{\Gamma}\mathbf{C} - \mathbf{Y}$. Gradient descent minimization starts from the null solution $\mathbf{C}^0 = 0$ and at each iteration updates it by an amount proportional to the negative of the gradient, that is exactly (3.4).

The corresponding filter function can be found noting that

$$\begin{aligned}
\mathbf{C}^0 &= 0 \\
\mathbf{C}^1 &= \eta\mathbf{Y} \\
\mathbf{C}^2 &= \eta\mathbf{Y} + \eta(I - \eta\mathbf{\Gamma})\mathbf{Y} \\
\mathbf{C}^3 &= \eta\mathbf{Y} + \eta(I - \eta\mathbf{\Gamma})\mathbf{Y} \\
&\qquad +\eta(\mathbf{Y} - \mathbf{\Gamma}(\eta\mathbf{Y} + \eta(I - \eta\mathbf{\Gamma})\mathbf{Y})) \\
&= \eta\mathbf{Y} + \eta(I - \eta\mathbf{\Gamma})\mathbf{Y} + \eta(I - 2\eta\mathbf{\Gamma} + \eta^2\mathbf{\Gamma}^2)\mathbf{Y} \\
&\dots
\end{aligned}$$

and indeed one can prove by induction (Engl et al., 1996) that the solution at the $t-$th iteration is given by

$$\mathbf{C}^t = \eta \sum_{i=0}^{t-1} (\mathbf{I} - \eta\mathbf{\Gamma})^i \mathbf{Y}.$$

Therefore, the corresponding filter function is $g_t(\sigma) = \eta \sum_{i=0}^{t-1}(1-\eta\sigma)^i$. Interestingly, this filter function has another interpretation that can be seen recalling that, given a matrix A, $\|A\| \in (0,1)$ ($||\cdot||$ is the operator norm)

$$\sum_{i=0}^{\infty} A^i = \frac{1}{\mathbf{I} - A}.$$

If we replace A with $1 - \eta\mathbf{\Gamma}$ and $\|\mathbf{I} - \eta\mathbf{\Gamma}\| < 1$, we get

$$\mathbf{\Gamma}^{-1} = \eta \sum_{i=0}^{\infty} (I - \eta\mathbf{\Gamma})^i.$$

Therefore, the filter function of L2 Boosting corresponds to the truncated power series expansion of $\mathbf{\Gamma}^{-1}$.

Next, we briefly discuss three other methods.

3.2.2 Accelerated L2 Boosting

This method, also called the ν-method, can be seen as an accelerated version of L2 boosting. The coefficients are found by setting $\mathbf{C}^0 = 0$, $\omega_1 = (4\nu+2)/(4\nu+1)$, $\mathbf{C}^1 = \mathbf{C}^0 + \frac{\omega_1}{n}(\mathbf{Y} - \mathbf{\Gamma}\mathbf{C}^0)$ and considering for $i = 2, \ldots, t$ the iteration given by

$$\mathbf{C}^i = \mathbf{C}^{i-1} + u_i(\mathbf{C}^{i-1} - \mathbf{C}^{i-2}) + \frac{\omega_i}{n}(\mathbf{Y} - \mathbf{\Gamma}\mathbf{C}^{i-1})$$

$$u_i = \frac{(i-1)(2i-3)(2i+2\nu-1)}{(i+2\nu-1)(2i+4\nu-1)(2i+2\nu-3)}$$

$$\omega_i = 4\frac{(2i+2\nu-1)(i+\nu-1)}{(i+2\nu-1)(2i+4\nu-1)}.$$

The parameter ν is usually set to 1. The iteration is composed of three terms: the solution at the previous step, an *inertia* term that updates the solution in the same direction of the previous update, and the gradient term. Differently from the L2 Boosting, here the coefficients u_i and ω_i change at each iteration, converging for $t \to \infty$ to 1 and 4 respectively. The corresponding filter function is $g_t(\sigma) = p_t(\sigma)$ with p_t a polynomial of degree $t-1$. The derivation of the filter function is considerably more complicated and is given in (Engl et al., 1996), where this method is also proven to be faster than L2 Boosting, because the regularization parameter is the *inverse square* of the iteration number, rather than the inverse of the iteration number. In other words the ν-method can find in $\sqrt{t}$ steps the same solution found by L2 Boosting after t steps.

3.2.3 Iterated Tikhonov

This method is a combination of Tikhonov regularization and L2 Boosting. We set $\mathbf{C}^0 = 0$ and consider for $i = 0, \ldots, t$ the iteration

$$(\mathbf{\Gamma} + n\lambda\mathbf{I})\mathbf{C}^i = \mathbf{Y} + n\lambda\mathbf{C}^{i-1}.$$

Therefore, at each step, we obtain the solution

$$\mathbf{C}^i = (\mathbf{\Gamma} + n\lambda\mathbf{I})^{-1}\left(\mathbf{Y} + n\lambda\mathbf{C}^{i-1}\right).$$

The corresponding filter function is:

$$g_t(\sigma) = \frac{(\sigma + \lambda n)^t - (\lambda n)^t}{\sigma(\sigma + \lambda n)^t}.$$

Obviously, this method is computationally very demanding, since at each iteration we have to invert the matrix ($\mathbf{\Gamma}+n\lambda\mathbf{I}$). From another perspective, this methods requires to solve a Tikhonov problem per each iteration. Differently from Tikhonov regularization, though, Iterated Tikhonov is much more capable of exploiting the regularity of the solution (Lo Gerfo et al., 2008). This ability is expressed by a parameter $\overline{\nu}$, called qualification number, whose precise definition will be given in the next section, and plays an important role in the bound on the excess risk of the estimator.

3.2.4 Truncated Singular Values Decomposition

This method, also called Spectral Cut-off, is equivalent to (kernel) principal component regression (Schölkopf and Smola, 2001) and is akin to a projection onto the first principal components of the kernel matrix $\mathbf{\Gamma}$, in a vector valued setting. The number of components depends on the regularization parameter, λ. In fact, only the directions corresponding to eigenvalues greater than λ/n are kept, while the remaining discarded. The filter function is defined as

$$g_\lambda(\sigma) = \begin{cases} \frac{1}{\sigma} & \text{if } \sigma \geq \frac{\lambda}{n} \\ 0 & \text{otherwise} \end{cases}$$

The algorithm is based on the following simple idea. Perform the eigen-decomposition the kernel matrix $\mathbf{\Gamma} = \mathbf{USU}^T$, where $\mathbf{S} = diag(\sigma_1, \ldots, \sigma_{nd})$, with $\sigma_i \geq \sigma_{i+1}$. Replace the eigenvalues smaller than the threshold λ/n with 0. Then, the coefficients of the estimator are given by

$$\mathbf{C} = g_\lambda(\mathbf{\Gamma})\mathbf{Y},$$

where $g_\lambda(\mathbf{\Gamma}) = \mathbf{U}g_\lambda(\mathbf{S})\mathbf{U}^T$ and $g_\lambda(\mathbf{S}) = diag(1/\sigma_1, \ldots, 1/\sigma_m, 0, \ldots)$, where σ_m is the smallest eigenvalue greater or equal than λ/n. A variant of the algorithm is obtained by fixing the number of principal components to retain, instead of setting the threshold via λ.

3.3 Excess Risk for Spectral Regularization

In this section, we present a finite sample bound on the excess risk of the estimators obtained with spectral filters. The bound leads to consistency. In order to prove such a result in a unified framework for the various algorithms, we need a formal definition of admissible filter function. This definition is general, but it is given in terms of specific constants that might change from one algorithm to the other (Bauer et al., 2007).

For the sake of simplicity, we assume that the minimizer of the expected risk belongs to $\mathcal{H}$ and denote it with f_ρ. The excess risk is defined as the difference between the expected risk of the estimator $f_{\mathbf{z}}^\lambda$ and the expected risk of f_ρ, $\mathrm{I}(f_{\mathbf{z}}^\lambda) - \mathrm{I}(f_\rho)$.

Since $\mathrm{I}(f_\rho)$ is the smallest expected risk, the excess risk measures how good a learning algorithm is at producing estimators with small expected risk. A bound on the excess risk quantitatively informs us on the quality of a learning algorithm (or in our case, a class of learning algorithms). The bound we present depends on the number of training examples, n, and on several specific constants that vary from one spectral filter to the other. If the bound decreases to zero as the number of examples increases, we say that the algorithm is *consistent*.

Due to the use of the square loss, the bound on the excess risk also tells us how well the estimator obtained with a certain algorithm approximates the best estimator, f_ρ. In fact, we can write the excess risk as

$$\mathrm{I}(f_{\mathbf{z}}^\lambda) - \mathrm{I}(f_\rho) = \int_{\mathcal{X}\times\mathcal{Y}} (f_{\mathbf{z}}^\lambda(x) - y)^2 \rho(y|x)\rho_{\mathcal{X}}(x)dydx - \int_{\mathcal{X}\times\mathcal{Y}} (f_\rho(x) - y)^2 \rho(y|x)\rho_{\mathcal{X}}(x)dydx.$$

Recalling that $f_\rho(x) = \int_{\mathcal{Y}} y\rho(y|x)dy$, we obtain

$$\mathrm{I}(f_{\mathbf{z}}^\lambda) - \mathrm{I}(f_\rho) = \int_{\mathcal{X}} (f_{\mathbf{z}}^\lambda(x) - f_\rho(x))^2 \rho_{\mathcal{X}}(x)dx,$$

which corresponds to $||f_{\mathbf{z}}^\lambda - f_\rho||^2_{L2(\mathcal{X},\rho_{\mathcal{X}}(x)dx)}$, that is the squared distance between $f_{\mathbf{z}}^\lambda$ and f_ρ in the space of square-integrable functions defined on $(\mathcal{X},\, \rho_{\mathcal{X}}(x)dx)$.

Before stating our theorem, we need to make some preliminary assumptions. More precisely, the reproducing kernel is assumed to be bounded, recalling (2.6)

$$\sup_{x\in\mathcal{X}} ||\Gamma(x,x)||_{\mathcal{Y},\mathcal{Y}} = \kappa < \infty.$$

The input space $\mathcal{X}$ is a separable metric space (not necessarily compact). The output space $\mathcal{Y}$ is a bounded set in $\mathbb{R}^T$, that is $\sup_{y\in\mathcal{Y}} ||y||_T = M < \infty$.

Given the above assumptions, the definition of an *admissible* filter function is the following.

Definition 1. *We say that a filter $g_\lambda : [0,\kappa^2] \to \mathbb{R}$, $0 < \lambda \leq \kappa^2$, is* admissible *if the following conditions hold:*

- *There exists a constant D such that*

$$\sup_{0<\sigma\leq\kappa^2} |\sigma g_\lambda(\sigma)| \leq D \tag{3.6}$$

- *There exists a constant B such that*

$$\sup_{0<\sigma\leq\kappa^2} |g_\lambda(\sigma)| \leq \frac{B}{\lambda} \tag{3.7}$$

- *There exists a constant γ such that*

$$\sup_{0<\sigma\leq\kappa^2} |1 - g_\lambda(\sigma)\sigma| \leq \gamma \tag{3.8}$$

- *There exists a constant $\overline{\nu} > 0$, namely the* qualification *of the filter g_λ, such that, for any $\nu \in (0,\overline{\nu}]$, we have*

$$\sup_{0<\sigma\leq\kappa^2} |1 - g_\lambda(\sigma)\sigma|\sigma^\nu \leq \gamma_\nu \lambda^\nu, \tag{3.9}$$

  where the constant $\gamma_\nu > 0$ does not depend on λ.

We refer to (Lo Gerfo et al., 2008; Bauer et al., 2007) for a discussion of the above conditions and state our main theorem.

Theorem 1. *Let $\mathbf{z} = \{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}_{i=1}^n$ be a training set drawn i.i.d. according to $\rho(x,y)$ and let $f_{\mathbf{z}}^{\lambda}$ be the estimator obtained with a spectral filter g_λ according to (3.3). Assume $\overline{\nu} \geq \frac{1}{2}$ and $\|f_\rho\|_\Gamma \leq R$. Given $\eta > 0$, choose the regularization parameter $\lambda_n = \lambda(n)$ as*

$$\lambda_n = \frac{1}{\sqrt{n}} 2\sqrt{2}\kappa^2 \log\frac{4}{\eta}. \tag{3.10}$$

If we let $f_{\mathbf{z}} = f_{\mathbf{z}}^{\lambda_n}$, then with probability greater than $1-\eta$, we have

$$\mathrm{I}(f_{\mathbf{z}}) - \mathrm{I}(f_\rho) \leq \frac{C \log 4/\eta}{\sqrt{n}}, \tag{3.11}$$

where $C = 4\sqrt{2}\big(\gamma + \gamma_{\frac{1}{2}}\big)^2\kappa^2R^2 + 4\sqrt{2}(\kappa M + R)^2(B + \sqrt{BD})^2$.

The above result generalizes the analysis in (Bauer et al., 2007; Caponnetto and De Vito, 2007), to which we refer for the computation of the constants corresponding to each algorithm, and it leads to the following consistency result.

Theorem 2. *Let $\mathbf{z} = \{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y}\}_{i=1}^n$ be a training set drawn i.i.d. according to ρ and $f_{\mathbf{z}}^{\lambda}$ be the estimator obtained with a spectral filter g_λ according to (3.3). Assume $\overline{\nu} \geq \frac{1}{2}$ and $\|f_\rho\|_\Gamma \leq R < +\infty$. Choose the regularization parameter $\lambda_n = \lambda(n)$ such that*

$$\lim_{n\to+\infty} \lambda_n = 0, \tag{3.12}$$

$$\lim_{n\to+\infty} \lambda_n\sqrt{n} = +\infty. \tag{3.13}$$

Let $f_{\mathbf{z}} = f_{\mathbf{z}}^{\lambda_n}$, then, for any $\varepsilon > 0$,

$$\lim_{n\to+\infty} \mathrm{P}\left[\, I(f_{\mathbf{z}}) - I(f_\rho) > \varepsilon\right] = 0. \tag{3.14}$$

We give the proofs of both theorems in the next section and add three remarks.

Our proof is based on the assumption that the minimizer f_ρ of the expected risk belongs to $\mathcal{H}$. Even when the expected risk does not achieve a minimum in $\mathcal{H}$, one can still show that there is a parameter choice λ_n ensuring convergence to $\inf_{f\in\mathcal{H}} \mathrm{I}(f)$ – see (Caponnetto, 2006).

Second, if we strengthen the assumptions on the problem we can obtain faster convergence rates. For example, if $L_\Gamma : L^2(X, \rho_{\mathcal{X}}(x)dx) \to L^2(X, \rho_{\mathcal{X}}(x)dx)$,

$$(L_\Gamma f)(x) = \int_{\mathcal{X}} \Gamma(x,s) f(s) \rho_{\mathcal{X}}(s) ds,$$

is the integral operator with kernel Γ, we can consider the assumption[1] $f_\rho = L_\Gamma^r u$ for some $u \in L^2(X, \rho_{\mathcal{X}}(x)dx)$. In this case, by choosing $\lambda_n = n^{-\frac{1}{2r+1}}$, we can replace the rate $n^{-1/2}$ in (3.11) with $n^{-\frac{2r}{2r+1}}$, which is optimal in a minimax sense (Caponnetto and De Vito, 2007).

[1] From the theory of RKHS we know that assuming that $f_\rho \in \mathcal{H}$ exists corresponds to the index $r = 1/2$.

Third, the latter parameter choice depends on the unknown regularity index r and the question arises whether we can achieve the same rate choosing λ without any prior information on the regression function f_ρ, namely adaptively using only the information contained in the training set. Indeed this is the case, since we can directly apply the results in (De Vito et al., 2008) that propose a data-driven scheme to choose the regularizing parameter that yields optimal convergence rates for the excess risk.

3.4 Proofs of the error estimates

3.4.1 Kernel Matrix and Extension Operators

Before presenting the proofs of the bound on the excess risk, Theorem 1, and on the consistency, Theorem 2, we first recall the definitions of some operators based on the kernel.

Given a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H} \subseteq \{f : \mathcal{X} \to \mathbb{R}^d\}$, for any set of n points in $\mathcal{X}$, $\mathbf{x} = (x_1, \dots, x_n) \in \mathcal{X}^n$, we introduce the sampling operator $S_\mathbf{x} : \mathcal{H} \to \mathbb{R}^{nd}$ defined by $(S_\mathbf{x} f) := (f(x_1), \dots, f(x_n))$, where $f(x_i) \in \mathbb{R}^d$ for any $x_i \in \mathcal{X}$. The sampling operator essentially associates to a function in the hypothesis space $\mathcal{H}$ a vector containing the values of the function evaluated at each point x_i of $\mathbf{x}$. In $\mathbb{R}^{nd}$ we consider vectors of the form $\mathbf{C} = (c_1, \dots, c_n)$, with $c_i \in \mathbb{R}^d$, for all $i = 1, \dots, n$ and, for two such vectors $\mathbf{C}$ and $\mathbf{C}' = (c_1, \dots, c_n')$, we define the scalar product as

$$\langle \mathbf{C}, \mathbf{C}' \rangle_{nd} = \frac{1}{n} \sum_{i=1}^{n} \langle c_i, c_i' \rangle_d.$$

The adjoint of the sampling operator, $S_\mathbf{x}^* : \mathbb{R}^{nd} \to \mathcal{H}$, can be computed observing that, for any $\mathbf{C} = (c_1, \dots, c_n)$ and any $f \in \mathcal{H}$,

$$\langle S_\mathbf{x}^* \mathbf{C}, f \rangle_\mathcal{H} = \langle \mathbf{C}, S_\mathbf{x} f \rangle_{nd} = \frac{1}{n} \sum_{i=1}^{n} \langle c_i, f(x_i) \rangle_d = \frac{1}{n} \sum_{i=1}^{n} \langle \Gamma_{x_i} c_i, f \rangle_\mathcal{H}.$$

Therefore, for any $\mathbf{C} \in \mathbb{R}^{nd}$, we have

$$S_\mathbf{x}^* \mathbf{C} = \frac{1}{n} \sum_{i=1}^{n} \Gamma_{x_i} c_i, \tag{3.15}$$

which is a function in $\mathcal{H}$ that takes as argument a point in $\mathcal{X}$ and returns a vector in $\mathbb{R}^d$,

$$(S_\mathbf{x}^* \mathbf{C})(x) = \frac{1}{n} \sum_{i=1}^{n} \Gamma(x_i, x) c_i.$$

Note that this is the same formulation as the one given by the Representer Theorem for the minimizers of particular convex learning functionals, that depend on n examples $\{(x_i, y_i)\}_{i=1}^n$.

The kernel matrix $\mathbf{\Gamma}$ is defined as $\mathbf{\Gamma}_{ij} = \Gamma(x_i, x_j)$ for $x_i, x_j \in \mathbf{x}$ and, using the reproducing property (2.5), it can be written as $\mathbf{\Gamma} = n S_\mathbf{x} S_\mathbf{x}^*$. To check this fact, we first observe that, for any $i = 1, \dots, n$ and for any $\mathbf{C} \in \mathbb{R}^{nd}$, $(\mathbf{\Gamma C})_i = \sum_{j=1}^n \Gamma(x_i, x_j) c_j$. Secondly, using (3.15), we compute $(n S_\mathbf{x}(S_\mathbf{x}^* \mathbf{C}))_i = \sum_{j=1}^n \Gamma(x_i, x_j) c_j$. It follows that $\mathbf{\Gamma} = n S_\mathbf{x} S_\mathbf{x}^*$.

If we define $T_x := \Gamma_x \Gamma_x^* : \mathcal{H} \to \mathcal{H}$, and $T_{\mathbf{x}} := \frac{1}{n}\sum_{i=1}^n T_{x_i} = S_{\mathbf{x}}^* S_{\mathbf{x}}$, then the operator $T_{\mathbf{x}} : \mathcal{H} \to \mathcal{H}$ applied to any $f \in \mathcal{H}$ yields

$$\begin{aligned} T_{\mathbf{x}} f &= S_{\mathbf{x}}^* S_{\mathbf{x}} f \\ &= S_{\mathbf{x}}^* (f(x_1), \ldots, f(x_n)) \\ &= \frac{1}{n}\sum_{i=1}^n \Gamma(\cdot, x_i) f(x_i). \end{aligned}$$

It follows that

$$(T_{\mathbf{x}} f)(x) = \frac{1}{n}\sum_{i=1}^n \Gamma(x, x_i) f(x_i),$$

which allows us to interpret $T_{\mathbf{x}}$ as a natural out of sample extension of the kernel matrix $\mathbf{\Gamma}$.

Indeed, $T_{\mathbf{x}}$ and $\frac{1}{n}\mathbf{\Gamma}$ are positive Hilbert-Schmidt operators with the same eigenvalues, for more details see (Caponnetto and De Vito, 2007). In fact, let $\mathbf{u}_i \in \mathbb{R}^{nd}$ with $i = 1, \ldots, m$, $m \leq nd$, be the eigenvectors of $\frac{1}{n}\mathbf{\Gamma}$ associated to strictly positive eigenvalues $\sigma_i > 0$. For $i > m$, we have $\frac{1}{n}\mathbf{\Gamma}\mathbf{u}_i = 0$. For $i \leq m$, we can associate to each $\mathbf{u}_i$ a corresponding eigenfunction of $T_{\mathbf{x}}$,

$$f_i = \frac{1}{\sqrt{\sigma}} S_{\mathbf{x}}^* \mathbf{u}_i \qquad i = 1, \ldots, m. \tag{3.16}$$

such that $T_{\mathbf{x}} f_i = \sigma_i f_i$. At the same time

$$\mathbf{u}_i = S_{\mathbf{x}} f_i \qquad i = 1, \ldots, m. \tag{3.17}$$

In fact, using the first expression, we get $T_{\mathbf{x}} f_i = S_{\mathbf{x}}^* S_{\mathbf{x}} f_i = \frac{1}{\sqrt{\sigma}} S_{\mathbf{x}}^* S_{\mathbf{x}} S_{\mathbf{x}}^* \mathbf{u}_i$. But $\mathbf{u}_i$ is an eigenvector of $S_{\mathbf{x}} S_{\mathbf{x}}^*$, so that $S_{\mathbf{x}} S_{\mathbf{x}}^* \mathbf{u}_i = \sigma \mathbf{u}_i$. Performing another substitution, we obtain $T_{\mathbf{x}} f_i = \frac{1}{\sqrt{\sigma}} S_{\mathbf{x}}^* \sigma \mathbf{u}_i$ and, using again (3.16), we get the eigenfunction equation for $T_{\mathbf{x}}$, $T_{\mathbf{x}} f_i = \sigma f_i$. Using the second expression (3.17), we obtain the eigenvector equation for $\frac{1}{n}\mathbf{\Gamma}$.

The operator $T_{\mathbf{x}}$ can be viewed as a discretized version of $T = \int_{\mathcal{X}} T_x \rho_{\mathcal{X}}(x) dx$, where $\rho_{\mathcal{X}}$ is the marginal probability distribution on the input space. Recall that $\rho(x, y) = \rho(y|x)\rho_{\mathcal{X}}(x)$. T is a positive and Hilbert-Schmidt operator and

$$(Tf)(x) = \int_{\mathcal{X}} \Gamma(x, x') f(x') \rho_{\mathcal{X}}(x') dx',$$

which justifies considering the kernel matrix $\mathbf{\Gamma}$ as an empirical proxy of the integral operator T with kernel Γ.

3.4.2 Proofs

The proofs of Theorem 1 and Theorem 2 are based on the following lemmas.

First we show that the estimator can be written in a form which is more suitable for theoretical studies.

Lemma 1. *The estimator obtained with a spectral filter can be written as*

$$f_{\mathbf{z}}^{\lambda} = g_{\lambda}(T_{\mathbf{x}}) h_{\mathbf{z}},$$

with $h_{\mathbf{z}} = S_{\mathbf{x}}^* \mathbf{Y} = \frac{1}{n}\sum_{i=1}^n \Gamma_{x_i} y_i$.

Proof. We first rewrite the solution of the empirical risk minimization principle as $\mathbf{C} = \left(\frac{\mathbf{\Gamma}}{n}\right)^{-1} \frac{\mathbf{Y}}{n}$. Secondly, we write the coefficient of the solution obtained with a spectral filter as $\mathbf{C} = g_\lambda\left(\frac{\mathbf{\Gamma}}{n}\right) \frac{\mathbf{Y}}{n}$. The estimator itself can now be written as $f_{\mathbf{z}}^\lambda = S_{\mathbf{x}}^* g_\lambda\left(\frac{\mathbf{\Gamma}}{n}\right) \mathbf{Y} = S_{\mathbf{x}}^* g_\lambda\left(S_{\mathbf{x}} S_{\mathbf{x}}^*\right) \mathbf{Y}$.

If we now consider the eigen-system $(\mathbf{u}_1, \ldots, \mathbf{u}_{nd})$ of $S_{\mathbf{x}} S_{\mathbf{x}}^*$, we have that $g_\lambda(S_{\mathbf{x}} S_{\mathbf{x}}^*)\mathbf{u}_i = g_\lambda(\sigma_i)\mathbf{u}_i$, therefore

$$\begin{aligned} S_{\mathbf{x}}^* g_\lambda\left(S_{\mathbf{x}} S_{\mathbf{x}}^*\right) \mathbf{Y} &= \sum_{i=1}^{nd} S_{\mathbf{x}}^* g_\lambda(S_{\mathbf{x}} S_{\mathbf{x}}^*)\mathbf{u}_i \langle \mathbf{Y}, \mathbf{u}_i \rangle_{nd} \\ &= \sum_{i=1}^{nd} S_{\mathbf{x}}^* g_\lambda(\sigma_i)\mathbf{u}_i \langle \mathbf{Y}, \mathbf{u}_i \rangle_{nd}. \end{aligned}$$

Since $S_{\mathbf{x}}^* \mathbf{u}_i = 0$ for $i > m$, we can truncate the sum at the m-th term, and recalling (3.16) that $f_i = \frac{1}{\sqrt{\sigma_i}} S_{\mathbf{x}}^* \mathbf{u}_i$, for $i = 1, \ldots, m$, we obtain

$$\sum_{i=1}^{nd} S_{\mathbf{x}}^* g_\lambda(\sigma_i)\mathbf{u}_i \langle \mathbf{Y}, \mathbf{u}_i \rangle_{nd} = \sum_{i=1}^{m} \sqrt{\sigma_i} g_\lambda(\sigma_i) f_i \langle \mathbf{Y}, \mathbf{u}_i \rangle_{nd}.$$

Observing that $g_\lambda(\sigma_i) f_i = g_\lambda(T_{\mathbf{x}}) f_i$, we get

$$\begin{aligned} \sum_{i=1}^{m} \sqrt{\sigma_i} g_\lambda(\sigma_i) f_i \langle \mathbf{Y}, \mathbf{u}_i \rangle_{nd} &= \sum_{i=1}^{m} g_\lambda(T_{\mathbf{x}}) \sqrt{\sigma_i} f_i \langle \mathbf{Y}, \mathbf{u}_i \rangle_{nd} \\ &= g_\lambda(T_{\mathbf{x}}) \sum_{i=1}^{m} \sqrt{\sigma_i} f_i \langle \mathbf{Y}, \mathbf{u}_i \rangle_{nd} \\ &= g_\lambda(T_{\mathbf{x}}) \sum_{i=1}^{m} S_{\mathbf{x}}^* \mathbf{u}_i \langle \mathbf{Y}, \mathbf{u}_i \rangle_{nd} \end{aligned}$$

We can now extend the sum up to nd, since for $m < i < nd$, the terms $S_{\mathbf{x}}^* \mathbf{u}_i$ are equal to zero, which yields

$$\begin{aligned} g_\lambda(T_{\mathbf{x}}) \sum_{i=1}^{nd} S_{\mathbf{x}}^* \mathbf{u}_i \langle \mathbf{Y}, \mathbf{u}_i \rangle_{nd} &= g_\lambda(T_{\mathbf{x}}) S_{\mathbf{x}}^* \sum_{i=1}^{nd} \mathbf{u}_i \langle \mathbf{Y}, \mathbf{u}_i \rangle_{nd} \\ &= g_\lambda(T_{\mathbf{x}}) S_{\mathbf{x}}^* \mathbf{Y} \\ &= g_\lambda(T_{\mathbf{x}}) h_{\mathbf{z}} \end{aligned}$$

□

Let us introduce the operator $\overline{h} = T f_\rho$ and recall the following lemma from (De Vito et al., 2005a).

Lemma 2. *Let* $\kappa = \sup_{x \in \mathcal{X}} ||\Gamma(x,x)||_{\mathcal{Y},\mathcal{Y}}$, $M = sup_{y \in \mathcal{Y}} ||y||_d$ *and* $0 < \eta \leq 1$. *For any* $n \in \mathbb{N}$ *let*

$$G_\eta = \{\mathbf{z} \in (\mathcal{X} \times \mathcal{Y})^n : ||\overline{h} - h_{\mathbf{z}}||_{\mathcal{H}} \leq \delta_1, ||T - T_{\mathbf{x}}|| \leq \delta_2\},$$

with

$$\begin{aligned} \delta_1 := \delta_1(n,\eta) &= \frac{1}{\sqrt{n}} 2\sqrt{2}\kappa M \log\frac{4}{\eta} \\ \delta_2 := \delta_2(n,\eta) &= \frac{1}{\sqrt{n}} 2\sqrt{2}\kappa^2 \log\frac{4}{\eta} \end{aligned}$$

then

$$Pr(G_\eta) \geq 1 - \eta.$$

For the proof of the main theorem, we need to introduce the operator $h = T_{\mathbf{x}} f_\rho$ and the following lemma.

Lemma 3. *Let* $\kappa = \sup_{x \in \mathcal{X}} ||\Gamma(x,x)||_{\mathcal{Y},\mathcal{Y}}$, $M = sup_{y \in \mathcal{Y}} ||y||_d$ *and* $||f_\rho||_{\mathcal{H}} < R$. *For any* $0 < \eta \leq 1$ *and* $n \in \mathbb{N}$ *let*

$$G_\eta = \{\mathbf{z} \in (\mathcal{X} \times \mathcal{Y})^n : ||h - h_{\mathbf{z}}||_{\mathcal{H}} \leq \delta_3, ||T - T_{\mathbf{x}}|| \leq \delta_2\},$$

with

$$\begin{aligned} \delta_3 := \delta_1(n,\eta) &= \frac{1}{\sqrt{n}} 2\sqrt{2}(\kappa^2 R + \kappa M) \log \frac{4}{\eta} \\ \delta_2 := \delta_2(n,\eta) &= \frac{1}{\sqrt{n}} 2\sqrt{2}\kappa^2 \log \frac{4}{\eta} \end{aligned}$$

then

$$Pr(G_\eta) \geq 1 - \eta.$$

Proof. We only need to prove that $||h - h_{\mathbf{z}}||_{\mathcal{H}} \leq \delta_3$ with probability greater than $1 - \eta$, since the rest is directly derived from Lemma 2.

Using the triangle inequality, we have

$$||h - h_{\mathbf{z}}||_{\mathcal{H}} \leq ||h - \overline{h}||_{\mathcal{H}} + ||\overline{h} - h_{\mathbf{z}}||_{\mathcal{H}},$$

where the second term is bounded by δ_1 from Lemma 2. The first term can be bounded in the following way, using again Lemma 2 and the hypothesis that $||f_\rho||_{\mathcal{H}} \leq R$,

$$\begin{aligned} ||h - \overline{h}||_{\mathcal{H}} &= ||T_{\mathbf{x}} f_\rho - T f_\rho||_{\mathcal{H}} \\ &\leq ||T - T_{\mathbf{x}}||||f_\rho||_{\mathcal{H}} \\ &\leq R\delta_2. \end{aligned}$$

Combining the bounds on the two terms, we have that with probability greater than $1 - \eta$

$$\begin{aligned} ||h - h_{\mathbf{z}}||_{\mathcal{H}} &\leq R\delta_2 + \delta_1 \\ &= \frac{1}{\sqrt{n}} 2\sqrt{2}(\kappa^2 R + \kappa M) \log \frac{4}{\eta} = \delta_3. \end{aligned}$$

□

We are now ready to state the following theorem.

Theorem 3. *Let $n \in \mathbb{N}$ and $0 < \eta \leq 1$ and assume that $\nu \geq 1$, $\lambda < 1$ and*

$$\lambda \geq \frac{1}{\sqrt{n}} 2\sqrt{2}\kappa^2 \log\frac{4}{\eta}. \tag{3.18}$$

Also assume that $f_\rho \in \mathcal{H}$, $||f_\rho||_\Gamma \leq R$ and $M = sup_{y\in\mathcal{Y}}||y||_d$. Let $f_{\mathbf{z}}^\lambda$ be the estimator obtained with a spectral filter g_λ from a training set $\mathbf{z}$. Then with probability at least $1-\eta$ we have

$$\mathrm{I}(f_{\mathbf{z}}^\lambda) - \mathrm{I}(f_\rho) \leq 2\big(\gamma + \gamma_{\frac{1}{2}}\big)^2 \lambda R^2 + 2\frac{C}{\lambda n} \tag{3.19}$$

where $C = C(\eta, \kappa, M, R, B, D) = 8\big(\kappa M + \kappa^2 R\big)^2 \big(B + \sqrt{BD}\big)^2 \big(\log\frac{4}{\eta}\big)^2$ does not depend on λ and n.

Proof. From Proposition 2 in (De Vito and Caponnetto, 2005) we have that

$$\mathrm{I}(f) - \mathrm{I}(f_\rho) = \|\sqrt{T}(f - f_\rho)\|_\Gamma^2 \tag{3.20}$$

for all $f \in \mathcal{H}$.

We assume throughout that $\mathbf{z} \in G_\eta$, as given in the previous lemma, so that the above inequalities hold true with probability at least $1-\eta$. We consider the following error decomposition

$$||\sqrt{T}(f_{\mathbf{z}}^\lambda - f_\rho)||_\Gamma^2 \leq 2||\sqrt{T}(f_{\mathbf{z}}^\lambda - f^\lambda)||_\Gamma^2 + 2||\sqrt{T}(f^\lambda - f_\rho)||_\Gamma^2 \tag{3.21}$$

where $f^\lambda = g_\lambda(T_{\mathbf{x}})h$. The first term is called the sample error, while the second approximation error. We now separately bound the two terms in the right-hand side. The first term can be decomposed as

$$\begin{aligned} \sqrt{T}(f_{\mathbf{z}}^\lambda - f^\lambda) = \sqrt{T} g_\lambda(T_{\mathbf{x}})(h_{\mathbf{z}} - h) = \\ = \sqrt{T_{\mathbf{x}}} g_\lambda(T_{\mathbf{x}})(h_{\mathbf{z}} - h) + (\sqrt{T} - \sqrt{T_{\mathbf{x}}}) g_\lambda(T_{\mathbf{x}})(h_{\mathbf{z}} - h) \end{aligned} \tag{3.22}$$

The inequality

$$||\sqrt{T} - \sqrt{T_{\mathbf{x}}}|| \leq \sqrt{||T - T_{\mathbf{x}}||} \leq \sqrt{\delta_2} \leq \sqrt{\lambda} \tag{3.23}$$

follows from Theorem 1 in (Mathé and Pereverzev, 2002), Lemma 3 and assumption (3.18). Furthermore, using the properties (3.6) and (3.7) of an admissible filter and standard results of spectral theory, it is easy to show that

$$||\sqrt{T_{\mathbf{x}}} g_\lambda(T_{\mathbf{x}})|| \leq \frac{\sqrt{BD}}{\sqrt{\lambda}}. \tag{3.24}$$

Using these two results, (3.23) and (3.24), property (3.7), we can bound the norm of (3.22) using Lemma 3

$$||\sqrt{T}(f_{\mathbf{z}}^\lambda - f^\lambda)||_\Gamma \leq \frac{1}{\sqrt{\lambda}}(B + \sqrt{BD})\delta_3 \tag{3.25}$$

We now deal with the second term in the r.h.s of (3.21). We can write

$$\begin{aligned} \sqrt{T}(f_\rho - f^\lambda) &= \sqrt{T}(I - g_\lambda(T_{\mathbf{x}})T_{\mathbf{x}})f_\rho \\ &= (\sqrt{T} - \sqrt{T_{\mathbf{x}}})(I - g_\lambda(T_{\mathbf{x}})T_{\mathbf{x}})f_\rho + \\ &\quad \sqrt{T_{\mathbf{x}}}(I - g_\lambda(T_{\mathbf{x}})T_{\mathbf{x}})f_\rho \end{aligned} \tag{3.26}$$

We can bound this term using (3.23) and the following properties of admissible filters, that can be derived from (3.8) and (3.9),

$$\begin{aligned} ||I - g_\lambda(T_{\mathbf{x}})T_{\mathbf{x}}|| &\leq \gamma, \\ ||(I - g_\lambda(T_{\mathbf{x}})T_{\mathbf{x}})\sqrt{T_{\mathbf{x}}}|| &\leq \gamma_{\frac{1}{2}}\sqrt{\lambda}, \end{aligned}$$

to obtain

$$||\sqrt{T}(f^\lambda - f_\rho)||_{\Gamma} \leq (\gamma + \gamma_{\frac{1}{2}})\sqrt{\lambda}R \tag{3.27}$$

The estimate in (3.19) follows plugging (3.27) and (3.25) into (3.21) and using the definition of δ_3.

□

We are now ready to give the proof of Theorem 1.

Proof. From Theorem 3 we have that the bound on the excess risk

$$\mathrm{I}(f_{\mathbf{z}}^\lambda) - \mathrm{I}(f_\rho) \leq 2\big(\gamma + \gamma_{\frac{1}{2}}\big)^2\lambda R^2 + 2\frac{C}{\lambda n}$$

is composed of two terms. The first term depends on λ, while the second on $\frac{1}{\lambda n}$. In order to get the best error, we should choose the value of λ which guarantees that the two terms tend to zero with the same order in n. That is

$$\begin{aligned} \lim_{n\to\infty} \lambda &= 0 \\ \lim_{n\to\infty} \frac{1}{\lambda n} &= 0 \\ \lim_{n\to\infty} \frac{\lambda}{\frac{1}{\lambda n}} &= c > 0. \end{aligned}$$

It then follows that

$$\lambda_n = O\left(\frac{1}{\sqrt{n}}\right),$$

and, in order to be consistent with condition (3.18), we can choose the following value for λ_n

$$\lambda_n = \frac{1}{\sqrt{n}} 2\sqrt{2}\kappa^2 \log\frac{4}{\eta}.$$

Substituting λ_n in (3.19) we get the rate (3.11). □

Now we give the proof of the consistency result, Theorem 2.

Proof. Let choose $\eta > 0$ such that condition (3.10)

$$\lambda_n = \frac{1}{\sqrt{n}} 2\sqrt{2}\kappa^2 \log\frac{4}{\eta}$$

of Theorem 1 holds. Hence, we must set

$$\eta_n = 4e^{-\frac{\lambda_n\sqrt{n}}{2\sqrt{2}\kappa^2}}.$$

Due to hypothesis (3.13), $\lim_{n\to+\infty} \lambda_n\sqrt{n} = +\infty$, we have that $\lim_{n\to+\infty} \eta_n = 0$ and thanks to Theorem 1 we have

$$\mathrm{P}\left[I(f_{\mathbf{z}}) - I(f_\rho) \leq \frac{C\log\frac{4}{\eta_n}}{\sqrt{n}} \right] \geq 1 - \eta_n.$$

Observe now that

$$\lim_{n\to+\infty} \frac{C\log\frac{4}{\eta_n}}{\sqrt{n}} = 0,$$

since

$$\frac{\log\frac{4}{\eta_n}}{\sqrt{n}} = \lambda_n 2\sqrt{2}\kappa^2,$$

and $\lim_{n\to+\infty} \lambda_n = 0$ for hypothesis (3.12).

Therefore it exists a $n_\varepsilon > 0$ such that $\frac{C\log\frac{4}{\eta_n}}{\sqrt{n}} \leq \varepsilon$ for $n \geq n_\varepsilon$. Putting all together we have

$$\begin{aligned} \mathrm{P}\left[I(f_{\mathbf{z}}) - I(f_\rho) > \varepsilon \right] &= 1 - \mathrm{P}\left[I(f_{\mathbf{z}}) - I(f_\rho) \leq \varepsilon \right] \\ &\leq 1 - \mathrm{P}\left[I(f_{\mathbf{z}}) - I(f_\rho) \leq \frac{C\log\frac{4}{\eta_n}}{\sqrt{n}} \right] \leq \eta_n \end{aligned}$$

and the thesis follows since $\lim_{n\to\infty} \eta_n = 0$.

□

3.5 Eigen-decomposition for Matrix-valued Kernels

Before discussing complexity issues, we describe some specific properties of kernels of the form $\Gamma(x, x') = K(x, x')A$, with K a scalar kernel and A a positive semi-definite $d \times d$ matrix — see Section 2.3. The main point we make is that, for this class of kernels, we can use the eigen-system of the matrix A to define a new coordinate system where the problem can be solved in a easier way.

We start observing that, if we denote with $u_1, \ldots, u_d \in \mathbb{R}^d$ the orthonormal basis of eigenvectors of A, given any vector $\mathbf{C} = (c_1, \ldots, c_n)$, with $c_i \in \mathbb{R}^d$, we have that $c_i = \sum_{j=1}^d \langle c_i, u_j \rangle_d u_j$ and

$$\mathbf{C} = \sum_{j=1}^d \tilde{c}^j \otimes u_j, \tag{3.28}$$

where $\tilde{c}^j = (\langle c_1, u_j \rangle_d, \ldots, \langle c_n, u_j \rangle_d)$ and $\otimes$ is the tensor product.

Similarly

$$\mathbf{Y} = \sum_{j=1}^d \tilde{y}^j \otimes u_j, \tag{3.29}$$

with $\tilde{y}^j = (\langle y_1, u_j \rangle_d, \ldots, \langle y_n, u_j \rangle_d)$. The above transformations are simply rotations in the output space. Moreover, for the considered class of kernels, the kernel matrix $\mathbf{\Gamma}$ is given by the tensor product of the $n \times n$ scalar kernel matrix $\mathbf{K}$ and A, that is $\mathbf{\Gamma} = \mathbf{K} \otimes A$.
If we denote with λ_i, v_i $(i = 1, \ldots, n)$, the eigenvalues and eigenvectors of $\mathbf{K}$ respectively and with σ_j $(j = 1, \ldots, d)$ the eigenvalues of A, we have the following result.

Proposition 2. *The solution of the vector valued problem* $\mathbf{C} = g_\lambda(\mathbf{\Gamma})\mathbf{Y}$ *can be obtained by solving* d *scalar problems*

$$\tilde{c}^j = g_\lambda(\sigma_j \mathbf{K})\tilde{y}^j, \qquad j = 1, \dots, d \tag{3.30}$$

Proof. Substituting (3.28) and (3.29) into $\mathbf{C} = g_\lambda(\mathbf{\Gamma})\mathbf{Y}$, we obtain

$$\sum_{j=1}^{d} \tilde{c}^j \otimes u_j = \sum_{j=1}^{d} g_\lambda(\mathbf{K} \otimes A)\tilde{y}^j \otimes u_j.$$

Working in the eigen-system $v_i \otimes u_j$ ($i = 1, \dots, n$ and $j = 1, \dots, d$) of the matrix $\mathbf{K} \otimes A$ and recalling that the spectral filters operate on the eigenvalues of the kernel matrix, we have

$$\sum_{j=1}^{d} \tilde{c}^j \otimes u_j = \sum_{j=1}^{d} \sum_{i=1}^{n} g_\lambda(\lambda_i \sigma_j)\langle \tilde{y}^j, v_i \rangle v_i \otimes u_j.$$

Considering only the eigenvectors of the matrix A, we obtain

$$\sum_{i=1}^{n} g_\lambda(\lambda_i \sigma_j)\langle \tilde{y}^j, v_i \rangle v_i = g_\lambda(\sigma_j \mathbf{K})\tilde{y}^j.$$

Therefore, we now have

$$\sum_{j=1}^{d} \tilde{c}^j \otimes u_j = \sum_{j=1}^{d} g_\lambda(\sigma_j \mathbf{K})\tilde{y}^j \otimes u_i,$$

and, since the eigenvectors u_j are orthonormal, the two sides of the equation must be equal term by term. It follows that

$$\tilde{c}^j = g_\lambda(\sigma_j \mathbf{K})\tilde{y}^j, \qquad j = 1, \dots, d. \tag{3.31}$$

□

The above equation shows that in the new coordinate system $\{u_1, \dots, u_d\}$, we have to solve d essentially independent problems. Indeed, after rotating the outputs (and the coefficients) the only coupling is the rescaling of each kernel matrix by σ_j. For example, in the case of Tikhonov regularization, the j-th component is found solving

$$\tilde{c}^j = (\sigma_j \mathbf{K} + \lambda n \mathbf{I})^{-1}\tilde{y}^j = (\mathbf{K} + \frac{\lambda}{\sigma_j} n \mathbf{I})^{-1}\frac{\tilde{y}^j}{\sigma_j}$$

and we see that the scaling term is changing the scale of the regularization parameter and of the outputs. The above calculation shows that all kernels of this form allow for a simple implementation at the price of the eigen-decomposition of the matrix A. Also, it shows that the coupling among the different tasks can be seen as a rotation and rescaling of the output points.

3.6 Model Selection

In practice, the choice of a consistent algorithm is not sufficient to guarantee good results. It is therefore essential to introduce a robust methodology to select the model parameter not

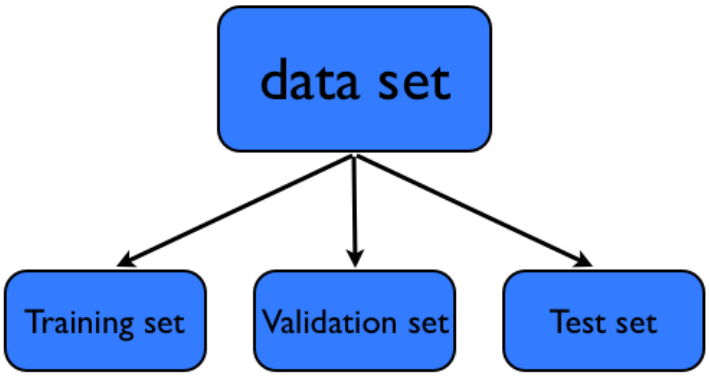


Figure 3.1: Data splitting for large data sets

susceptible of selection bias [3]. Therefore, we need a robust estimate of the generalization error. In general, the error obtained on an independent data set, i.e. not seen during the learning phase, is used as an estimate of the generalization error.

Ideally, if the data set size is sufficiently large, we can perform hold-out validation and testing, by splitting the samples into training, validation and test set, see Figure 3.1. This allows us to run the algorithm on the training set for different values of the parameter, each time evaluating the error committed on the (independent) validation set. Validation error is thus an unbiased estimate of the prediction error and can be minimized to obtain the optimal value of the parameter. Finally, a good estimate of the algorithm performance is provided by the error committed on the test set by the solution obtained with the optimal parameter.

In most cases, however, the number of available samples does not allow for such a hold-out procedure, and some expediency is needed for guaranteeing unbiased parameter selection. Generalization error is thus estimated by "resampling" the data set; the most common resampling technique is *cross-validation* (Hjorth, 1994). Cross-validation (leave-one-out or k-fold cross-validation) is a form of iterative hold-out testing which repeatedly uses part of the available data to fit the model, and a different part to validate it. We now show how to apply cross-validation for model selection in two conditions that differ in data size.

A first, and very common, situation is found when the number of available samples allows only for one hold-out procedure, and the samples are hence partioned into two sets, one will be used for training and validation, while the other will be used for testing; due to the absence of an independent data set for validation, cross-validation is applied to the the first set of data to estimate the generalization error, see Figure 3.2. The data set is initially divided in training and test set. The training set is further partitioned in k subsamples $\Psi_1, \ldots, \Psi_k$ with k depending on the cardinality of the training set. In Stage I, for each subsample Ψ_i, a model is first built using as training set the remaining $k-1$ subsamples with the parameter (e.g. regularization parameter, number of iterations or kernel parameter) ranging on a grid in the parameters space, and then validated on Ψ_i. For each parameter value the validation error is estimated as the average error over the k subsamples. Finally, the optimal parameter valueis selected as the minimizer of the validation error. In Stage II, a model is built on the entire training set with the optimal parameter and the generalization error is evaluated on the remaining test set.

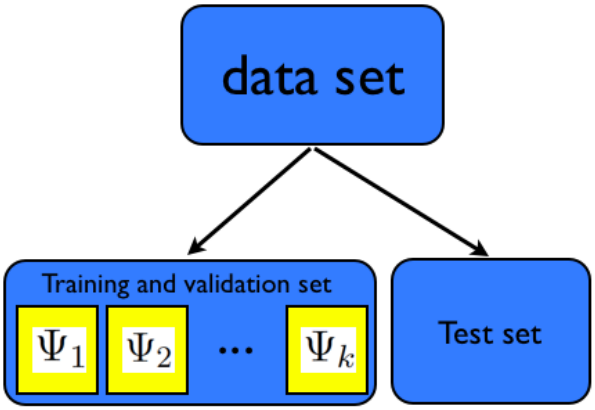


Figure 3.2: Data splitting for intermediate data sets

A second, extreme situation, is found whenever the data is so scarce that we cannot split it in two sets, training and test, without them containing too few samples. In this case, we apply a protocol based on two steps of cross-validation: we first split the data set in B subsets, using $B-1$ subsets for training and the remaining for testing. The current training set is further divided in k subsets, used for cross-validation for model selection, as sketched in Figure 3.3. The optimal parameter value is then used to train a predictive model in the entire training set of $B-1$ subsets and its generalization error assessed on the test set, composed of the remaining part. The procedure is repeated until all B subsets are used for testing and the overall model assessment is given by the average of the test errors computed for each split.

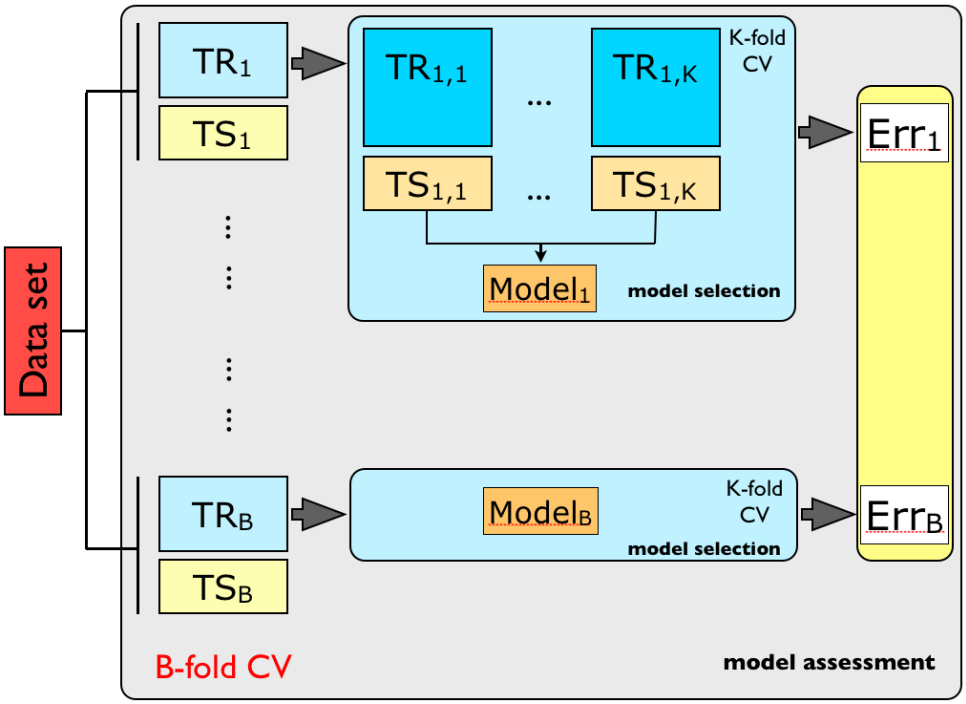


Figure 3.3: The bias-aware model selection and assessment protocol applied when dealing with very small data sets.

We thus obtain a unbiased estimate of the generalization error of a predictive model trained on the data set, although losing the uniqueness of the estimator. In fact, the protocol provides us with B different estimators. However, the aim of this protocol is to use the estimated generalization error for comparison with other models. If one is only interested in building a predictive model, a single loop of cross-validation is sufficient to select the optimal model parameter value.

3.7 Regularization Path and Complexity

In this section we discuss the time complexity of the different algorithms. In particular we compare Tikhonov regularization with accelerated L2 boosting, since, in the scalar case, this algorithm was shown to be fast and reliable (Lo Gerfo et al., 2008). In practice, when considering the complexity of a learning algorithm that depends on one regularization parameter, it is important to take into account the cost of finding the optimal parameter value. The set of solutions corresponding to many different regularization parameter values is called the *regularization path* and, using this terminology, we are interested in discussing the complexity corresponding to the whole regularization path.

For Tikhonov regularization, in general we have to run the algorithm for any new value of the regularization parameter. For iterative algorithms each step of the algorithm corresponds to a solution for a value of the regularization parameter, so that at step N we have computed the entire regularization path up to N. Further, for each iteration, iterative methods require only matrix vector multiplication. This means that, in general, if we consider N parameter values we will have $O(N(nd)^3)$ time complexity for Tikhonov regularization and $O(N(nd)^2)$ for iterative methods.

In the special case of kernels of the form $\Gamma(x, x') = K(x, x')A$, with A positive semi-definite, the complexity of the problem can be drastically reduced. Given the result in the previous section, we can diagonalize the matrix A and then work in a new coordinate system where the kernel matrix is block diagonal and all the blocks are the same, up to a rescaling. In this case the complexity of the multi-output algorithm is essentially the same as the one of a single scalar problem – $O(Nn^3)$ for Tikhonov and $O(Nn^2)$ for iterative methods – plus the cost of computing the eigen-decomposition of A which is $O(d^3)$.

We add two comments. First, we note that for Tikhonov regularization, we can further reduce the complexity from $O(Nn^3)$ to $O(n^3)$ choosing the regularization parameter with Leave One Out Cross Validation (LOO) as described in (Rifkin and Lippert, 2007). Second, we observe that for iterative methods we also have to take into account the cost of fixing the step size. As discussed for the Landweber iteration (see Section 3.2), the step size can be chosen as $1/\sigma_{max}$ where σ_{max} is the maximum eigenvalue of the kernel matrix induced by Γ, so that we have to add the cost of computing the maximum eigenvalue (O'Leary et al., 1979).

Chapter 4

Spectral Regularization in Multi-task Learning

In this chapter we discuss the use of spectral regularization methods for a general multi-task problem. The latter problem has been proposed in the machine learning literature in (Caruana, 1997) and has recently received a certain attention. Besides methods based on the use of regularization to enforce correlation among the tasks (see references in the introduction and (Evgeniou et al., 2005; Caponnetto et al., 2008; Micchelli and Pontil, 2004)), it is worth mentioning a growing literature based on Bayesian techniques and Gaussian processes in particular – see for example (Bonilla et al., 2007; Chai et al., 2009).

In multi-task learning (MTL) the goal is to learn several correlated scalar problems simultaneously. For each task $j = 1, \dots, d$ we are given a training set of n_j examples, $S_j = (x_{ij}, y_{ij})_{i=1}^{n_j}$. The examples are often assumed to belong to the same space $\mathbb{R}^p \times \mathbb{R}$. The main difference between multi-task and vector valued learning is that in multi-task learning the training sets for each task can be different, while in vector valued learning the input examples are the same for every component — see Figure 4.1.

As in vector valued learning, we believe that if there exists a strong correlation among the tasks, by learning all of them simultaneously with a proper matrix valued kernel, we can improve the prediction abilities of the learned model. Furthermore, if the input space is the same for all tasks and each task has different input points, we can essentially augment the training set for each task by considering all training points simultaneously. As we will see in the experiments, this is a major contribution towards improving independent models trained for each task. On the other hand, this effect is appreciable only in multi-task learning, since in vector valued problems, the input examples are the same for every component of the output vector field.

To use spectral regularization techniques for multi-task problems we need to slightly adapt the derivation we proposed for learning vector valued functions. This is essentially due to the fact that, although we can simply view tasks as components of a vector valued function, now each task can have different input points. The description of vector valued RKHS given in Remark 1 turns out to be useful, since it allows to work component-wise.
Recall that according to Remark 1, we can view a RKHS of vector valued functions $f : \mathcal{X} \to \mathbb{R}^d$, as defined by a (joint) kernel $Q : (\mathcal{X} \times \Pi) \times (\mathcal{X} \times \Pi) \to \mathbb{R}$, where $\Pi = \{1, \dots, d\}$ is the index set

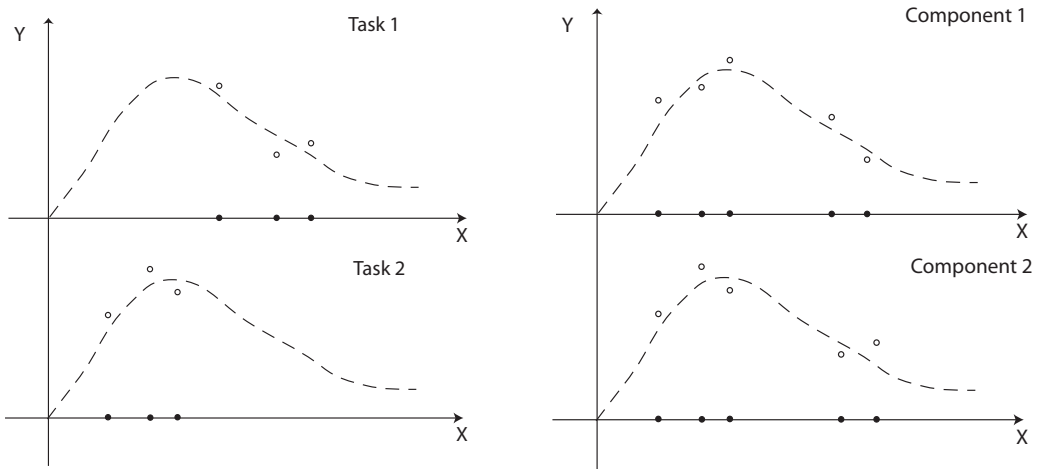


Figure 4.1: Comparison of a multi-task and a vector valued learning problem. We consider a simplified situation in which there are only two tasks/components and they are the same function. In the multi-task case, the tasks can be sampled at different input points, whereas in the vector valued case the components are sampled at the same input points.

of the output components and we have grouped $\mathcal{X}$ with Π in order to emphasize that the kernel takes as input two points, each of whom is specified by two variables.

A function in this RKHS can be written as

$$f(x,t) = \sum_i Q((x,t),(x_i,t_i))c_i,$$

with norm

$$\|f\|_Q^2 = \sum_{i,j} Q((x_j,t_j),(x_i,t_i))c_i c_j,$$

where $c_i \in \mathbb{R}$ and t_i represents which task point x_i belongs to. The functions $f(\cdot,t) = f^t(\cdot)$ are simply the components corresponding to each task and the above notation can be thought of as a component-wise definition of a vector valued function.

In view of the above representation, it is natural to reindex the training set points for all tasks into a single training set, associating to each example x_i a task indicator t_i. Hence we consider the training set $\{(x_1,y_1,t_1),\ldots,(x_N,y_N,t_N)\}$ with $N = \sum_{j=1}^d n_j$ and where, if the points are ordered task-wise, example (x_1,y_1,t_1) is example (x_{11},y_{11}) in the previous notation.

Instead of considering the sum of the empirical errors for each task (or *combined* empirical error)

$$\sum_{j=1}^d \left(\frac{1}{n_j} \sum_{i=1}^{n_j} (y_{ij} - f^j(x_{ij}))^2 \right),$$

we consider the *overall* empirical error

$$\frac{1}{N} \sum_{i=1}^N (y_i - f(x_i,t_i))^2.$$

The difference between the two approaches is on the weight they put on the errors for each task. A task with very few examples contributes more to the combined empirical error than to the overall empirical error. If the tasks are provided with the same number of examples the two errors differ only by the factor d.

The representer theorem ensures that the solution of (overall) empirical risk minimization is of the form

$$f(x,t) = f_t(x) = \sum_{i=1}^{N} Q((x,t),(x_i,t_i))c_i$$

with coefficients given by

$$\mathbf{\Gamma C} = \mathbf{Y}$$

where $\mathbf{C} = (c_1, \ldots, c_N)$, $\mathbf{\Gamma}_{ij} = Q((x_i,t_i),(x_j,t_j))$ and $\mathbf{Y} = (y_1, \ldots, y_N)$.

Directly inverting the matrix $\mathbf{\Gamma}$ leads to an unstable solution with very poor generalizing performance, in other words it overfits the training data. The spectral filters proposed in this dissertation tackle these issues by filtering its unstable components and are an alternative to Tikhonov regularization. The solution is obtained as

$$\mathbf{C} = g_\lambda(\mathbf{\Gamma})\mathbf{Y},$$

where g_λ is one of the spectral filter described in Section 3.2.

We observe that, in general, differently to vector valued regression, the matrix $\mathbf{\Gamma}$ is not a block matrix, since we compute only those entries corresponding to the tasks to which examples x_i and x_j belongs to. If they belonged to all tasks, then we would recover the $d \times d$ block $\Gamma(x_i, x_j)$ that we obtain with a matrix valued kernel for vector valued functions. For the same reason, when the kernel is $Q((x,t),(x',t')) = K(x,x')A_{t,t'}$ the kernel matrix is no longer the Kronecker product between the scalar kernel matrix $\mathbf{K}$ and A. This implies that it is no longer possible to decompose the multi-task learning problem into d scalar learning problems using the technique described in the end of Section 3.7. Therefore, as we will show with some experiments on synthetic data in Section 6.1 and real data in Section 6.4, iterative methods might be considerably more efficient than Tikhonov regularization, that requires to directly invert the $N \times N$ kernel matrix $\mathbf{\Gamma}$.

We conclude noting that, since the sampling procedures are somewhat different, the error analyses for multi-task and vector valued learning are also different. The latter case is closer to the scalar setting and was amply discussed in the previous chapter, whereas in the multi-task case the situation is more complex: one might have different cardinalities for the various tasks or be interested to evaluate performances for each task individually. Most of the few analyses (Maurer, 2006a,b; Lounici et al., 2009) consider linear regression or classification tasks, equal number of examples for each task and evaluate the average performance across all tasks. We reserve to future work the error analysis for multi-task learning.

Chapter 5

Multi-category Classification as Learning a Vector Valued Function

In this chapter we analyze multi-class problems in the framework of vector valued learning.

Multi-class, also called multi-category, problems are ubiquitous in applications. While a number of different algorithms have been proposed over the years, a theory of multi-class learning is at the beginning and most algorithms come with no theoretical guarantees in terms of generalization properties. In Section 5.1 we shortly review some previous approaches to multi-class learning, while in Section 5.2 we show that approaches based on vector valued learning are natural and help to understand the multi-class problem. In particular, we show how spectral regularization methods for vector valued learning can be used to build multi-classification rules that are Bayes consistent. We also present a thorough comparison with a recently proposed multi-class method that transforms the multi-category problem into a vector valued one, but does not completely leverage the vector valued framework. We conclude the chapter with the proof of the Bayes consistency theorem for spectral filters in the multi-class setting.

5.1 Previous approaches to multi-class problems

The algorithms previously proposed in the literature can be roughly divided into three classes. The first comprises methods based on nearest neighbor strategies (Hastie et al., 2001). These techniques are appealing for their simplicity, but are considered to be prone to over-fitting, especially in the presence of high dimensional data. The second class includes approaches where the multi-class problem is reduced to a family of binary classification problems, e.g. one versus all, where we train a binary classification model to separate one class from all the others, or all versus all (also called all pairs), where we train a binary classifier for every pair of classes. Finally, the third class corresponds to the so called *single machine* approaches. Extensive list of references can be found in (Rifkin and Klautau, 2004). The latter work gives a detailed discussion and exhaustive experimental comparisons, suggesting that one versus all might be a winning strategy both from the point of view of performances and computations (see discussion below).

From a theoretical point of view, the analysis of methods based on (penalized or constrained)

empirical risk minimization was started in (Tewari and Bartlett, 2005; Zhang, 2004). A main message of these works is that straightforward extensions of binary classification approaches might lead to methods that fail to have the property of Bayes consistency. The latter can be probably considered as a minimal theoretical requirement for a good classification rule.

In this chapter, we argue that multi-category classification can be naturally modeled as the problem of learning a vector valued function, obtained by associating each class to an appropriate coding vector. A basic observation supporting this approach is that when we describe the classes using a finite set of scalar labels, we are introducing an unnatural ordering among them, which is avoided by adopting a vector coding. Besides this fact, the idea of considering multi-category classification as the problem of learning a vector valued function is appealing since it opens the route to exploit the correlation which is present among the considered classes, and can be the key towards designing more efficient multi-class learning machines.

To better explain this last observation, we recall that among the proposed approaches to solve multi-class problems, one of the simplest, and seemingly effective, is the so called one versus all approach where a classifier is learned to discriminate each individual class from all the others. Each classifier returns a value that should quantify the affinity of an input to the corresponding output class, so that the input can be assigned to the class with highest affinity. Though extremely simple, in this method each class is learned independently to the others and the possible information about the correlation among the classes is not exploited. Indeed in several practical problems the classes are organized in homogenous groups or hierarchies. The intuition is that exploiting such an information might lead to better performances. Here we illustrate how this can be done using the framework of vector valued learning.

5.2 Multi-class as a vector valued problem

Let us start by fixing some basic concepts and notation. In multi-category classification the examples belong to either one of d classes, that is we can set $\Pi = \{1, 2, \dots, d\}$ and let $\rho(j|x)$, with $j = 1, \dots, d$, denote the conditional probability distribution for each class. The training set $\mathbf{z} = \{(x_i, y_i)\}_{i=1}^n$, with $x_i \in \mathcal{X}$ and $y_i \in \Pi$ is composed of pairs of input points and the corresponding class labels. A classifier is a function $c : \mathcal{X} \to \Pi$, assigning each input point to one of the d classes. The classification performance can be measured via the *misclassification error*

$$R(c) = \mathrm{P}\left[(x, y) \in \mathcal{X} \times \Pi : c(x) \neq y\right],$$

that is the probability that the classifier predicts the wrong class. We can rewrite the misclassification error as

$$R(c) = \int_{\mathcal{X}} \sum_{j=1}^{d} \mathbf{1}_{\{c(x) \neq j\}} \rho(j|x) \rho_{\mathcal{X}}(x) dx,$$

where $\mathbf{1}_{\{A\}}$ is the indicator function of the set A. Since c is an arbitrary function, we can compute the minimizer of the misclassification error by minimizing the error for each point x with respect to $c(x)$

$$\operatorname*{argmin}_{c(x) \in \Pi} \sum_{j=1}^{d} \mathbf{1}_{\{c(x) \neq j\}} \rho(j|x) = \operatorname*{argmin}_{c(x) \in \Pi} \left(1 - \rho(c(x)|x)\right).$$

The minimum is obviously achieved when the classifier c returns the class with highest conditional probability. We call the minimizer of the misclassification error the *Bayes rule*, defined as

$$b(x) = \operatorname*{argmax}_{j=1,\dots,d} \rho(j|x). \tag{5.1}$$

and $R(b)$ is called *Bayes misclassification error.*

Bayes consistency is then the property of a learning algorithm to return an estimator $c_{\mathbf{z}} : \mathcal{X} \to \Pi$ whose misclassification error converges to the Bayes misclassification error as the number of training examples increase. It is therefore an essential property of any supervised method for classification, since we are guaranteed that we will tend to the minimum possible classification error providing more and more training examples. Below we give a formal definition of Bayes consistency.

Definition 2. Bayes Consistency. *An algorithm is said to be* Bayes consistent *if the estimator* $c_{\mathbf{z}}$ *satisfies*

$$\lim_{n \to +\infty} \mathrm{P}\left[R(c_{\mathbf{z}}) - R(b) > \varepsilon \right] = 0 \tag{5.2}$$

for any $\varepsilon > 0$.

A standard approach for binary classification is based on viewing classification as a regression problem with binary values. Following this idea we might consider a real valued function to fit the labels $\Pi = \{1, 2, \dots, d\}$, but we would force an unnatural ordering among the classes. Another possibility is to define a *coding*, that is a one-to-one map $C : \Pi \to \mathcal{Y}$ where $\mathcal{Y} = \{\bar{\ell}_1, \dots, \bar{\ell}_d\}$ is a set of d distinct coding vectors $\bar{\ell}_j \in \mathbb{R}^d$ for $j = 1, \dots, d$. For example $\bar{\ell}_1 = (1, 0, 0, \dots, 0)$, $\bar{\ell}_2 = (0, 1, 0, \dots, 0)$ and so on. Once we fix a coding, we can use algorithms for vector valued regression to learn from the data whose outputs are given by the coding vectors. In practice, the algorithm will return an estimator that takes values in the whole space $\mathbb{R}^d$, rather than in the set of coding vectors. Therefore, we need to define a classification rule that takes as input a vector in $\mathbb{R}^d$ and returns one of the d labels $\Pi = \{1, 2, \dots, d\}$. In the binary case, a classification rule is usually defined by taking the *sign* of the estimator. In the vector valued case there is no obvious strategy.

In summary the use of vector valued learning for multi-class problems requires the following choices:

1. a coding scheme,
2. a vector learning algorithm,
3. a classification rule.

If we measure errors using the square loss, a simple calculation guides us through some of the above choices. Given a vector valued function $f : \mathcal{X} \to \mathbb{R}^d$, we can use upper indexes to indicate vector components, so that the square loss can be written as $\|\bar{\ell} - f(x)\|_d^2 = \sum_{j=1}^d (\bar{\ell}^j - f^j(x))^2$. Note that, since the coding is a one-to-one map, the conditional probability distribution $p(y|x)$ is concentrated on the $\bar{\ell}_j$s and the conditional probability distribution for each coding vector $\bar{\ell}_j$ is given by $\rho(j|x)$ for all $j = 1, \dots, d$.

Therefore, the expected risk can be written as

$$\mathrm{I}(f) = \int_{\mathcal{X}\times\mathcal{Y}} \|y - f(x)\|_d^2 \rho(y|x)\rho_{\mathcal{X}}(x)dydx = \int_{\mathcal{X}} \sum_{j=1}^{d} \|\bar{\ell}_j - f(x)\|_d^2 \rho(j|x)\rho_{\mathcal{X}}(x)dx$$

and is minimized by the regression function, f_ρ, that we can express as

$$f_\rho(x) = (f_\rho^1(x), f_\rho^2(x) \ldots, f_\rho^d(x)) = \int_{\mathcal{Y}} y\rho(y|x)dy = \sum_{j=1}^{d} \bar{\ell}_j \rho(j|x).$$

Given a general coding

$$\bar{\ell}_1 = (a, b, \ldots, b),\ \bar{\ell}_2 = (b, a, b, \ldots, b), \ldots,\ \bar{\ell}_d = (b, b, \ldots, b, a), \quad a > b, \tag{5.3}$$

we can write the r-th component of the regression function as

$$\begin{aligned} f_\rho^r(x) &= \sum_{j=1}^{d} \bar{\ell}_j^r \rho(j|x) = \sum_{j\neq r}^{d} b\rho(j|x) + a\rho(r|x) \\ &= \sum_{j=1}^{d} b\rho(j|x) - b\rho_r + a\rho_r = (a-b)\rho(r|x) + b, \end{aligned}$$

where we used the fact that $\sum_{j=1}^{d} \rho(j|x) = 1$. It follows that each component of the regression function is proportional to the conditional probability of the corresponding label, so that, recalling the definition of the Bayes rule, we have

$$b(x) = \operatorname*{argmax}_{j=1,\ldots,d} f_\rho^j(x). \tag{5.4}$$

The above calculation is simple, but shows us three useful facts. First, vector learning algorithms approximating the regression function can be used to learn the Bayes rule for a multi-class problem. Second, in this view the choice of the coding can be quite general, see (5.3). Third, once we obtained an estimator for the regression function, Equation (5.4) shows that the natural way to define a classification rule is to take the argmax of the components of the estimator.

The above discussion shows that, provided a coding of the form (5.3), we can use spectral regularization methods to solve multi-class problems and use (5.4) to obtain a classification rule. We recall that the vector valued estimator obtained with a spectral filter is $f_{\mathbf{z}}^\lambda = g_\lambda(\mathbf{\Gamma})\mathbf{Y}$, where $\mathbf{\Gamma}$ is the Gram matrix computed on the training input examples and $\mathbf{Y} = (y_1, \ldots, y_n)$ is the nd-dimensional vector where we concatenate the vector codes assigned to each example of the training set. The spectral filters g_λ are characterized by the constants presented in Section 3.3. Here, we are interested in the qualification number $\overline{\nu}$ (3.9) that controls how well a spectral filter can exploit the regularity of the regression function (Bauer et al., 2006). Equation (5.4) and the fact that spectral regularization algorithms approximate the regression function, suggest that Bayes consistency can be achieved by spectral regularization methods. Indeed this can be proved using the results in Section 3.3, in (Tewari and Bartlett, 2005) and (Zhang, 2004).

Theorem 4. *Let* $\mathbf{z} = \{(x_i, y_i) \in \mathcal{X} \times \Pi\}_{i=1}^n$, *where* $\Pi = \{1, \ldots, d\}$, *be a training set drawn i.i.d. according to* $\rho(x, y)$ *and let* $C : \Pi \to \mathcal{Y} \subset \mathbb{R}^d$ *be a coding map from class labels to coding vectors. Consider the training set* $\mathbf{z}' = \{(x_i, C(y_i)) \in \mathcal{X} \times \mathcal{Y}\}_{i=1}^n$. *Let* $f_{\mathbf{z}'} : \mathcal{X} \to \mathbb{R}^d$ *be the vector valued estimator obtained with a spectral filter* g_λ *of qualification number* $\overline{\nu} \geq \frac{1}{2}$. *Assume* $f_\rho \in \mathcal{H}$ *and* $\sup_{x \in \mathcal{X}} ||\Gamma(x, x)||_{\mathcal{Y},\mathcal{Y}} = \kappa < \infty$. *Choose the regularization parameter* $\lambda_n = \lambda(n)$ *such that*

$$\lim_{n \to +\infty} \lambda_n = 0 \tag{5.5}$$

$$\lim_{n \to +\infty} \lambda_n \sqrt{n} = +\infty. \tag{5.6}$$

If we let b *be the Bayes rule (5.1),* $f_{\mathbf{z}'} = f_{\mathbf{z}'}^{\lambda_n}$ *and* $c_{\mathbf{z}} = \operatorname{argmax}_{j=1\ldots,d} f_{\mathbf{z}'}^j$, *then*

$$\lim_{n \to +\infty} \mathrm{P}\left[R(c_{\mathbf{z}}) - R(b) > \varepsilon \right] = 0, \tag{5.7}$$

for any $\varepsilon > 0$.

We add three comments. First, the proof of the above result is given in the next section and is based on the bound given in Theorem 1 together with a so called *comparison* result (Zhang, 2004) relating expected risk and misclassification error. Second, we note that the above result allows us to derive Bayes consistency, but no convergence rates, since they would depend on the specific form of ψ. Further investigations are left to future work. Third, in the above result we made the simplifying assumption that f_ρ is in $\mathcal{H}$. In fact, if the kernel is universal (Caponnetto et al., 2008) such an assumption can be dropped and (universal) Bayes consistency can be proved with similar techniques (Caponnetto, 2006).

5.2.1 Comparison with (Lee et al., 2004)

For loss functions other than the square loss the above conclusions are not so straightforward. One of the few cases where similar results are available is the variation of the hinge loss studied in (Lee et al., 2004), where they develop a multi-category Support Vector Machine (SVM). Comparisons with this choice of the loss give further information. In fact, for this loss function, it is proved in (Lee et al., 2004) that the minimizer of the corresponding expected error is the Bayes rule itself. Such a result is based on a specific choice of the coding, that satisfies $\sum_j \bar{\ell}_i^j = 0$, for all coding vector $\bar{\ell}_i, i = 1, \ldots, d.$. This constraint is not needed for the square loss, though we note that for the square loss and the general coding (5.3), the regression function satisfies

$$\sum_{j=1}^d f_\rho^j(x) = a + (d-1)b,$$

so that, if we take $a = 1$ and $b = -1/(d-1)$, the coding proposed in (Lee et al., 2004), the sum-to-zero constraint is guaranteed. Clearly there is actually no particular reason why this coding should be preferred to any other coding of the form (5.3). Also, for the multi-class hinge loss, there is no obvious reason why taking the maximum component of an estimator should be a good strategy to build a classification rule. The relation with the conditional probabilities does not hold anymore and the magnitude of the estimator components have no clear interpretation (see (Zhang, 2004) for related results and discussions). Finally, the multi-category SVM has a

straightforward interpretation in terms of vector valued regression. In fact, we recall that in our notation such an algorithm is defined by

$$\min_{f^j \in \mathcal{H}} \{\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{d} V(f^j(x), y_i^j) + \lambda \|f^j\|_K^2\} \quad \text{s.t.} \sum_{j=1}^{d} f^j(x_i) = 0, \quad i = 1, \ldots, n$$

where V is the modified hinge loss. Inspecting the form of the above functional, and in particular the form of the penalty, it is clear that we are considering a reproducing kernel Hilbert space of vector valued functions where there is *no* coupling among the components. The only coupling among the components of the estimator is enforced by the sum-to-zero constraint. If we drop such a constraint and consider the square loss we have the following problem

$$\min_{f^j \in \mathcal{H}} \{\frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{d} (\bar{\ell}_{y_i}^j - f^j(x_i))^2 + \lambda \|f^j\|_K^2\},$$

where $\bar{\ell}_{y_i}$ is the coding vector associated to the class the example x_i belongs to. For a general coding of the form (5.3), the optimization can be done independently for each component and corresponds to classifying a given class against all the others. It is then obvious that, by taking the maximum of each component, we recover the simple one versus all scheme, albeit with a common regularizing parameter for all classes.

5.2.2 Considerations

In the case of the square loss, it is clear that to go beyond one versus all, and try to enforce some kind of correlation among the classes, different penalties have to be considered. The kernels and penalties given in Section 2.3 are a step towards this direction. In particular for kernels of the form (2.9), the matrix A can viewed as encoding the class correlations. The choice of A is therefore crucial. In certain problems the matrix A can be defined using prior information. A fine example of this situation is the object recognition dataset Caltech-256 (Griffin et al., 2007), where images of 256 object categories are collected. A hand-made taxonomy relating the categories is available and can be exploited to design the matrix A to be used, for example, with the graph-regularization kernel given in Section 2.3. Note that empirically estimating the matrix A seems to be much harder in multi-category classification than in vector valued regression. In vector valued regression one can, as a first attempt, resort to the classical Pearson correlation between the components of the vector valued function, while in multi-category classification the outputs are user defined and do not bear any information.

5.3 Proof of Theorem 4

Proof. The proof follows from the bound given in Theorem 1 together with a, so called, *comparison* result relating expected risk and misclassification error. More precisely, from Corollary 26 in (Zhang, 2004) (see also (Tewari and Bartlett, 2005)), and since we assume $f_\rho \in \mathcal{H}$, we have that

$$R(c_{\mathbf{z}}) - R(b) \leq \psi(\mathrm{I}(f_{\mathbf{z}'}) - \mathrm{I}(f_\rho)), \tag{5.8}$$

where ψ is a concave function that goes to zero at the origin.

Now, we can choose η such that condition (3.10)

$$\lambda_n = \frac{1}{\sqrt{n}} 2\sqrt{2}\kappa^2 \log\frac{4}{\eta}$$

of Theorem 1 holds. Hence, we must set

$$\eta_n = 4e^{-\frac{\lambda_n\sqrt{n}}{2\sqrt{2}\kappa^2}}.$$

Due to hypothesis (5.6), $\lim_{n\to+\infty} \lambda_n\sqrt{n} = +\infty$, we have that $\lim_{n\to+\infty} \eta_n = 0$ and, thanks to Theorem 1, we have

$$\mathrm{P}\left[I(f_{\mathbf{z}'}) - I(f_\rho) \leq \frac{C\log\frac{4}{\eta_n}}{\sqrt{n}} \right] \geq 1 - \eta_n.$$

Using (5.8), we obtain

$$\mathrm{P}\left[R(c_{\mathbf{z}}) - R(b) \leq \psi\left(\frac{C\log\frac{4}{\eta_n}}{\sqrt{n}}\right) \right] \geq 1 - \eta_n.$$

Observe now that

$$\lim_{n\to+\infty} \frac{C\log\frac{4}{\eta_n}}{\sqrt{n}} = 0,$$

since

$$\frac{\log\frac{4}{\eta_n}}{\sqrt{n}} = \lambda_n 2\sqrt{2}\kappa^2,$$

and $\lim_{n\to+\infty} \lambda_n = 0$ for hypothesis (5.5). It follows that, since ψ goes to zero at the origin,

$$\lim_{n\to+\infty} \psi\left(\frac{C\log\frac{4}{\eta_n}}{\sqrt{n}}\right) = 0.$$

Therefore, it exists a $n_\varepsilon > 0$ such that $\psi\left(\frac{C\log\frac{4}{\eta_n}}{\sqrt{n}}\right) \leq \varepsilon$ for $n \geq n_\varepsilon$. Putting all together we have

$$\begin{aligned} \mathrm{P}\left[R(c_{\mathbf{z}}) - R(b) > \varepsilon \right] &= 1 - \mathrm{P}\left[R(c_{\mathbf{z}}) - R(b) \leq \varepsilon) \right] \\ &\leq 1 - \mathrm{P}\left[R(c_{\mathbf{z}}) - R(b) \leq \psi\left(\frac{C\log\frac{4}{\eta_n}}{\sqrt{n}}\right) \right] \leq \eta_n \end{aligned}$$

and the thesis follows because $\lim_{n\to\infty} \eta_n = 0$. □

Part II

Applications

Chapter 6

Synthetic Data

In this chapter we present some empirical results using spectral regularization algorithms on sinthetic data and on a widely used dataset for multi-task learning.

In Section 6.1, we consider an academic example aimed at showing a computational comparison of the various methods while illustrating the differences between multi-task and vector valued learning.

In Section 6.2 and 6.3, we present some artificial examples of $2D$ vector fields for which our approach outperforms regressing on each component independently with the same filter function. Our simulations of $2D$ vector fields recall the flow of an incompressible fluid. A common practical problem in experimental physics is that of estimating a velocity field from scattered spatial measurements. Using kernel functions tailored for physical vector fields (see Section 2.3), we show how this problem can be effectively solved.

Finally, in Section 6.4, we present the results obtained with spectral filters on a dataset widely used to compare multi-task learning algorithms.

6.1 Vector Valued Regression vs Multi-task learning

We consider an academic situation where each task is given by the same function plus a task specific perturbation. More precisely, we study the case where the input space is the interval $[0, 1]$ and we have four tasks. Each task $t = 1, \dots, 4$, is given by a target function $f_t = f_{com} + \alpha f_{pert,t}$ corrupted by normal noise of variance 0.01. The target function common to all tasks is $f_{com} = sin(2\pi x)$. The weight α is set to be equal to 0.6. The perturbation function is generated with three gaussians of width $\sigma = 0.1$ centered at $x_1 = 0.1$, $x_2 = 0.4$ and $x_3 = 0.7$ and multiplied by coefficients that vary from one task to the other. We have designed these perturbations to yield tasks that are still strongly related by the common target function, but also present local differences, as shown in Figure 6.1. Despite from the plot, it may appear that the tasks are simply shifted versions of the common *sine* function, an approach based on computing the sample covariance to estimate the phase differences will not succeed in this case. Notwithstanding the simplicity of this example, we believe it is illustrative of the different behaviors of the multi-task and vector valued settings.

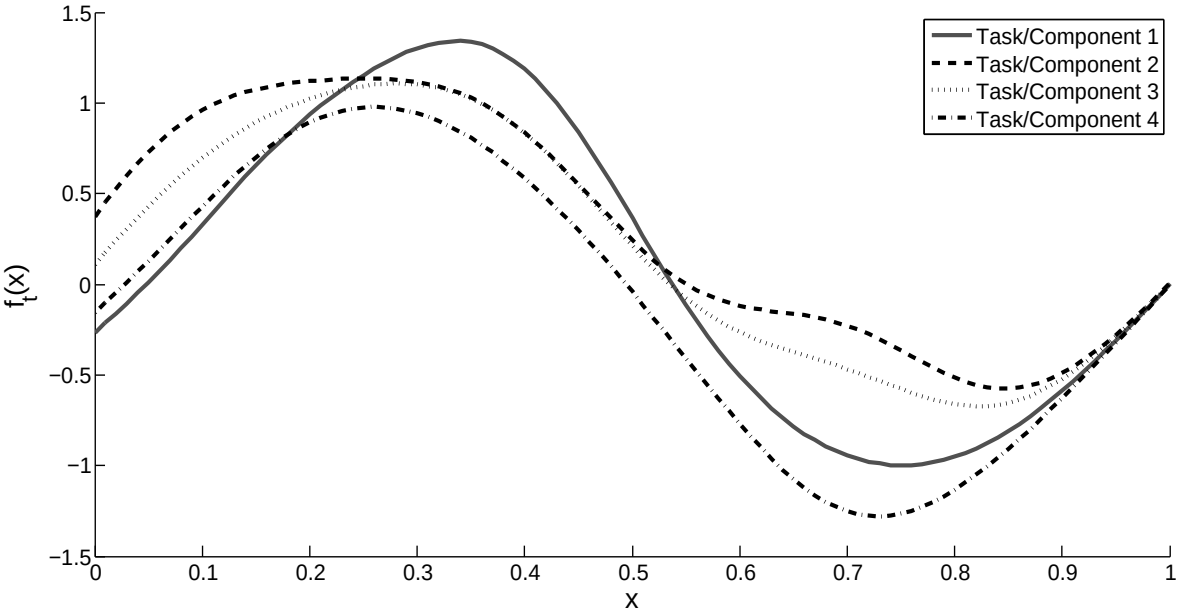


Figure 6.1: The four tasks/components, before being affected by gaussian noise of variance 0.01, used to compare multi-task and vector valued learning. The tasks are generated perturbing the common task (a sine function) with three gaussians of width 0.1, centered in $x_1 = 0.05$, $x_2 = 0.4$ and $x_3 = 0.7$. The gaussians are multiplied with task specific coefficients.

Since the performance of the spectral filters is very similar, we choose the ν-method to illustrate the typical behavior. In the multi-task case each task is uniformly randomly sampled in different input points, whereas in the vector valued case the input points are the same for all the components and still uniformly sampled in the interval $[0, 1]$. We use the kernel (2.8)

$$\Gamma_\omega(x, x') = K_\sigma(x, x')(\omega \mathbf{1} + (1 - \omega)\mathbf{I})$$

that imposes a common similarity among all components, adopting a gaussian kernel for its scalar part. The width of the gaussian kernel is a reference distance for assigning the similarity between two input points. As an heuristic, we use the mean distance from each point of the training set to its k nearest neighbors, where k is equal to 10% of the number of training points. This distance measure is an estimate of the average neighborhood width and it allows to consider the points within this neighborhood size as very similar, that is $K_\sigma(x, x') > 0.6$, if x, x' are two points within the neighborhood. The parameter ω and the regularization parameter were selected on a validation set of the same size of the training set. The performance of the algorithm is measured by the mean squared error (MSE) on an independent test set, as a function of the number of training set points available for each task/component. To evaluate the average performance and its variance, we run the simulation 10 times for each training and validation set size, resampling both sets. The performance obtained using matrix valued kernels, that exploit the relationship between the tasks or the components, is compared against the performance of using the same filter function to estimate each task or component independently with the a gaussian kernel of same width.

We show the results for multi-task learning case in Figure 6.2 (left). We observe that exploiting

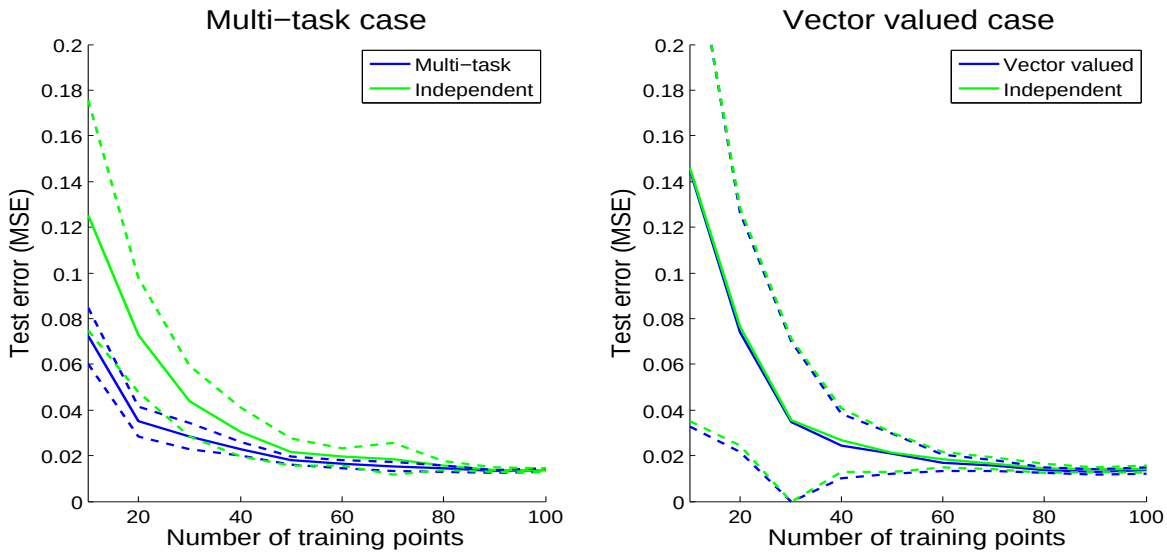


Figure 6.2: Results for the multi-task case (left) and for the vector valued case (right) using the ν-method with a maximum of 200 iterations. Solid lines represent average test error, while dotted lines show the average test error plus/minus one standard deviation. The test error is evaluated on an independent test set of 1000 examples as the mean squared error on all examples, counting the components of the vector valued function as different points. For the multi-task case, the training points are sampled independently for each task, whereas in the vector valued case the training points are the same for all the components. The experiments are run 10 times for each training set cardinality, with different sampling of the training and validation examples.

the coupling among the tasks is significantly advantageous. The median of the selected values for the kernel parameter ω is 0.6, indicating that the validation process selects an intermediate correlation between the tasks. The results for vector valued learning are given in Figure 6.2 (right), where we see that there is no gain in using a not-diagonal kernel.

We repeated the experiments with different filter functions in order to compare their computational efficiencies. In Figure 6.3 (left) we report the time required to select the optimal regularization parameter on the validation set in the multi-task case. The vector valued case presented the same behavior (plot not shown). The algorithms are Tikhonov regularization with 25 regularization parameter values, Landweber iteration with a maximum of 1000 iterations and ν-method with a maximum of 200 iterations. The number of parameters was chosen so that the validation error achieves the minimum within the range. As expected from the complexity consideration of Section 3.7, the ν-method is outperforming the other methods. In Figure 6.3 (right) we report the time required to select the optimal regularization parameter via Leave One Out Cross Validation (LOO) in the vector valued scenario. In this case it is possible to exploit the results of Section 3.5 and the closed form solution for the LOO error for Tikhonov regularization. Indeed, Tikhonov regularization combined with these two results is faster than the iterative algorithms, which require to evaluate the regularization path for each LOO loop. Note that this sort of computational advantage is not applicable if one is dealing with a multi-task problem.

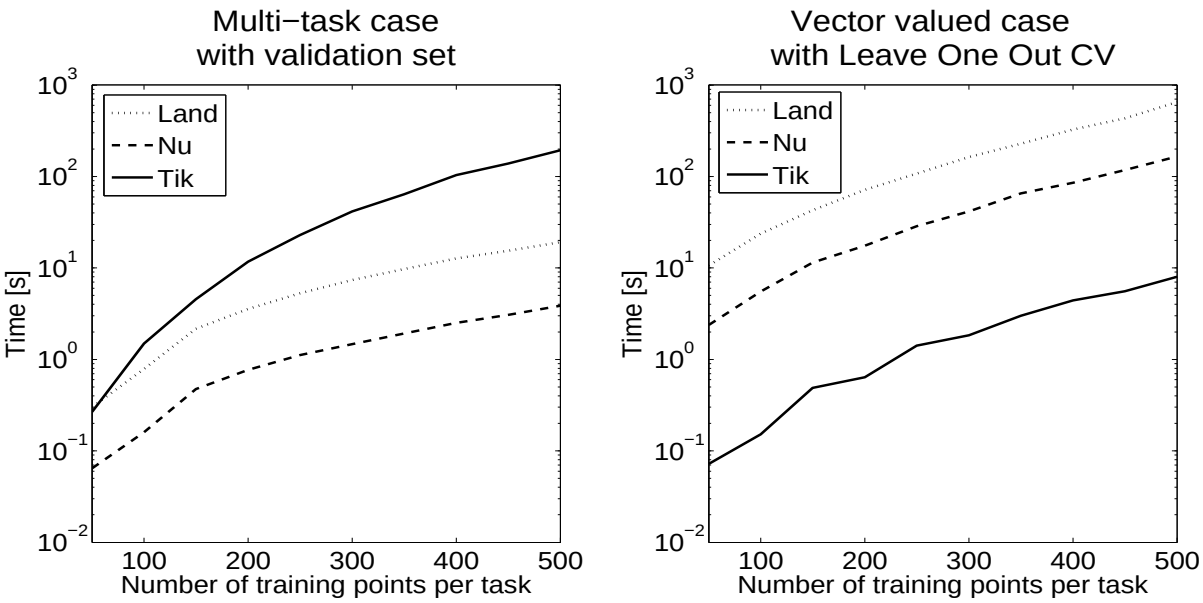


Figure 6.3: Time needed to select the best regularization parameter for different algorithms and settings. In the left panel are reported the times required to select the regularization parameter for the multi-task setting with respect to the number of training examples. The regularization parameter is chosen on a validation set of the same size of the training set. On the right are shown the times needed to select the regularization parameter via Leave One Out Cross Validation on the training set only. We implemented the optimization described in Section 3.5 and the closed form solution to compute the LOO errors for Tikhonov. The range of the parameters evaluated is 25 values for Tikhonov, a maximum of 1000 iterations for Landweber and 200 iterations for the ν-method. The computations were performed with MATLAB on a laptop with 2GB of RAM and a 2GHz Intel Core 2 Duo CPU.

6.2 2D Vector Field composed of a divergence-free part and a curl-free part

The following set of simulations are aimed at showing the advantages of using the divergence and curl-free kernels, (2.16) and (2.17) respectively, for the estimation of a 2-dimensional vector field defined on a 2-dimensional input space. By adopting a convex combination of the two kernels, weighted by a parameter $\tilde{\gamma}$, it is possible to reconstruct the divergence-free and curl-free parts of the field (Macêdo and Castro, 2008).

The vector field is generated from a scalar field defined by the sum of 5 gaussians centered at $(0, 0)$, $(1, 0)$, $(0, 1)$, $(-1, 0)$ and $(0, -1)$ respectively. The covariances are all set to be diagonal, namely $0.45\mathbf{I}$, where $\mathbf{I}$ is the 2×2 identity matrix. We compute the gradient of the scalar field and the field perpendicular to it, applying a $\pi/2$ rotation. We consider a convex combination of these two vector fields, controlled by a parameter γ. One instance of the resulting field for $\gamma = 0.5$ is shown in Figure 6.4, while examples for $\gamma = 0, 0.3, 0.6, 1$ are shown in Figure 6.5.

We compare our vector valued regression approach with estimating each component of the field independently. We use the ν-method, which is the fastest algorithm when the matrix valued kernel is not of the form $\Gamma = KA$. We adopt a 5-fold cross validation to select the optimal number of iterations and the parameter $\tilde{\gamma}$. The scalar kernel is a gaussian kernel of width 0.8.

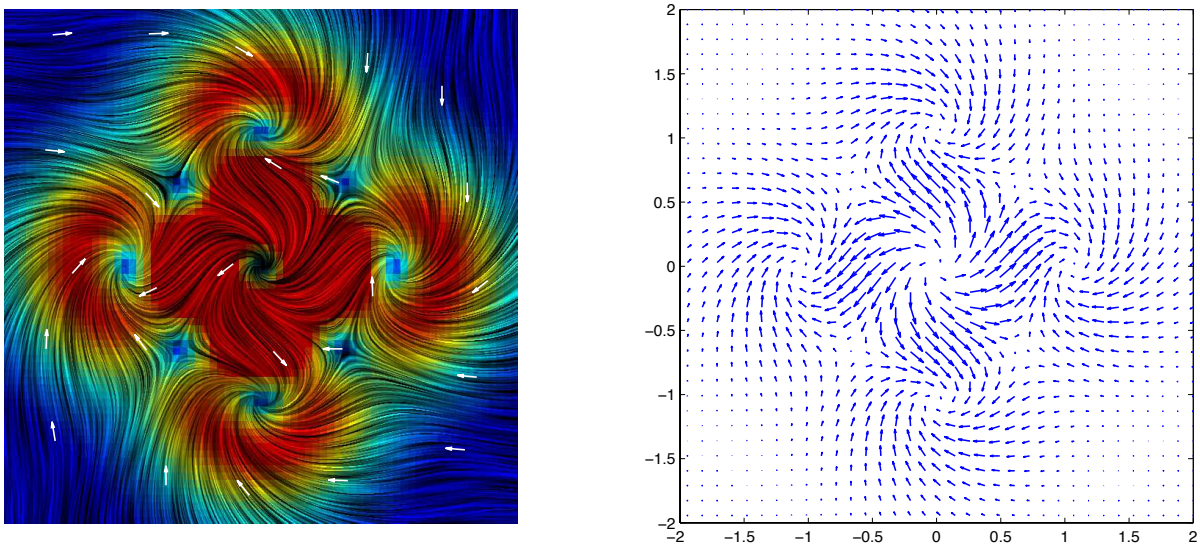


Figure 6.4: Visualization of the first artificial 2-dimensional vector field for $\gamma = 0.5$

Firstly, we consider the noiseless case. The vector field is constructed specifying a value of the parameter γ, which we vary from 0 to 1 at 0.1 increments. The field is then computed on a 70×70 points grid over the square $[-2, 2] \times [-2, 2]$. The models are trained on a uniform random sample of points from this grid and their predictions on the whole grid (except the training points) compared to the correct field. The number of training examples is varied from 10 to 1000 at 10 examples increments until 100 examples, after that the increments become larger for computational economy. For each cardinality of the training set, the training and prediction process is repeated 10 times with a different randomization of the training points.

Following (Barron et al., 1994), we use an angular measure of error to compare two fields. If $v_o = (v_o^1, v_o^2)$ and $v_e = (v_e^1, v_e^2)$ are the original and estimated fields, we consider the transformation $v \to \tilde{v} = \frac{1}{||(v^1, v^2, 1)||}(v^1, v^2, 1)$. The error measure is then

$$err = arccos(\tilde{v}_e \cdot \tilde{v}_o). \tag{6.1}$$

This error measure was derived by interpreting the vector field as a velocity field and it is convenient because it handles large and small signals without the amplification inherent in a relative measure of vector differences.

The results for the noiseless case are reported in Figure 6.6, which clearly shows the advantage of using a vector valued approach with the combination of curl-free and divergence-free kernels. We present only the results for the field generated with $\gamma = 0$ and $\gamma = 0.5$ since for the remaining fields the errors are set within these two examples. The prediction errors of the proposed approach via the ν-method are always lower than the errors obtained by regressing on each component independently, even when the training set is very large. The average value of the estimated parameter $\tilde{\gamma}$, shown in Figure 6.7, converges to the true value of γ as the number of training points increases, indicating that it is possible for the model to learn the field decomposition in an automatic way.

We then consider the case with normal noise whose standard deviation is independent from the signal and is chosen to be 0.3. We follow the same experimental protocol adopted for the noiseless

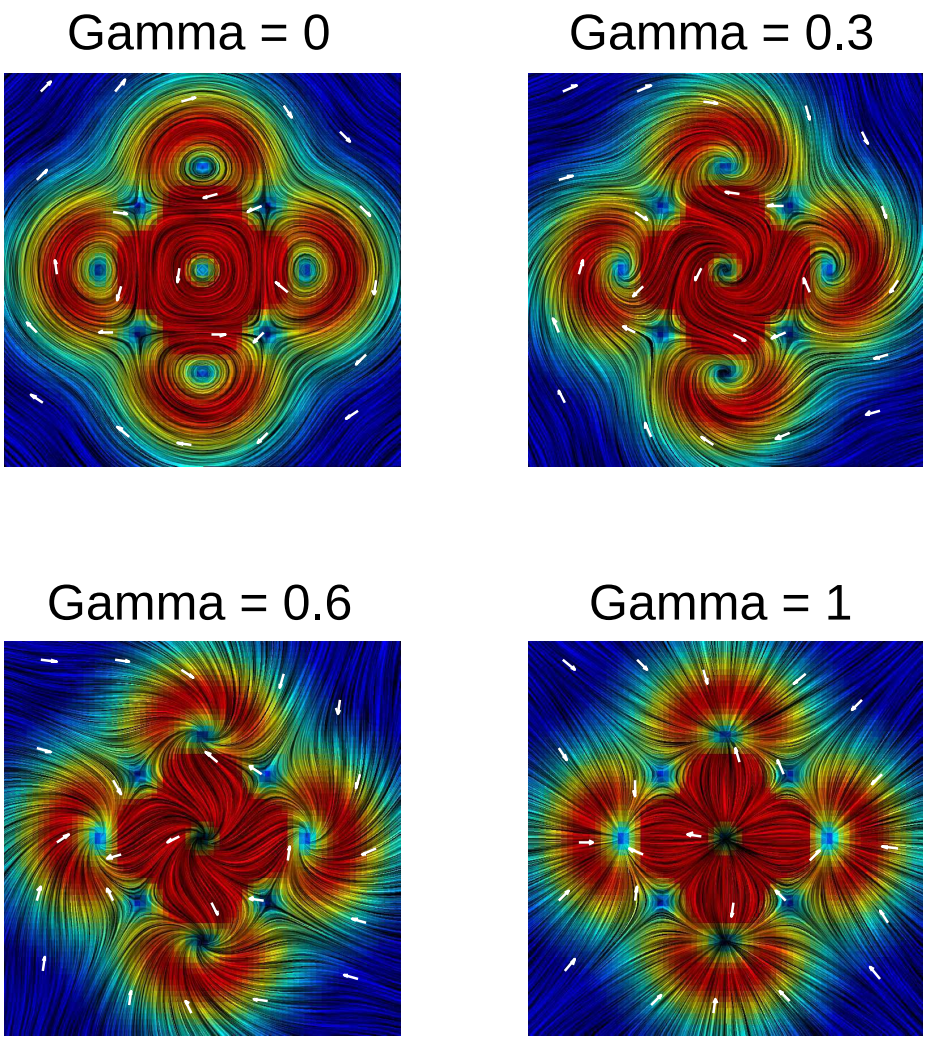


Figure 6.5: Visualization of the first artificial 2-dimensional vector field for several values of the parameter γ that balances the divergence-free and the curl-free parts.

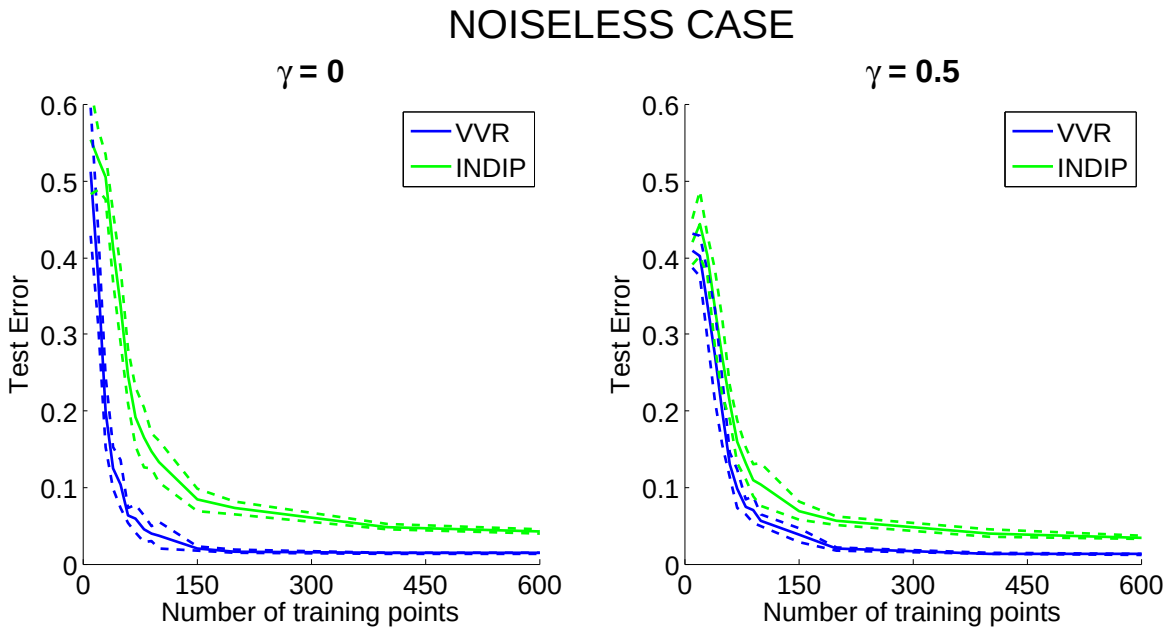


Figure 6.6: Vector field 1 - noiseless case. Test errors for the proposed vector valued approach and for learning each component of the field independently as a function of the number of training points used for learning. Solid lines represent average test error, while dotted lines show the average test error plus/minus one standard deviation of the corresponding error. The test error is evaluated according to the error measure (6.1).

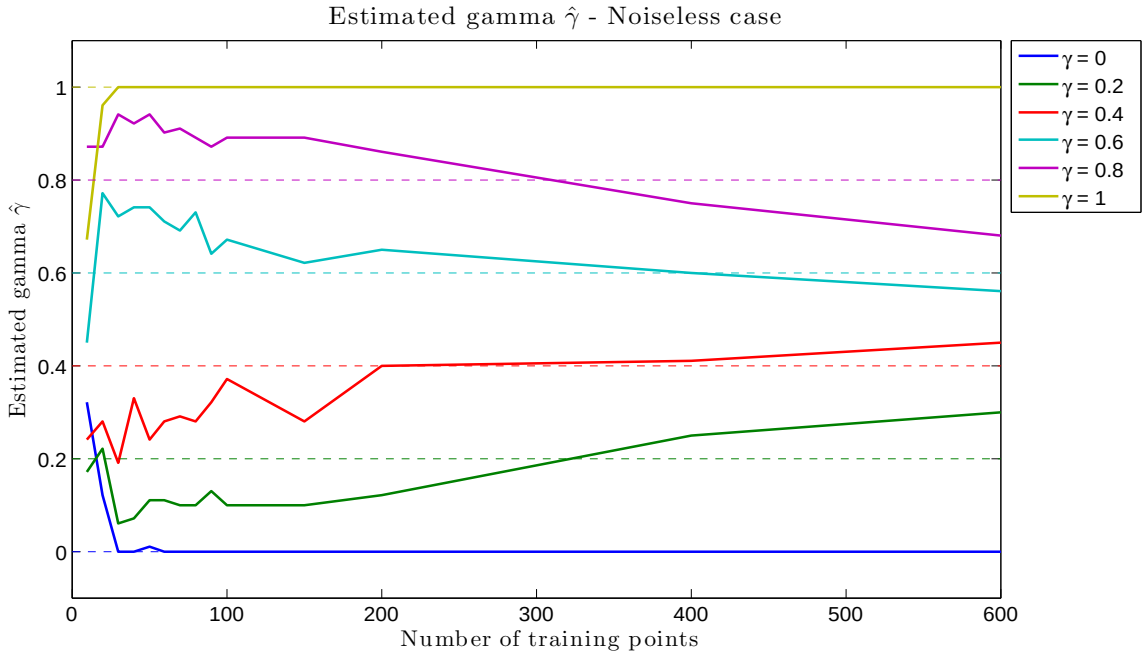


Figure 6.7: Vector field 1 - Noiseless case. Estimated values for the parameter $\hat{\gamma}$ balancing the divergence-free and the curl-free kernels. As the training sets become larger the estimated value better approximates the true value γ. Note, however, the tendency to stray towards the value 0.5 for fields that are not completely divergence-free or curl-free.

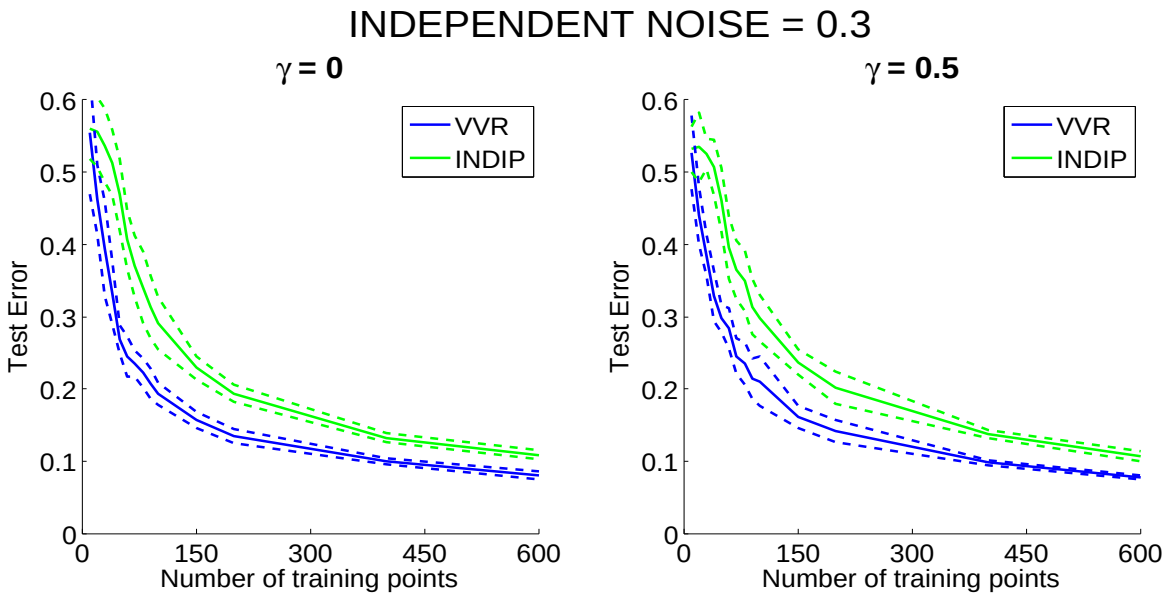


Figure 6.8: Vector field 1 corrupted by independent noise of standard deviation 0.3. Test errors for the proposed vector valued approach and for learning each component of the field independently as a function of the number of training points used for learning. Solid lines represent average test error, while dotted lines show the average test error plus/minus one standard deviation of the corresponding error. The test error is evaluated according to the error measure (6.1).

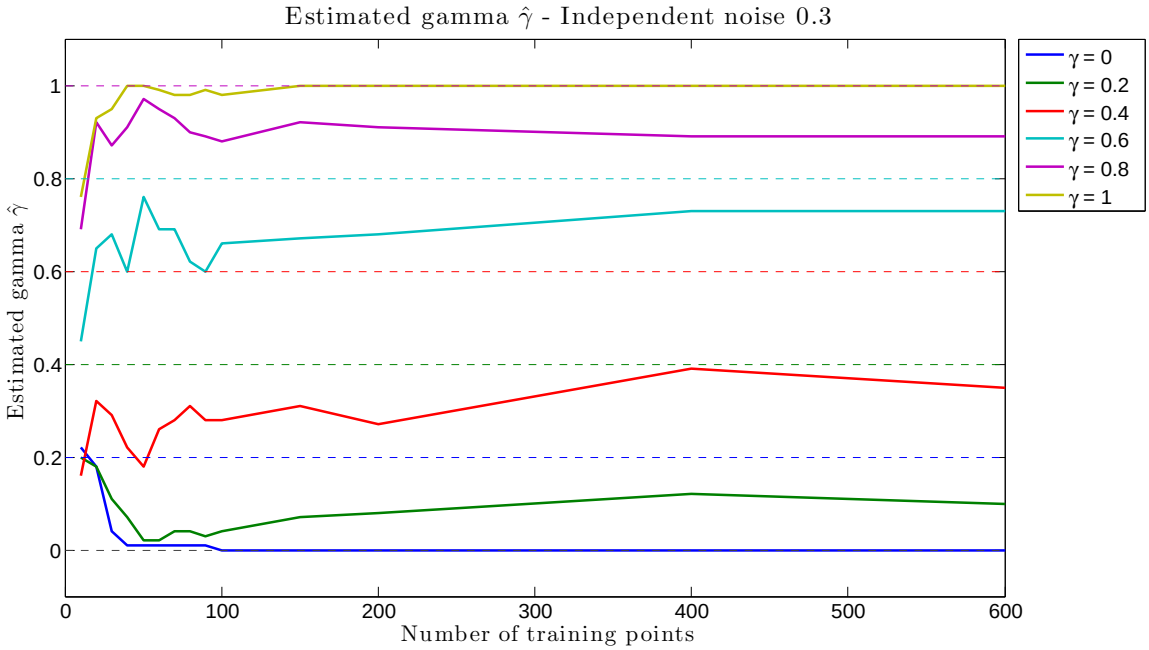


Figure 6.9: Vector field 1 - Independent noise of width 0.3. Estimated values for the parameter $\hat{\gamma}$ balancing the divergence-free and the curl-free kernels. As the training sets become larger the estimated value better approximates the true value γ.

case. The results are reported in Figure 6.8 and indicate that also in the presence of noise the proposed approach consistently outperforms regressing on each component independently. The advantage is stronger when fewer training points are available, but it is still present even at higher training set cardinalities. Again, the estimated value for the parameter $\hat{\gamma}$, that weighs the contributions of the two kernels, well approximates the true value used for the creation of the vector field, indicating that it is possible for the model to learn the field decomposition in an automatic way also in presence of noise (see Figure 6.9).

It is now interesting to apply this approach to a vector field that is not directly given as the sum of a divergence-free and a curl-free part, but that satisfies the hypotheses of the Helmholtz Theorem on the decomposition of a vector field (Arfken and Weber, 2005).

6.3 Generic 2D Vector Field

The Helmholtz Theorem states that a vector field, which is twice continuously differentiable and which vanishes faster than $1/r$ at infinity (r is the distance from the origin), can be decomposed as the sum of a gradient-free part and a curl-free part (Arfken and Weber, 2005). Therefore, if we are dealing with such a vector field, we expect to be able to estimate it via a combination of the divergence-free and curl-free kernels. This second artificial experiment aims at showing that, when the vector field satisfies the assumptions of the Helmholtz Theorem, it is indeed possible to obtain a better estimate using these kernels than treating each component independently.

On a grid of 70×70 points within $[-2, 2] \times [-2, 2]$, we generate a vector field whose components are given by

$$\begin{aligned} v^1(x, y) &= 2sin(3x)sin(1.5y) \\ v^2(x, y) &= 2cos(3y)cos(1.5x) \end{aligned}$$

In order to enforce the decay at infinity the field is multiplied to a gaussian function centered at the origin and of width 1.2. The field without noise is shown in Figure 6.10.

We follow the same experimental protocol adopted for the previous artificial experiments. In this case there is no field parameter to vary, but only the amount of noise which we consider proportional to the signal. This means that for each point of the field, the standard deviation of the noise added to the field in that point is proportional to the magnitude of the field. The results for the noiseless and noisy cases are shown in Figure 6.11. We can still see an advantage in using the proposed vector valued approach with a combination of the divergence-free and curl-free kernels, but the gain drops to insignificant as the number of training examples increases. However, in Figure 6.12, relative to the noisy case, we note how the estimated weight of the combination of the two kernels is stable and offers a clue on the nature of the vector field. It is then possible to reconstruct the decomposition of the field into a divergence-free and a curl-free part, as shown in Figure 6.13.

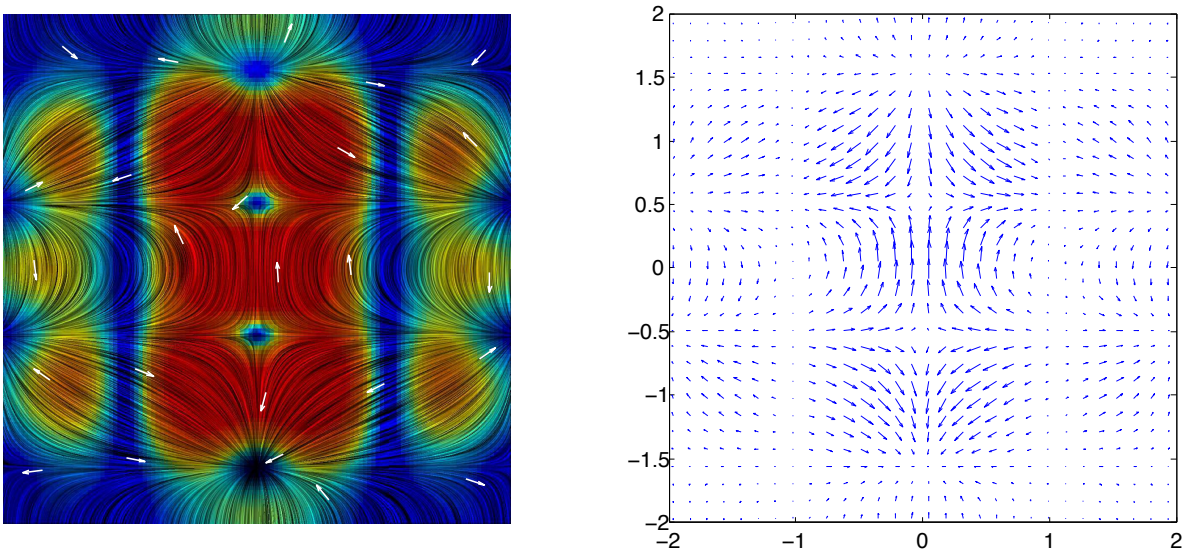


Figure 6.10: Visualization of the second artificial vector field without noise.

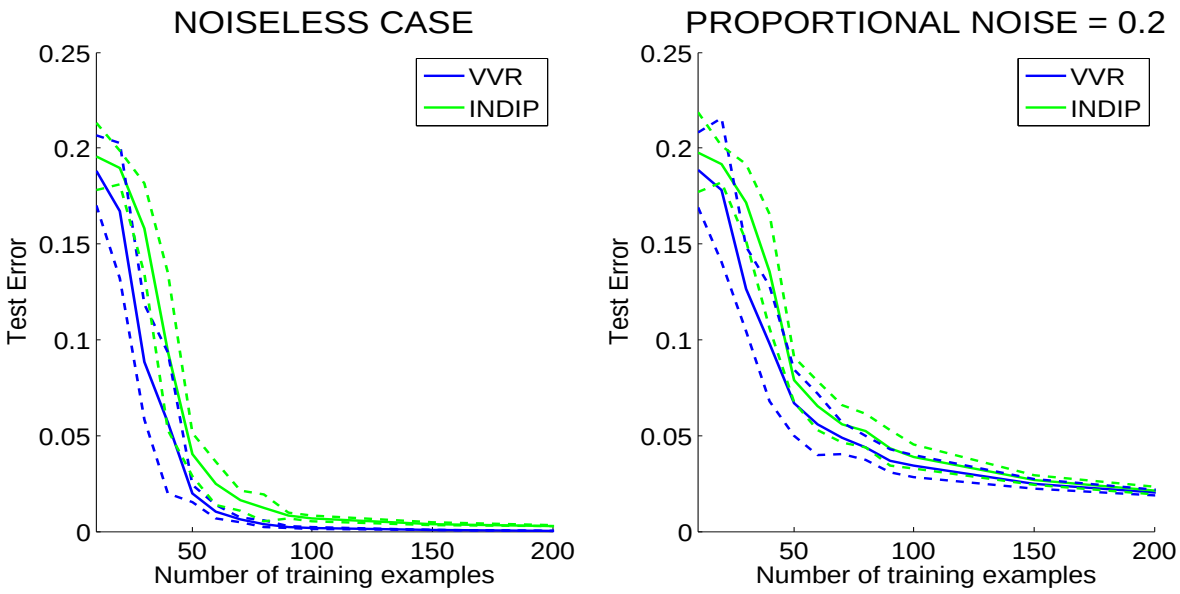


Figure 6.11: Vector field 2. Test errors for the proposed vector valued approach and for learning each component of the field independently in the noiseless case (left) and when the standard deviation of the noise is equal to 20% of the field magnitude (right). The test error is evaluated according to the error measure (6.1) and is plotted against the number of training examples used for learning.

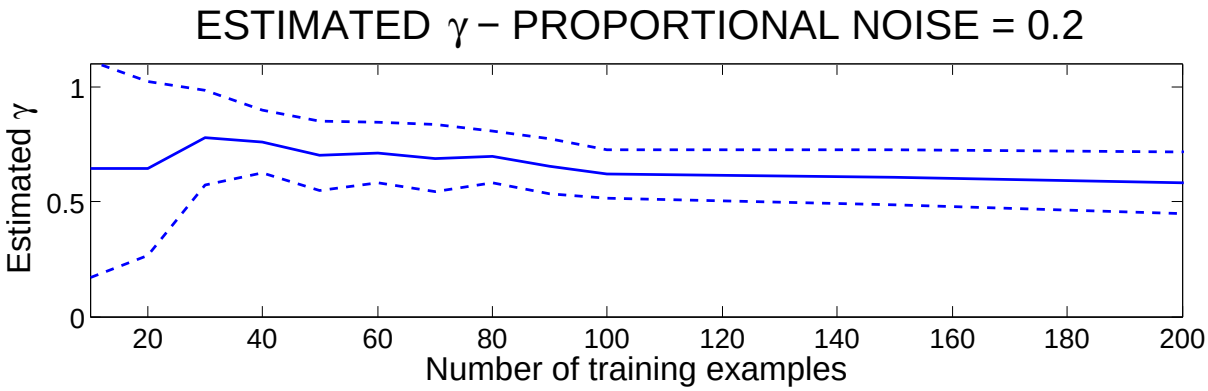


Figure 6.12: Vector field 2 with proportional noise (noise standard deviation equal to 20% of field magnitude.) Estimated value for the parameter γ that balances the divergence-free and the curl-free kernels as a function of the number of training points used for learning.

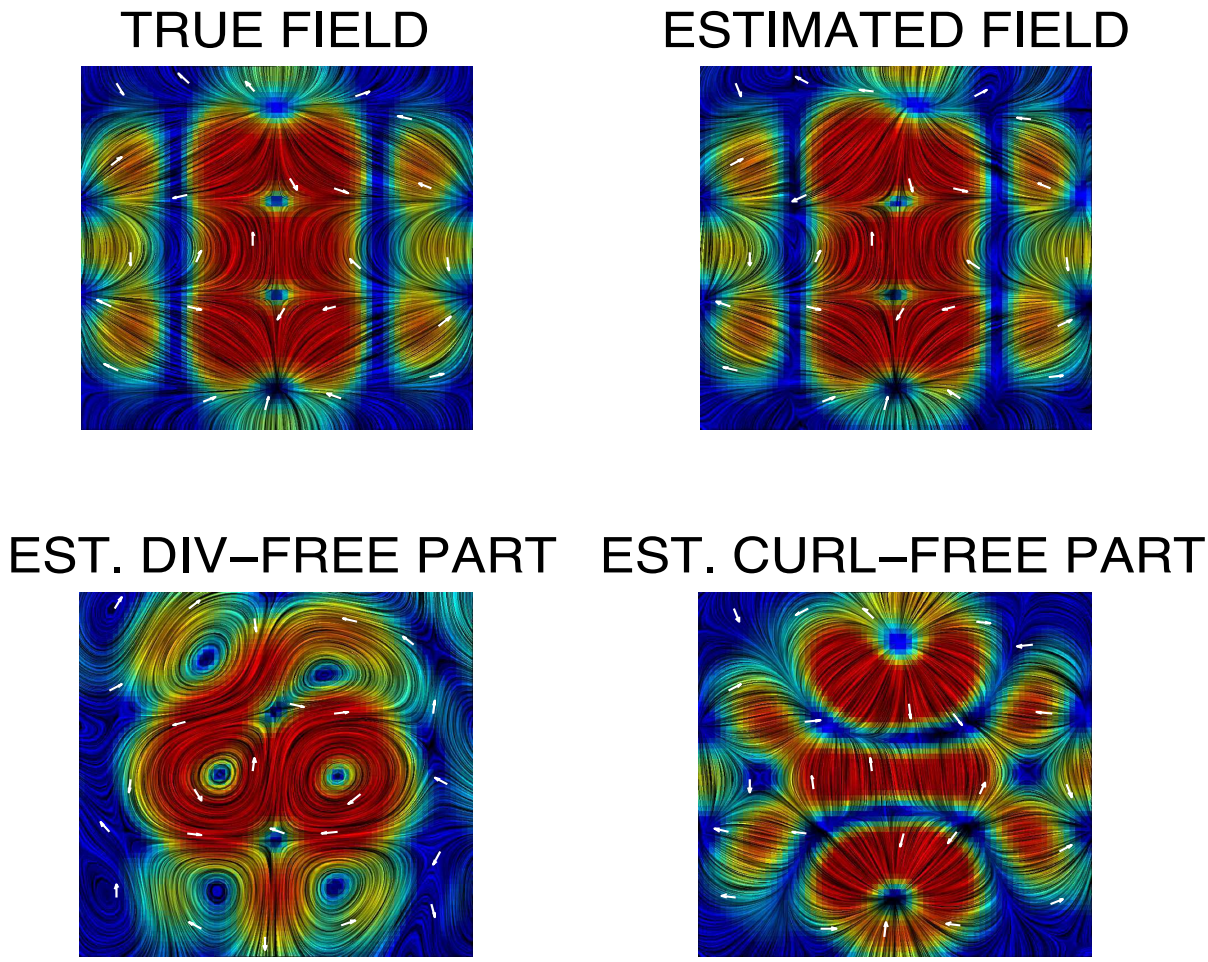


Figure 6.13: (Top-left) Original vector field, (top-right) learned field from a noisy sample of 100 examples affected by proportional noise of width 0.2. (Bottom row) Decomposition of the learned field into a divergence-free part (left) and a curl-free part (right).

6.4 School data

This dataset from the Inner London Education Authority[1] has been used in previous works on multi-task learning (Bakker and Heskes, 2003; Evgeniou et al., 2005; Argyriou et al., 2008a) and has become a standard benchmark over the recent years. It consists of the examination scores of 15362 students from 139 secondary schools in London during the years 1985, 1986 and 1987. One can define 139 tasks or regression problems, corresponding to predicting student performance in each school. The input data for each student consist of school attributes and personal attributes. The school attributes are: percentage of students eligible for free school meals, percentage of students in VR band one (highest band in a verbal reasoning test), school gender (male, female or mixed) and school denomination. Student specific attributes are: gender, VR band (can take the values 1, 2 or 3) and ethnic group (among 13 possible values). Following the literature, we converted the categorical attributes using one binary variable for each possible attribute value, but we only considered student specific attributes. Each student is thus characterized by a feature vector of 19 bits. The school attributes could be used to define a similarity score between the schools, which we reserve to possible future work. For now, we are only interested in comparing the results and computation times of the Landweber, ν-method and Tikhonov algorithm using the simple common similarity kernel (2.8)

$$\Gamma_\omega(x, x') = K(x, x')(\omega \mathbf{1} + (1 - \omega)\mathbf{I}).$$

We randomly selected 60% of students from each school and divided their data equally into three sets: training, validation and test. Each set has 3124 students and, on average, 22 students per school. The validation set is used to select the regularization parameter and the value of the parameter ω that forces the tasks to be more or less related (a higher value of ω corresponds to higher similarity among the tasks). On the test set we evaluated the generalizing performance of the three algorithms using the measure of *explained variance* from (Bakker and Heskes, 2003). Explained variance is defined as one minus the mean squared test error over the total variance of the data (across all tasks). We opted for a gaussian scalar kernel whose width was chosen to be the mean distance from each training point to its k nearest neighbors, where k is set to be 20% of the cardinality of the training set.

In Table 6.4 we report the test performance and the time needed to select the optimal parameters on the validation set (without taking into account the time needed to compute the kernel matrices, because these are common to all algorithms). The range of the parameter omega is $[0, 1]$ and was sampled at 0.1 intervals. The three algorithms, despite being trained only on 20% of the available data, plus an additional 20% for parameter selection, perform consistently and similarly to the reported results in (Argyriou et al., 2008a), which obtains a performance of $26.4\% \pm 1.9\%$. The results in (Argyriou et al., 2008a) were achieved using 75% of the data for training and adopting a 15-fold cross validation to select the regularizing parameter. In the previous works no computation time is reported, while from our results the ν-method is at least three orders of magnitude faster than Tikhonov and more than one order of magnitude faster than Landweber. Obviously the validation time depends on the number of iterations or the number of values of the regularizing parameter to evaluate. For Landweber, after a first preliminary assessment, we opted for a maximum of 3000 iterations while for the ν-method a

[1] Available at http://www.mlwin.com/intro/datasets.html

maximum of only 150 iterations. For Tikhonov, we choose 30 values sampled geometrically in the interval $[10^{-4}, 10^{-2}]$. Furthermore for Tikhonov there is also the double issue of choosing the range and the grain, since in some occasions (as we have experienced) a good value might reside just between the values we are testing. Conversely, iterative methods could seen as rough, not allowing a finer exploration of parameter space, but this is not the case, since one can choose a smaller step size.

Algorithm	**Performance**	**Model Selection Time** [10^4 sec]
ν-method	26%	13
Landweber	22%	259
Tikhonov	25%	2026

Table 6.1: Performance as measured by the explained variance and model selection time for the Landweber, ν-method and Tikhonov algorithms on the School dataset. The multi-task feature learning method proposed in (Argyriou et al., 2008a) obtains a performance of $26.4\% \pm 1.9\%$. The computations were performed on a desktop computer with 4GB RAM and 2.2Ghz dual-core processor

6.5 Considerations

From these preliminary experiments, it is evident that matrix valued kernels that exploit the relationship among the components of the vector valued function (or the tasks in a multi-task scenario), combined with an efficient spectral filter, such as the ν-method, provide a competitive alternative to simpler regression schemes that treat each component independently. Even when the prediction accuracy is not much improved (e.g. in the second vector field example), the computational speed-up provided by iterative spectral filters and the decomposition scheme for matrix valued kernels of the form $\Gamma = KA$, can be seen as essential steps towards designing more appropriate multi-output kernels, since they allow to compare different kernels in a shorter time.

Chapter 7

Magnetic Iron Detector

In this chapter we present the application of the proposed vector valued framework for the estimation of a signal that allows to assess the iron overload in the liver of patients affected by Thalassemia or Hereditary Hemochromatosis. After introducing the measurement challenge and discussing a previous model to estimate the iron overload, in Section 7.3 we present our approach that is based on, first, designing a proper feature map to exploit the reasonable correlation we expect from the measures, and then computing the corresponding matrix valued kernel in order to apply the spectral filters. The experimental protocol outlined in Section 7.3.2 is applied and results are discussed in Section 7.4.

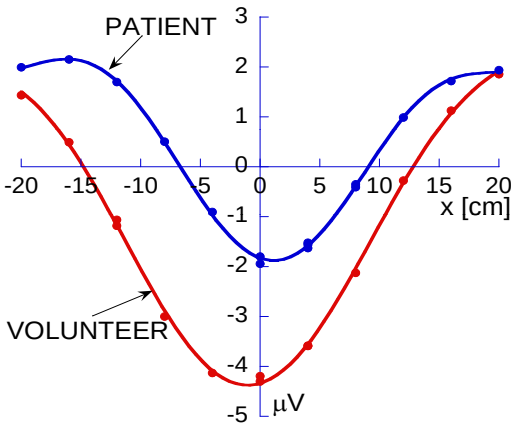
7.1 Introduction

Iron (Fe) is essential to human life, but is toxic in excessive amounts. There are several diseases characterized by liver iron overload, such as Thalassemia or Hereditary Hemochromatosis. The accurate assessment of body iron excess is therefore essential for managing the therapies for these diseases. Each disease is characterized by a different mechanism of iron accumulation. For thalassemia patients iron overload is induced by periodical blood transfusions, while for hemochromatosis patients it is induced by an incorrect dietary absorption of iron.

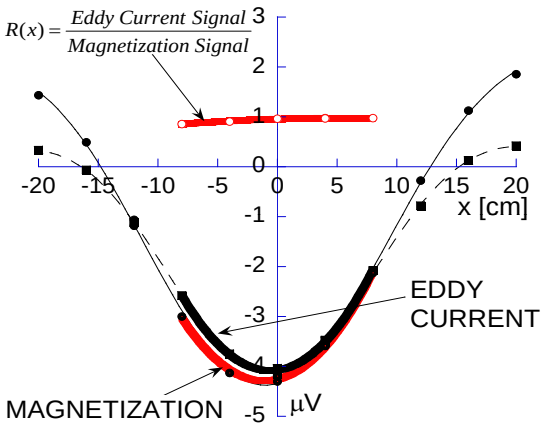
The invasive liver biopsy is still considered the best way to perform iron overload evaluation, but being a local measure, it is affected by large errors due to the heterogeneous distribution of iron deposition in the liver. Moreover, not many patients allow frequent biopsies to assess the progression of the therapy.

Recently, Marinelli and colleagues (Marinelli et al., 2007) have developed a room-temperature biosusceptometer, the Magnetic Iron Detector (MID), which measures the variation of a magnetic field at different positions along the axis that crosses the patient's liver. This instrument allows the non-invasive assessment of the iron overload in the whole liver. Given an estimate of the background signal of a patient, that is the signal that would be generated in absence of iron overload, it is possible to recover the iron burden by comparing the estimated signal from the measured signal.

In order to estimate the background signal, Marinelli and coworkers (Marinelli et al., 2006) have



7.1.1: Volunteer and patient magnetization signals.



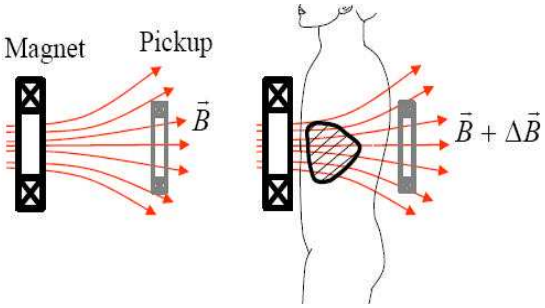
7.1.2: Eddy current and magnetization signals of a volunteer.

developed a statistical parametric model that is currently used at the "E.O. Ospedali Galliera" Hospital in Genoa, Italy. As we will better explain in the following, the measurement generates two signals, shown in Figure 7.1.2, of which only the magnetization signal depends on the liver iron content. To reduce the common dependencies on the geometry of the body of the patient, the model estimates the ratio $R(x)$ between the two signals and has been trained on two datasets of 84 and 142 healthy volunteers, respectively. The core idea behind their approach is that the magnetization signal of a well-treated patient is indistinguishable from the one generated by a healthy volunteer with the same biometric features, see Figure 7.1.1. Furthermore, they assume that the ratio $R(x)$ of the two signals, in the range between -8cm and 8cm, which include the greatest contributions from the liver, resembles a parabola.

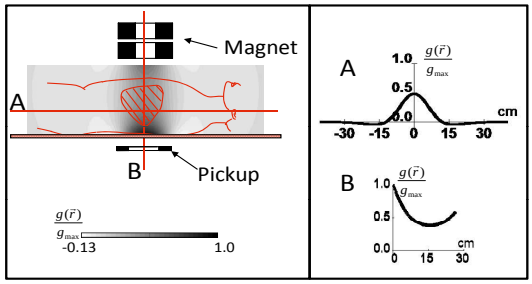
We reformulate this problem in the context of supervised learning presenting a method to transform a curve fitting task into a vector valued regression model. Since the measures are always taken at fixed positions along the measurement axis, they can be thought of as components of a vector and a high correlation among them can be assumed, since they lie on a parabola with an approximation error smaller than the actual prediction error of the parametric model. In this way we eschew from directly estimating the magnetization curve. Our vector valued regression model simultaneously estimates the five measures of the background signal. The correlation between these points is introduced by an appropriate kernel, which is non-linear in the biometric features and quadratic in the measurement positions.

7.2 Liver Iron Overload Estimation

The biosusceptometer is composed of an AC magnetic source and a pickup coil which measures the electromotive force (*emf*) produced by the oscillation of the magnetic field flux, as shown in Figure 7.1.3. A body placed between the magnet and the pickup slightly modifies the flux and therefore the *emf* measured: the amount of the variation depends on the magnetic properties of the body, on its geometry and on its positioning in the field. The *emf* produced in the pickup by the field is about $4V$ and the diamagnetic signal of a body is about $-4\mu V$. A moderate



7.1.3: Magnet and pickup configuration.



7.1.4: Weight function $g(\vec{r})$ and its cross section in the direction A and B.

Figure 7.1: When the patient is positioned between the magnet and the pickup a *emf* is produced by the magnetization on the tissues of the patient. With this positioning, the whole liver falls in the region where the function $g(\vec{r})$ is greater.

iron overload adds a paramagnetic contribution of about $0.4\mu V$. The symmetry of the system and the use of synchronous detection, via two pickups, make this difficult measurement feasible (Marinelli et al., 2006).

A sample of susceptibility χ, placed between the magnet and one of the pickups, generates the signal

$$V = \int_{Volume} \chi(\vec{r}) g(\vec{r}) d\vec{r} \tag{7.1}$$

The weight function $g(\vec{r})$ is reported in Figure 7.1.4: all body tissues contribute to the signal generation, but the major contribution comes from those between the magnet and the pick up coil (measurement region).

The patient lies supine on a stretcher, such that in the measurement region the liver center of mass crosses the magnetic field axis. The magnetic track of the patient is the complete scan of the magnetic properties of the body section. It is composed of the measurements of the magnetic signal 4cm apart. Position x = 0cm corresponds to the center of the body; negative positions indicate the liver side, while positive positions the spleen side. Figure 7.1.1 reports the magnetic tracks of a healthy volunteer and of a patient with similar anthropometric features: the liver iron overload produces an evident variation of the signal in the left part of the track.

Due to the oscillating magnetic field, the magnetic signal generated by the human body has two independent sources: the magnetization signal, from the diamagnetic and paramagnetic properties of the tissues, and the eddy current signal, from their electrical conductivity. For each patient a double track is recorded (an example is shown in Figure 7.1.2), but only the magnetization signal depends on the iron overload.

The aim of our research is to find a model which best approximates the magnetization signal of a healthy volunteer (background signal) from its anthropometric data and the eddy current signal. The iron overload can be evaluated by computing the difference between the measured magnetization signal and the estimate of the background signal. Therefore, by increasing the prediction accuracy of the model, we increase our ability to detect slight iron overloads. However, the maximum accuracy we can obtain is limited by the measurement error, mainly due to the positioning of the patient on the stretcher. This error is about $150nV$ and corresponds to an iron overload of $\sim 0.4g$.

7.3 Supervised Learning Approach

7.3.1 Designing the feature map

For each volunteer $i = 1, \ldots, 84$ or 142, we consider only the 5 measures y_{ik} at positions $t_k = \{-8, -4, 0, 4, 8\} cm$. The measures can be thought as the components of a five-dimensional vector and approximately lie on a parabola, hence we can model them as $y_{ik} = f(\mathbf{x_i})^k + \epsilon_{ik}$, where $\mathbf{x_i} \in \mathbb{R}^p$ is the biometric data of the volunteer i as measured by p features, ϵ_{ik} represents the noise and

$$f(\mathbf{x_i})^k = c_0(\mathbf{x_i}) + c_1(\mathbf{x_i})t_k + c_2(\mathbf{x_i})t_k^2. \tag{7.2}$$

If we assume that the coefficients c_j depend linearly on $\mathbf{x}$

$$c_j(\mathbf{x}) = \boldsymbol{\beta}_j \cdot \mathbf{x}, \qquad j = 0, 1, 2$$

we can introduce the vector valued feature map, $\varphi : \mathbb{R}^p \to \mathbb{R}^{5\times 3p}$

$$\varphi(\mathbf{x}) = \left(\begin{array}{c|c|c} \mathbf{x} & \mathbf{x}t_1 & \mathbf{x}t_1^2 \\ \mathbf{x} & \mathbf{x}t_2 & \mathbf{x}t_2^2 \\ \mathbf{x} & \mathbf{x}t_3 & \mathbf{x}t_3^2 \\ \mathbf{x} & \mathbf{x}t_4 & \mathbf{x}t_4^2 \\ \mathbf{x} & \mathbf{x}t_5 & \mathbf{x}t_5^2 \end{array}\right). \tag{7.3}$$

Let us define $\boldsymbol{\beta}$ as the vector obtained by concatenating the coefficient vectors $\boldsymbol{\beta}_j$.

The vector valued estimator can be rewritten as a linear combination of these new features

$$\mathbf{f}(\mathbf{x}) = \varphi(\mathbf{x})\boldsymbol{\beta}, \qquad \boldsymbol{\beta} \in \mathbb{R}^{3p}, \tag{7.4}$$

and the corresponding empirical risk is:

$$\mathcal{E}_n(\boldsymbol{\beta}) = \frac{1}{n}\sum_{i=1}^{n} \|\mathbf{y}_i - \varphi(\mathbf{x}_i)\boldsymbol{\beta}\|^2_{\mathbb{R}^5}.$$

From the vector valued feature map φ we can calculate the corresponding matrix valued kernel. Following (Caponnetto et al., 2008)

$$\begin{aligned} (\Gamma(x, x'))_{pq} &= \sum_{\ell=1}^{3p} \varphi_{p\ell}(x)\varphi_{q\ell}(x') \\ &= (x \cdot x')(1 + t_p t_q + t_p^2 t_q^2). \end{aligned}$$

We can this matrix valued kernel to compute the Gram matrix and apply the spectral filters as discussed in Chapter 3. The kernel is of the form $\Gamma(x, x') = K(x, x')A$, with $K(x, x') = x \cdot x'$ and $A_{pq} = 1 + t_p t_q + t_p^2 t_q^2$. Thus, we can replace the scalar linear kernel K, with other non-linear scalar kernels, without having to go through the design of the corresponding feature maps.

Our aim is to compare the performance of the MID parametric model versus the vector valued model estimated via the ν-method and a scalar model trained for each measurement position independently, again with the ν-method.

Feature	Description
1	Eddy current at -12 cm
2	Eddy current at -8 cm
3	Eddy current at -4 cm
4	Eddy current at 0 cm
5	Eddy current at 4 cm
6	Eddy current at 8 cm
7	Eddy current at 12 cm
8	Thorax section area at 0 cm
9	Thorax section area at 18 cm
10	Thorax section area at -18 cm
11	Thorax height at 0 cm
12	Thorax height at 18 cm
13	Thorax height at -18 cm
14	Adam's apple position
15	Navel position
16	Age
17	Height
18	Weight
19	Thorax circumference
20	Circumference under ribs arch
21	BMI (Body Mass Index)
22	Body area

Table 7.1: Anthropometric features used

7.3.2 Model selection and assessment

We adopt an experimental protocol in order to select the model parameters and assess the generalization capabilities of our method in an unbiased way. We perform two nested loops of K-fold Cross Validation. Higher values of K reduce the bias of the estimator, since the model is trained on more data, but increase its variance, since fewer data are used for testing. On the other hand, more splits imply more computations, which require more time.

The inner loop is a 5-fold Cross Validation and is performed to select the number of iterations and the width of the gaussian kernel chosen for the scalar part of the matrix valued kernel. For each value of the parameters, an estimate of the generalization error is computed. The values that minimize the error are used for training. The outer loop is a LOO Cross Validation evaluating the performance of the models. The estimate of the generalization error is the mean of the $K = n$ empirical errors.

7.4 Results

The first data set is composed of 84 healthy volunteers represented by vectors of features which are reported in Table 7.4. The second data set is composed of 142 volunteers whose features are the ones reported in Table 7.4 with the addition of the estimates of the magnetic track obtained considering the shape of the volunteer as filled with water. Note that from now on we will refer to n as the number of examples in the training set within the innermost loop of CV.

The features are highly inhomogeneous and can lead to numerical problems, therefore we decided to normalize our data. We set the columns of the $n \times p$ data matrix $X = (\mathbf{x}_1, \ldots, \mathbf{x}_n)^T$, to have zero mean and fixed range and changed the variable t from $\{-8, -4, 0, 4, 8\}$ to $\{-1, -0.5, 0, 0.5, 1\}$, since it only represents a label for the components of the vector $\mathbf{y}$. In the test phase, we apply the normalizing factors computed on the training set to the test data to avoid biased results, that could easily be obtained if we normalized the entire dataset at once.

For model selection and assessment we used the experimental protocol outlined in Section 7.3.2. The model parameters to be selected are the width of the scalar gaussian kernel, among 16 values, and the number of iterations for the ν-method, setting the maximum to 150.

We report the values of parameters selected during the cross validation in Table 7.2. Note that the values correspond to the median of the parameters for each model learned during each loop of the outer LOO cross validation.

Dataset	**Model**	**Position** [cm]	**Iterations**	σ
84 volunteers	Vector Valued	n.a.	53	1.45
	Independent	-8	81	2.58
		-4	148	3.87
		0	94	2.59
		4	74	2.56
		8	47	1.72
142 volunteers	Vector Valued	n.a.	65	0.96
	Independent	-8	93	2.16
		-4	88	2.16
		0	135	3.22
		4	148	3.23
		8	105	1.44

Table 7.2: Selected parameters

In Figure 7.2 we report the box plots for the LOO errors distributions on the 84 volunteers and the 142 volunteers datasets respectively, compared with the model in use at the "E.O. Ospedali Galliera" Hospital. It is evident that the vector valued approach does not bring any significant advantage over regressing on each measurement position independently. This may be due to many factors and still requires a thorough investigation. Obviously, the first culprit could be the parabolic feature map we designed, that imposes a too strong relationship among the measures. It might be that the parabolic assumption is valid on average, but for many volunteers it may not hold with a good approximation and thus it leads to predictions that are wrong. Furthermore, the experimental protocol we adopted to select the number of iterations for the ν-method and the width of the scalar gaussian kernel, in the vector valued case selects parameters that yield the best average error across all five components. Regressing on each component independently may allow to tailor the model parameters more accurately for each measurement position. However, the ν-method in the vector valued case, as well as in the scalar case, performs better than the parametric model developed by the researchers working with Prof. Marinelli. Our models show fewer outliers and improve on the second dataset, while the parametric model performs worse, probably due to the increased heterogeneity of the volunteers.

Our model regressing on each component independently is starting to be used alongside the parametric model in the hospital, since it has been shown to predict more accurately the background

signal for those patients that are particularly different from the volunteer population used to train the models. This assessment has been carried out visually by experts at the hospital that have acquired a strong experience on this problem. This evaluation confirms the expectation that the ν-method achieves a better generalization ability on samples not seen during training.

The accuracies obtained with these methods correspond to a precision in the iron overload estimation of about $0.8g$. Iron overload lower than $1g$ is considered mild: currently no model is capable to detect this kind of iron burden.

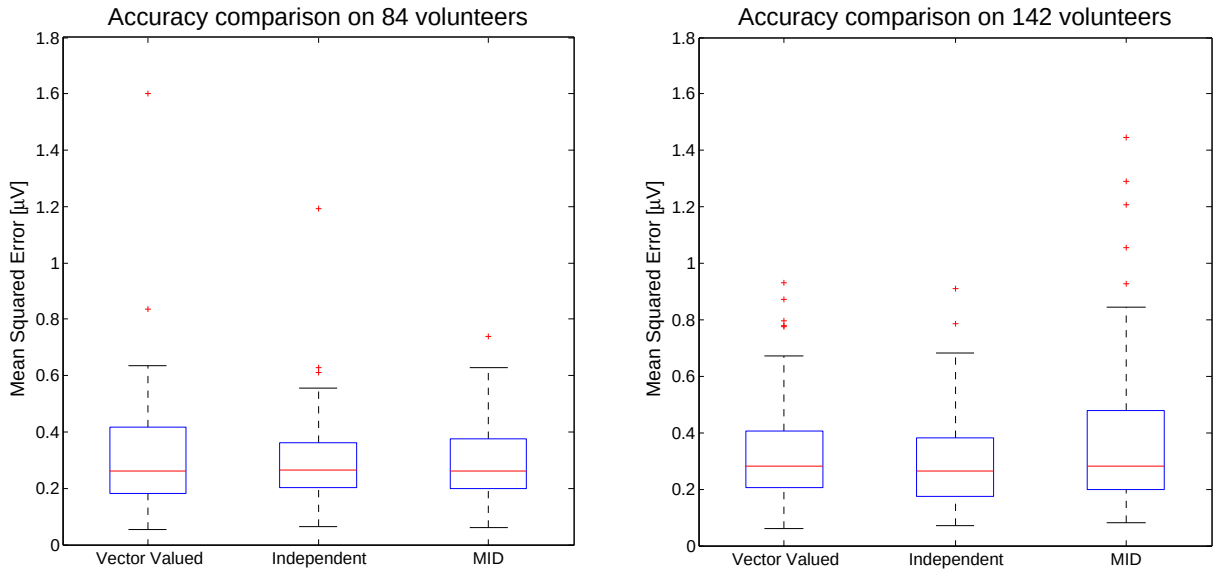


Figure 7.2: LOO errors distributions on the two datasets.

Chapter 8

Multi-modal Learning

In this chapter we present a theoretical framework for *Multi-modal learning*, that is, learning from patterns associated with very different kinds of sensors (e.g. video and audio). Our aim is to reconstruct an active modality, i.e. a sensorial pattern associated with grasping actions, from a passive one, i.e. a pattern describing the visual appearance of the object to be grasped. The grasping action is recorded by a special glove with sensors that register the spatial configuration of the hand. The output is therefore vector valued and we are interested in applying the proposed vector valued learning algorithms to this problems. However, since the relationship between input and output is many-to-many, that is an object can be grasped in more than one way, we have devised three data analysis workflows that decompose the problem in simpler steps. We show experimental results first on a simplified scenario, where the mapping between objects and grasps is one-to-one and then in a more complex scenario, offering two approaches. Our results indicate that it is indeed possible to accurately reconstruct the grasping action from the visual description, but that it is not straightforward to exploit the correlations that exist among the positions of the fingers.

This chapter is structured as follows. In the first Section we introduce the concept of multi-modality and relevant references. We propose our theoretical framework in Section 8.2, discussing the vision unit in Section 8.3. In Section 8.4 we present our experimental setup, the data acquisition procedure and the datasets we work with. The simpler scenario is tackled in Section 8.5, while in Section 8.7 we present our first approach on the more complex scenario that consists in first classifying the grasp type and then estimating the correct hand posture. In Section 8.6 we propose the alternative approach to first classify the object and then, knowing which grasp types are appropriate for that object category, estimate the hand configuration.

8.1 Introduction

Multi-modal learning, that is, learning from sensorial patterns associated with very different kinds of sensors, is paramount for biological systems. Coupled acoustic and visual information is essential, for instance, for animals to determine whether they are facing a predator or a prey. From the point of view of artificial intelligence, multi-modal learning is a potentially excellent way of enriching the input space of pattern recognition problems which could be otherwise more

difficult.

Indeed, sensorial inputs are available to biological systems in an endless, inextricably mixed flow coming from various sensorial apparatuses. It is not completely clear, then, how this information can be used to improve pattern recognition methods. For example, one could argue that the sight of a certain kind of predator is generally associated with a particular set of sounds and smells, and that animals learn to associate these multi-modal patterns during their infanthood; later on, this fact is employed in recognizing the associated danger in a dramatically improved way. It seems apparent, then, that there is a mapping among sensorial modalities; e.g., the auditory stimulus corresponding to a predator should be reconstructible from its visual appearance. Therefore, even though not all modalities are always available, it should be possible to recover one from another, to various degrees of precision.

Here we focus upon *active* perception modalities vs. *passive* ones. By active modality we mean perception arising from the *action* an agent performs in its environment, while, by passive modality, we mean perception of stimuli which are independent from the agent's will. Our paradigmatic case is grasping: objects must be grasped in the right way in order to use them as desired, but visual information only about an object is seldom sufficient to determine what kind of grasp to apply. Therefore the action of grasping must be reconstructed when the object is seen, so that the correct action can be performed.

In order to reconstruct actions from perception, we draw inspiration from the work on *mirror neurons* (Gallese et al., 1996; Rizzolatti and Craighero, 2004). Mirror neurons are clusters of neural cells which will fire if, and only if, an agent grasps an object *or* sees the same object grasped by another agent; they encode the semantics of an action associated to an object, and form the basis of *internal models* of actions, by which animals reconstruct the grasping and can therefore plan the grasp with greater robustness and effectiveness. Following the path laid out, e.g., in (Metta et al., 2006; Castellini et al., 2007), where perception-action maps have been built into artificial systems, we hereby propose a theoretical framework for multi-modal learning in which an active modality is reconstructed via statistical regression from a passive modality. In the worked example, visual patterns describing the sight of an object are used to reconstruct the related grasping postures of the hand. This framework can be theoretically extended to any such active-passive coupling.

Grasping classification is a keystone of many robotics applications. Different methods have been proposed in the literature according to the amount of prior knowledge available and the input data at disposal. Focusing on methods that exploit visual information at some level, a rather common approach starts from the computation of a full 3D model of the object to be grasped, with a subsequent association of an appropriate grasp type. Machine learning methods have been applied to this setting — see for instance (Pelossof et al., 2004), where a SVM regression model is built to estimate the optimal grasp from a set of synthetic objects. A practical problem with this approach, otherwise effective, is that often a 3D model of the object is not available nor easy to compute. Some methods based on 2D visual cues have been proposed (Piater, 2000; Saxena et al., 2006). Grasp classification is a loose definition that may refer to associating a grasp type from a pre-defined taxonomy to an object, or may be based on the explicit estimation of measurements modeling the grasp (e.g. describing the relative angles between joints). Our data-driven approach is related to the latter choice: it allows us

to learn a hand configuration appropriate to grasp a given object. A recent trend of robotic grasping relies on gathering some understanding on the models for human grasping. In this work we refer to human grasp classification (i.e. our data are originated by grasping actions performed by humans) and discuss how a possible grasp appropriate for an object may be estimated before actual grasping occur. If a mapping from human to robot grasping is available, the proposed work may be applied to human-oriented robotic grasp learning — see, for instance, (Ekvall and Kragic, 2004).

8.2 A theoretical framework for multi-modal learning

As outlined in the introduction to this chapter, we assume that there exists a mapping between sets of patterns belonging to different modalities — here we focus upon the relations which exist between a passive and an active modality. In the aforementioned example dealing with objects (as seen) and grasping them, something like what is shown in Figure 8.1 is sought for.

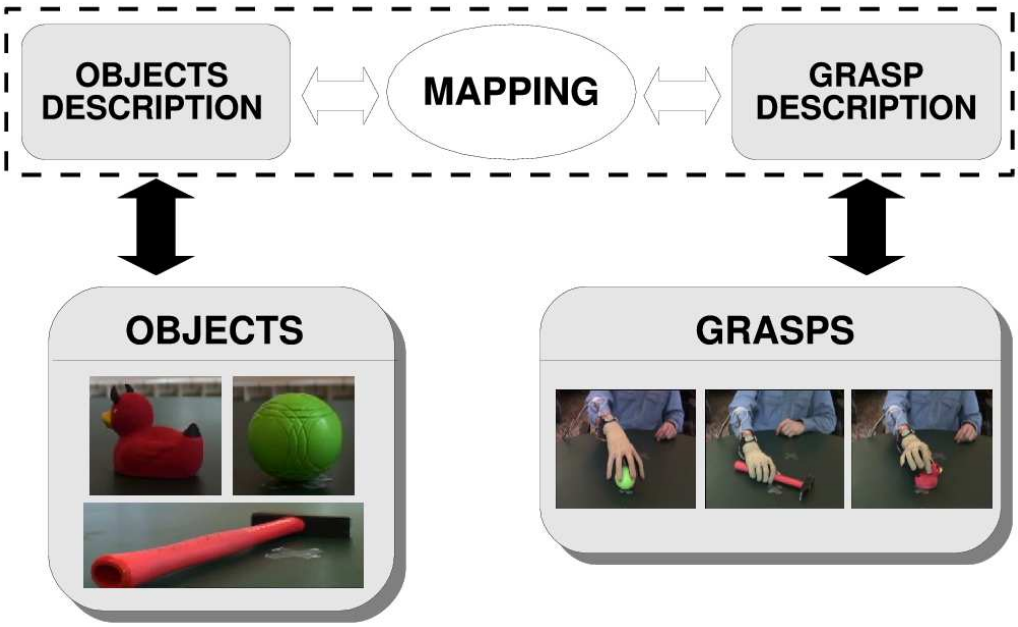


Figure 8.1: An instance of the framework we propose: estimating a mapping between appropriate visual descriptions of objects and classes of grasp actions.

In general, active modalities are not available to a biological system during the prediction phase, but only during the training phase. A paradigmatic example is that of a human infant learning how to grasp an object: by repeatedly trying to apply, e.g., a cylindric grasp to a bottle, he will learn not only to do it more and more efficiently, but also that a bottle is better be grasped cylindrically when moving it or bringing it close to the mouth. Later on, the sight of a bottle will remind the young human what one of the correct grasps is for that particular object. A *perception-to-action map* (PAM) is the equivalent of such training for a biological system: a model to reconstruct an active modality from a passive one. The PAM of our example is a mapping from visual features of an object to motor features of the grasping action used for that object. In general such a map is many-to-many: both a hammer and a bottle can be grasped

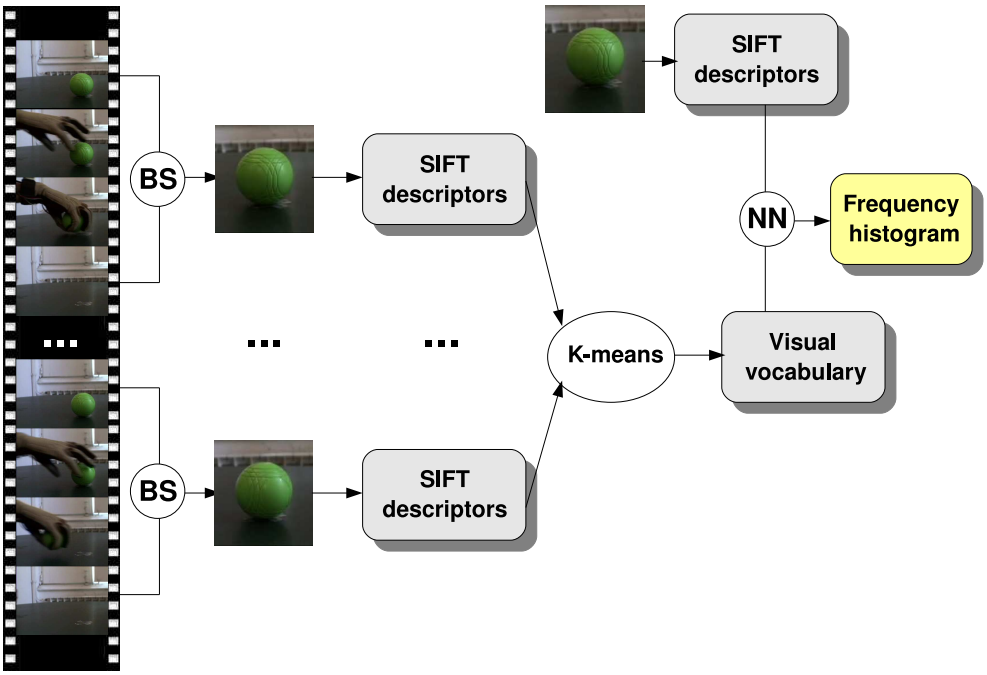


Figure 8.2: A schema of the vision unit. First, suitable frames are extracted from the sequence and objects are located by means of background subtraction (BS). SIFT descriptors of a set of random points are used as inputs for a clustering step to get to the final visual vocabulary. Finally, each image is represented with respect to the vocabulary adopting a nearest neighbor (NN) strategy (see text for details).

cylindrically[1], and as well a mug can be handled either cylindrically or by the handle. In the first application that we present in Section 8.5, we make the simplifying assumption that for a specific object there is just one acceptable grasping action — the PAM is one-to-one. In Section 8.7 and Section 8.6, we present the experiments performed on a more complicated scenario where the PAM is many-to-many. A PAM is useful in passive pattern recognition (e.g., classifying an object just by seeing it) since it augments the input space with PAM-reconstructed active patterns (e.g., classifying the same object from its sight *and the associated grasp*). Here we focus upon a simpler problem, namely that of checking whether, given the visual features of an object, the PAM-reconstructed grasp is similar to the one associated with that particular object. For example, we might train a model to reconstruct a pinch grip (hand posture) from the visual features of a pen; then in the prediction phase, given the visual features of another pen, will the reconstructed hand posture of a pinch grip look like a true pinch grip?

In particular, what is needed is: (*i*) a *vision unit* to extract visual features from an image or a series of images, (*ii*) a proper workflow to deal with a many-to-many PAM and exploit prior information, and (*iii*) a *regression unit* to build the PAM.

[1]the nomenclature of grasp types loosely follows that of Cutkosky (Cutkosky, 1989).

8.3 Vision unit

As we will discuss in Section 8.4, the system gathers, as one input, a video sequence acting as *spectator*, whose focus is on object appearance. The goal of the vision unit is to process the signal to obtain a global model of a set of given objects. Figure 8.2 shows the pipeline of the vision unit when considering only one object (the same procedure is applied to the whole set of objects). Among the sequence, we first select the frames showing only the object without any occlusion, then we locate more precisely its position by means of a simple background subtraction. We adopt an approach based on local features to describe image structures: due to their popularity a rich variety of local measurements have been proposed in the literature (Harris and Stephens, 1988; Mikolajczyk and Schmid, 2004; Lowe, 2004) and applied successfully to objects recognition and categorization problems (see (Csurka et al., 2004; Ferrari et al., 2006) just to name a few). Local approaches typically include two distinct steps: keypoints extraction and description. However, in our case, a keypoint based-representation often ends up into a poor description due to the limited size of the images. We thus built our representation by extracting enough random points to guarantee a more homogenous sampling. We chose to adopt SIFT descriptors (Lowe, 2004; Mikolajczyk and Schmid, 2005) to model image patches around these points, obtaining a set of *words* for each image.
To avoid redundancy and include some global information in our model, we follow the well-known bag-of-words approach (Csurka et al., 2004). In particular, we apply a clustering algorithm, namely k-means (Hartigan and Wong, 1979), to the words extracted from every image, building a *global* vocabulary of k words, containing SIFT descriptions of all known objects. Image representation is obtained as frequency histograms of visual words, selecting for each random point extracted from the image the most similar visual word as nearest neighbor. A normalization step is applied in order to set the sum of the frequencies obtained for each image equal to one.

8.4 Experimental setup

Data were collected in a highly controlled environment, using two Watec *WAT-202D* color cameras for the images and a 22-sensors Immersion *CyberGlove*[2] for the hand posture. An ascension *Flock-Of-Birds* magnetic tracker mounted on the subject's wrist, and a standard force sensing resistor glued to the subject's thumb were used to determine the hand position and speed, and the instant of contact with the object.

The cameras return two video sequences, one placed laterally with focus on the object (the *spectator*) and one placed in front of the subject (observing the *actor*). We process only the *spectator* video sequence, because it supplies all the information required for preliminary testing. The video sequence is acquired at 25Hz by each camera, while the glove is sampled at 100Hz. Since the three devices are independent of one another, a system of common time-stamps was used in order to synchronize the data.

The CyberGlove returns 22 8-bit numbers linearly related to the angles of the subject's hand joints. The resolution of the sensors is on average about 0.5 degree. The sensors describe the position of the three phalanxes of each finger (for the thumb, rotation and two phalanxes), the

[2] `http://www.cyberglovesystems.com`

Figure 8.3: (Top row) The objects used in our experiments: (left to right) *ball, pen, duck, pig, hammer, tape, lego.* (Bottom row) The grasp types we consider: (left to right) *cylindric power grasp, flat grasp, pinch grip, spherical and tripodal grip.*

four finger-to-finger abductions, the palm arch, the wrist pitch and the wrist yaw.

Among the motor data, it is reasonable to consider only the 22 measures of hand joints as the most relevant for accurately describing the intrinsic properties of each grasping type. When a grasp occurs the pressure on the force sensing resistor increases, causing the signal to vary, hence fixing the time-stamp of the event. Concurrently the values on each hand joint are stored as our output data. By synchronizing motor data and video sequence we select as input data the frame showing the object without clutter, going back along the sequence from the time-stamp in which the event occurs for a fixed amount of frames (see Figure 8.2, left). Our data are thus generated as pairs of image descriptors and sensor-motor values, respectively input and output data used to learn the models.

Setting 1. For the preliminary experiments of Section 8.5, we considered 7 objects and 5 grasping types identified by different hand postures (see Figure 8.3 and Table 8.1) with a one-to-one PAM, that is every object was associated to a unique grasp type. 2 volunteers have joined the experiment: for each object, the volunteer was asked to perform the required grasping action 20 times, so that the total data collected amounts to 280 (image, sensors data) examples.

Object	Ball	Pen	Duck	Pig	Hammer	Tape	Lego
Grasp type	Spherical	Pinch	Tripodal	Cylindrical	Flat	Spherical	Pinch

Table 8.1: The one-to-one object-to-grasp type associations for the simpler Setting 1.

Setting 2. For the experiments in Section 8.7 and Section 8.6, the problem is complicated by a many-to-many object-to-grasp type map, using the same objects and grasp types of the simpler setup. The data was collected from the actions performed by 20 volunteers grasping 7 objects with 5 grasp types in 13 different $\langle object, grasp \rangle$ pairs. Since each subject repeats a given $\langle object, grasp \rangle$ action 20 times, each pair is associated to 400 examples and the whole dataset contains 5200 grasping actions. The dataset is called VMGdB[3] and more details are

[3] `http://slipguru.disi.unige.it/Research/VMGdB/`

given in (Noceti et al., 2009b). Table 8.2 reports the 13 $\langle object, grasp \rangle$ pairs included in the dataset and the number of examples recorded per each.

	Tripodal	Spherical	Pinch	Cylindrical	Flat
BALL	400	400	-	-	-
PEN	400	-	400	-	-
DUCK	400	-	400	-	-
PIG	-	-	-	400	-
HAMMER	-	-	-	-	400
TAPE	400	400	400	-	-
LEGO	-	-	400	-	400

Table 8.2: The (object,grasp) pairs included in the VMGdB (Noceti et al., 2009b).

8.5 A simpler problem

In this section we deal with the estimation of the correct hand posture for grasping one of the 7 objects in the simpler Setting 1, where each object is associated to a single grasp type. This setting is quite straightforward, since it only requires to train a vector valued model from the visual description of an object onto the appropriate hand configuration to grasp it. We first describe the regression model we used, followed by the discussion of the experimental protocol and results.

8.5.1 Methods

Regression model

The mapping between object description and grasp description (Fig. 8.1) corresponds to a vector valued regression problem. We aim at estimating a function from feature vectors describing images of objects to sensor values, able to generalize on new data. We use the ν-method to train a vector valued model adopting the simple common similarity kernel (2.8) (see Section 2.3)

$$\Gamma_\omega(x, x') = K(x, x')(\omega \mathbf{1} + (1 - \omega)\mathbf{I}),$$

where K is a gaussian kernel, and compare its performance against independent models for each sensor trained with the ν-method. In order to render the magnitudes of the values recorded for each sensor more homogeneous, we normalized the measures diving by the constant factor $255 \cdot 22$, so that the sum of the components of each measurement vector is less than 1.

Experimental Protocol

We compare four different image representations, based on bag-of-words descriptors where the frequency histograms are computed for 20 and 50 words vocabularies on the entire image or on its four quadrants and then concatenated. We call the representations *W20, W20conc, W50* and *W50conc.* We consider two settings to evaluate the prediction performance of the

proposed algorithms. In the first setting (V1-V2) we build a training set with the data from the first volunteer and a test set with the data of the second volunteer (140 examples each). In the second setting (MIXED), we mix the data of both volunteers and perform a 5-fold cross validation (5-CV) to assess the performance of the algorithms. For both settings an internal 5-CV for each split of the external validation is used to select the stopping iteration for the ν-method and the value of the parameter ω that controls the similarity we are enforcing among the components of the vector valued function. The width of the gaussian kernel was computed on the training set as the mean of the euclidean distances between each input example and its k neighbors, where k is equal to 10% of the training examples.

8.5.2 Results

Table 8.3 summarizes the prediction errors evaluated according to the square loss on all 22 components. The values for the second setting are markedly lower because mixing the data of both volunteers reduces the variance between training and test sets in each split of the 5-CV. Therefore, if we aim at building a model generalizing on several people, it is crucial to collect data from a large variety of volunteers (even though other issues arise, see Section 8.6).

We observe that the regression model does not perform better than regressing on each sensor independently. Obviously, the relationships that might exist among the sensors is not as simple as the one imposed by the chosen matrix valued kernel. A more thorough investigation of the relations among the sensor values during each grasping action is required in order to design a custom matrix valued kernel to exploit them.

Setting	Representation	Vector Valued $[10^{-4}]$	Independent $[10^{-4}]$
V1-V2	W20conc	15.2	15.1
	W20	11.5	11.9
	W50conc	12.6	12.6
	W50	13.3	13.5
MIXED	W20conc	1.9 ± 0.4	2.0 ± 0.4
	W20	2.4 ± 0.4	2.6 ± 0.4
	W50conc	1.9 ± 0.3	2.0 ± 0.2
	W50	2.3 ± 0.7	2.3 ± 0.6

Table 8.3: Prediction accuracies of the ν-method for the vector valued and the independent model, for the two settings and four representations. The accuracy is expressed as mean square error over all sensors. In the MIXED scenario the associated standard deviation is reported as well.

Finally, in order to understand the quality of the predictions produced by the regression models, we aim at classifying the grasp type given the estimated sensor values. We consider only the MIXED setting, the dictionary with 50 words and the histogram computed on each of the four quadrants of the image, W50conc, and the predictions of the vector valued model. The input data are the sensor measures and the output data are the grasp classes, represented with 5-dimensional coding vectors (e.g. the tripodal grasp type is associated to the vector $(1, 0, 0, 0, 0)$ and so on). We train a RLS multi-category classifier in a One-vs-All configuration, that is training a regressor independently on each component of the multi-class coding vector, see Chapter 5 and (Rifkin et al., 2003). A 5-fold cross validation is performed to select the regularization

parameter, λ. For each split of the cross validation we use as training set the actual set of measures from the sensors paired with the corresponding vector coding for the grasp type, while as test set we use the estimates obtained with the vector valued regression model. The grasp classifier obtains an average prediction accuracy of 99.6%. This result indicates that the vector valued regression model performs well, predicting hand configurations very similar to the true ones and that almost perfectly characterize the different grasp types, confirming the validity of the idea underlying the proposed framework.

8.6 Learning the object type and the grasp

Here we consider the Setting 2, where the mapping between objects and grasp types is many-to-many. We explicitly use such mapping to decompose the hand posture estimation problem in two steps, since the approach adopted in the simpler one-to-one setting of the previous Section (and in (Noceti et al., 2009a)), to learn a vector valued model on all training examples will fail in this case. In fact, the regression model would average all the hand postures associated to the same object, yielding a configuration that does not represent any actual grasp (although it could carry information on the object volume). We also describe a procedure to evaluate the goodness of the predicted posture, that allows us to gain some understanding on the peculiarities of a given grasp type.

Our workflow is as follows. First, we classify the object. Secondly, knowing the object type, we activate only the regressors tailored to that object. This allows to estimate the measurements vector describing the hand position for the grasps associated with that specific object, see Figure 8.4 (a). The validation phase is based on comparing the estimated measurement vectors obtained with the activated regressors against the true hand postures from the test set and marking a correct estimation if the prediction closest to the true measures corresponds to the correct object and grasp type. The whole procedure is data-driven and is based on the ν-method, that we use for both estimating the object class and predicting an appropriate hand posture.

Even though the prediction of the correct hand posture is a natural vector valued regression problem, in this section we treat each sensor independently, for two reasons. Firstly, a preliminary data analysis phase, see Section 8.6.1, shows that there is not much correlation among the sensor values when we consider the data aggregated from several volunteers. Secondly, treating the components independently allows us to assess the relevance of each sensor for different grasp types applied to different objects. Consequently, we can evaluate the quality of the predicted hand posture taking into account the importance of each sensor.

8.6.1 Data and methods

We consider the Setting 2 and the dataset VMGdB. We do not apply any preprocessing to the sensor measurements, but a visual inspection of the data allows to draw some observations on inter-subject variability. In Figure 8.5 (top-left) we report the distribution of the values of sensor 14 measured for the actions performed with the *tripodal* grasp on the *ball* by all the subjects. Selecting only the action performed by subject 1 (top-middle) and 2 (top-right), we note that their distributions, although more peaked, differ significantly. This is not the case for sensor 21

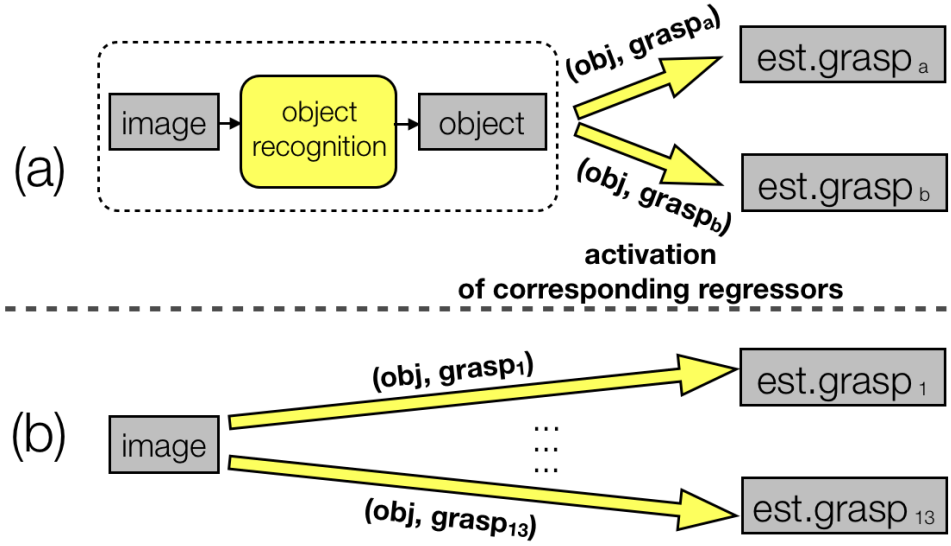


Figure 8.4: A schema of the pipeline: (a) the object classification step activates only the appropriate $\langle object, grasp \rangle$ regressors. (b) a more general pipeline used to validate all the regressors.

during the actions performed on the *pen* with the *tripodal* grasp. The overall distribution of its values, represented in Figure 8.5 (bottom-left) is more peaked and remains peaked around the same value even when we select subject 2 (bottom-middle) and 4 (bottom-right). It follows that sensor 21 in the $\langle pen, tripodal \rangle$ pair is less subject to inter-subject variability and we can expect to predict its values more accurately than for sensor 14 for the $\langle ball, tripodal \rangle$ pair.

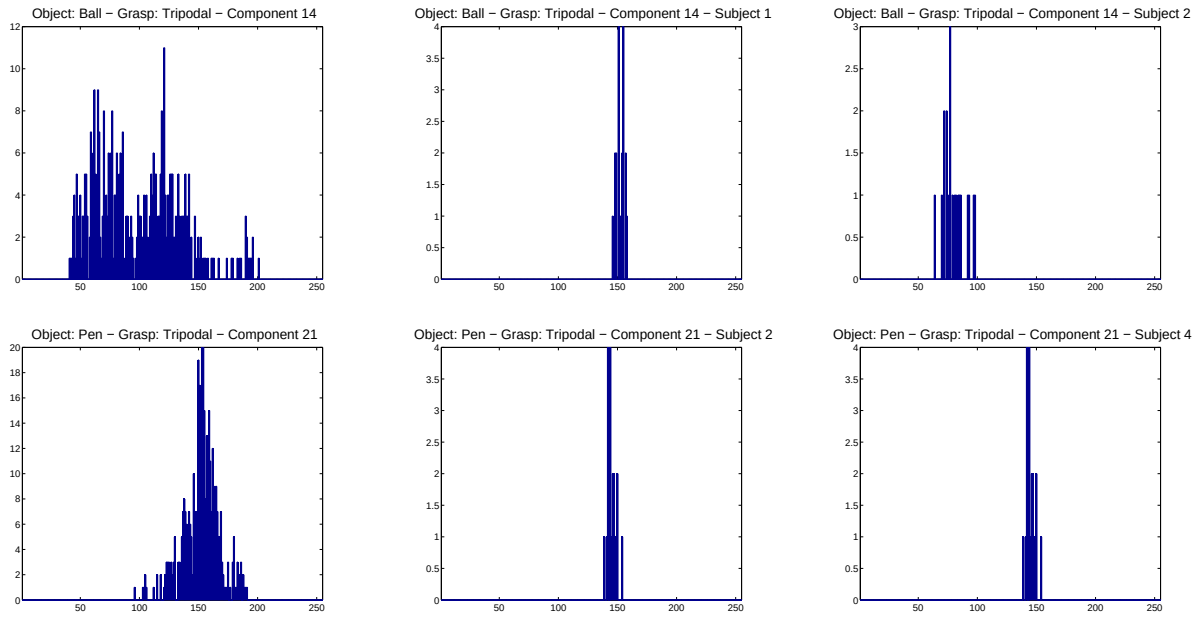


Figure 8.5: Top row: Distribution of values for sensor 14 for the $\langle Ball, Tripodal \rangle$ pair: all subjects (left), specific for subject 1 (center) and subject 2 (right). Bottom row: Distribution of values for sensor 21 for the $\langle Pen, Tripodal \rangle$ pair: all subjects (left), specific for subject 2 (center) and subject 4 (right).

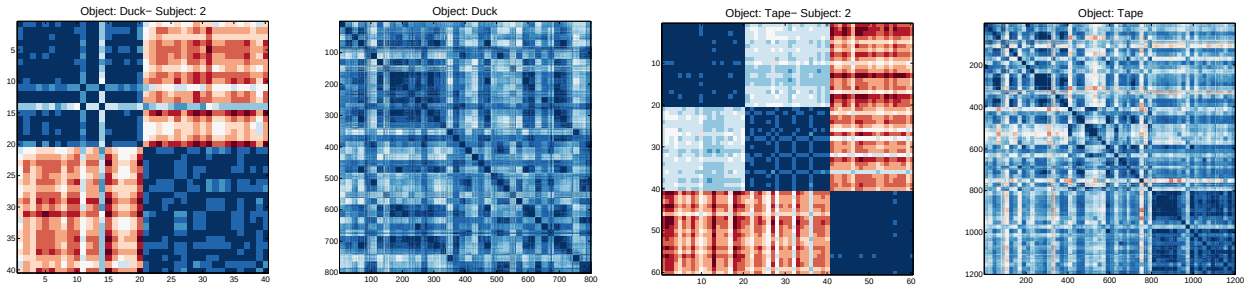


Figure 8.6: Color-coded distances among motor data. Grasping actions on *Duck* by subject 2 only (left), all the grasping actions for *Duck* (middle-left), grasping actions on *Tape* by subject 2 only (middle-right), all the grasping actions for *Tape* (right)

In Figure 8.6 we observe the euclidean distances between measurements vectors obtained from different grasping actions. We chose two different objects (*duck* and *tape*) to illustrate a common property of the data. The *duck* can be grasped in two different ways (*tripodal* and *pinch*) and this is evident if we observe the matrix of distances evaluated on subject 2 shown in Figure 8.6 (left). However this pattern does not emerge on the distances matrix evaluated on all subjects (sorted by grasp type). The *tape* can be grasped in three different ways (*tripodal*, *spherical* and *pinch*). Again, the differences among the grasps are evident if we observe just one subject. Conversely, this differentiation almost disappears when taking into account all the subjects, but for the *pinch* grasp, which is clearly more different than the *tripodal* or *spherical* grasp for all subjects.

To cope with the inter-subject variability of the sensor measurements, we apply a weighting scheme in the validation procedure. More relevance is assigned to those sensors whose values vary less from one subject to the other and therefore we can predict more accurately, see Section 8.6.2.

Regression model

Differently than the approach discussed in the previous Section and in (Baldassarre et al., 2010), we treat each sensor independently. This allows us to tailor the regressors for each sensor, evaluate which of them are most informative and discard those sensors that are mostly noise, due to inter-subject variability or irrelevance for the grasp type under consideration, as detailed in Section 8.6.2.

For every $\langle object, grasp \rangle_k$ pair, denoted by index k, each sensor $\ell = 1, \ldots, 22$ is associated to a regression function $f_k^\ell : \mathbb{R}^p \rightarrow \mathbb{R}$ trained on a set of examples $\{(x_i, y_i^\ell)\}_{i=1}^n$, where input vectors $x_i \in \mathbb{R}^p$ are $p-$dimensional visual features describing an object and $y_i^\ell \in \mathbb{R}$ is the sensor measurement corresponding to the grasping action performed on it.

Classification model

As shown in Figure 8.4 (a), a classification step is performed before the grasp estimation. The object classifier associates to a p-dimensional input vector of visual features the label of the depicted

object. We solve this multi-category problem with a common one-vs-all approach. Each object class is described with a coding vector: for object 1 we associate the vector $(1,0,0,0,0,0,0)$, to object 2 the vector $(0,1,0,0,0,0,0)$ and so on. We train an independent regressor for each component of these output vectors, using the ν-method. Given a new input example, we compute the predictions for all the classifiers and assign the example to the class corresponding to the regressor with highest output.

8.6.2 Training and validation of the regressors

For each of the 13 $\langle object, grasp \rangle_k$ pairs, the available data are divided in 300 examples for training and the remaining 100 for testing (the latter participate to a unique test set of 1300 data). We use the ν-method with a gaussian kernel to train a regressor f_k^ℓ for each sensor $\ell = 1, \ldots, 22$. The kernel parameter and the number of iterations are selected via 5-fold cross validation on the training set. The regressor is then trained again on the whole training set of 300 data with the best combination of parameters. In order to evaluate the effectiveness of the estimated grasp, we compare it with ground truth, weighting the sensors in order to favor the most meaningful for the case. An interesting side effect of the adopted distance is that we gain some understanding on the sensors relevant for a given grasp. The predictions on the training set are compared with the correct measurements and the average square error $Etr(f_k^\ell)$ for each $\ell = 1, \ldots, 22$ is computed. To each regressor f_k^ℓ we assign a score

$$w_{k,\ell} = \frac{Etr(f_k^\ell)}{\sigma_{k,\ell}^2},$$

where $\sigma_{k,\ell}$ are the standard deviations of the training values of each sensor. The scores $w_{k,\ell}$ are in the range $[0,1]$ — a small value of $w_{k,\ell}$ indicates that the corresponding regressor f_k^ℓ is predicting well.

Following the general evaluation pipeline shown in Figure 8.4 (b), given a test example x_i, for each pair k, the regressors $f_k = (f_k^1, \ldots, f_k^{22})$ are used to predict the hand configurations $\hat{\mathbf{y}}_{k,i} = (\hat{y}_{k,i}^1, \ldots, \hat{y}_{k,i}^{22}) = (f_k^1(x_i), \ldots, f_k^{22}(x_i))$. We assess the estimates computing a weighted distance between the predicted vector $\hat{\mathbf{y}}_{k,i}$ and the correct vector $\mathbf{y}_i$,

$$d(\hat{\mathbf{y}}_{k,i}, \mathbf{y}_i) = \sum_{\ell=1}^{22} (1 - w_{k,\ell}) \frac{(y_i^\ell - \hat{y}_{k,i}^\ell)^2}{\sigma_{k,\ell}^2}. \tag{8.1}$$

This distance confers more weight to those components that the regressors are able to predict more accurately (Figure 8.5 bottom) and less to the ones that perhaps are more noisy or less relevant for that specific grasp type (Figure 8.5 top). We then sort the regressors f_k according to this distance and select the one whose prediction $\hat{\mathbf{y}}_{k,i}$ is closest to the actual value $\mathbf{y}_i$. Since the selected regressor corresponds to a specific $\langle object, grasp \rangle_{k^*}$ pair, we can associate to the example x_i a predicted object category O_i^* and a grasp type G_i^*.

8.6.3 Results

The general evaluation workflow shown in Figure 8.4 (b) is applied to all 1300 test data. By comparing the estimated objects O_i^* and grasps G_i^* with the real object category O_i and grasp

type G_i, we can compute the average object classification error and the average grasp classification error, which are 27.6% and 14.8% respectively. Note that if one were to estimate the object and the grasp by chance, taking into account the different frequencies for each object and grasp type, the errors would be 84.0% and 75.7% respectively.

The pipeline summarized in Figure 8.4 (a) is evaluated by first predicting the object class for the test example x_i, with an accuracy of 99.4%, and then activating only the regressors associated with that object. Finding the closest prediction to the real value, according to the weighted distance (8.1), we obtain an average grasp classification error of 9.2%, improving on the accuracy of 14.9% obtained without pre-activation of the appropriate regressors. Table 8.4 summarizes the results.

	without activation	with activation
Grasp	14.8	9.2
Object	27.6	–
⟨O,G⟩ pair	32.3	9.2

Table 8.4: Classification errors for the two settings (a) and (b) - see Figure 8.4

8.7 Learning the grasp type and the grasp

In this section we propose an alternative approach to deal with the complexity of Setting 2. We use the many-to-many perception-to-action map (PAM) to train a vector valued multi-label classifier that predicts the most probable grasp type(s) associated to an image of an unknown object, without explicitly classifying the image into an object category. The most likely grasp(s) activate a visuo-motor regression module trained on pairs $\langle object, grasp \rangle$ related to a specific grasp type. In other words, we have a vector valued regressor to estimate the hand postures for the *tripodal* grasp type for all the objects that can be grasped with it, one for the *spherical* grasp type and so on. The regression module returns an estimate of the hand position for the grasp type(s) most appropriate for the (unknown) object under consideration. This final step is implicitly related to the object affordance, that is, the same grasp type (say, tripodal) applied to different objects will originate different hand positions, closely related to the object's size and consistency.

8.7.1 Data and Methods

The available 5200 data are organized in training and test sets as follows. Each group of 400 grasp actions is divided in two, where 200 actions are used for testing. From the remaining 200 actions we draw with no repetitions 10 actions for 10 times. We thus obtain 10 different training sets of 130 examples each, allowing us to check the dependence of the solution with respect to the specific data choice, and a test set of 2600 examples.

We use the *W50conc* visual representation, that corresponds to a dictionary containing 50 visual words whose frequencies are computed on the four quadrants of each image and then concatenated, resulting in a feature vector with 200 elements. Hence, each element corresponds to the relative frequency of a word of the dictionary in one of the four quadrants of the image. We

adopted a simple normalization of the measurements acquired by the CyberGlove, dividing each element by $22 \cdot 255$, so that every element, and the norm, of the measurement vector is less or equal to 1.

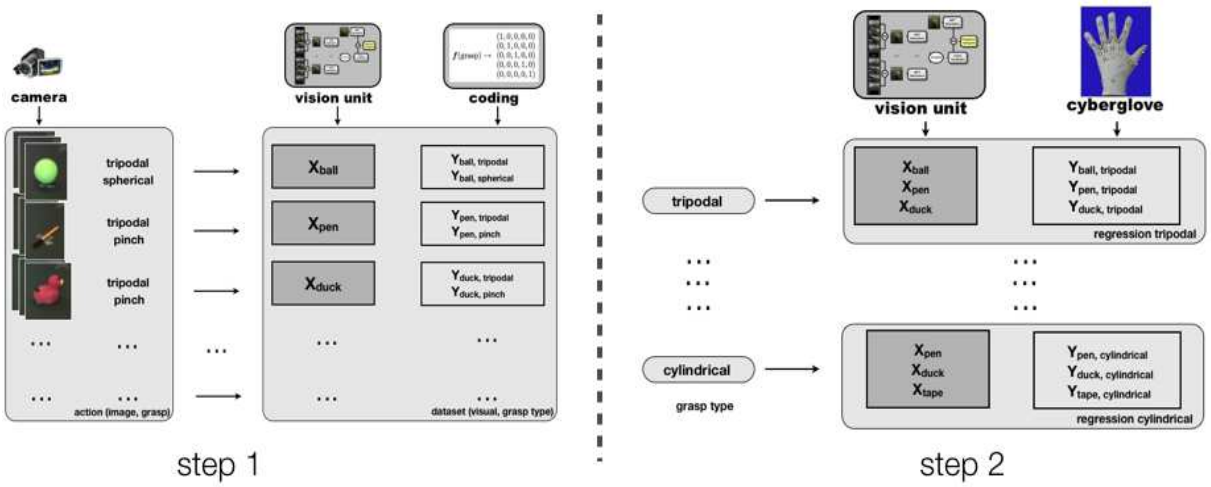


Figure 8.7: A schema of the two statistical learning modules. On the left the multi-category classifier associating to a visual representation the most likely grasp types; on the right the 5 vector valued regressors (one per grasp type) associating to visual representations measurements of hand positions.

The learning system consists of a cascade of two learning modules, that rely on vector valued models trained with the ν-method. More specifically, see Figure 8.7, the first module consists of a multi-category classifier. The multi-class problem is transformed into a vector valued regression problem by assigning to the examples of each class a vector valued coding, see Chapter 5 for more details. For instance, examples associated with *tripodal* grasp (grasp 1) are given the coding $(1, 0, 0, 0, 0)$. We use the ν-method to train five independent regressor for each component of the coding vectors. The optimal number of iterations and the width of the gaussian kernel used are chosen via a 5-CV on the training set. Given a new example, the classifier will estimate the probabilities associated to each grasp type and return the most probable grasp type or the grasp types whose probabilities are greater than a fixed threshold. The value of the threshold is varied during the experiments to assess the relationship between recall and specificity.

The second module consists of 5 vector valued regressors, one for each grasp type. Each regressor is trained only on examples that correspond to its specific grasp type. The models are trained using the ν-method and the simple common similarity kernel

$$\Gamma_\omega(x, x') = K(x, x')(\omega \mathbf{1} + (1 - \omega)\mathbf{I}),$$

with K a gaussian kernel. The optimal number of iterations and the kernel parameter ω are selected via a 5-CV on the training set. The width of the gaussian kernel is chosen as the average distance of each training point from its 10 nearest neighbors.

In the testing phase, a new visual representation of a given object is associated to a grasp type by module 1, which in turn activates the corresponding regressor in module 2. The final outcome is a vector estimating the hand posture specific for that $\langle object, grasp \rangle$ pair.

8.7.2 Results

We tested the classification performance of the first module by considering whether the most probable grasp type returned by the multi-category classifier is at least one of the possible grasp types associated to the given object. The average classification error over the 10 samplings of the training sets is $6.4\% \pm 1.5\%$.

A more detailed assessment of the grasp classifier is presented in Figure 8.8. The R.O.C. curve on the left side is computed varying the threshold on the probabilities returned by the first module and counting a *false negative* when a grasp type associated to that object has an estimated probability lower than the threshold. Conversely, for computing the r.o.c. curve on the right side, we consider a false negative only when none of the probabilities corresponding to the admissible grasp types are greater than the threshold.

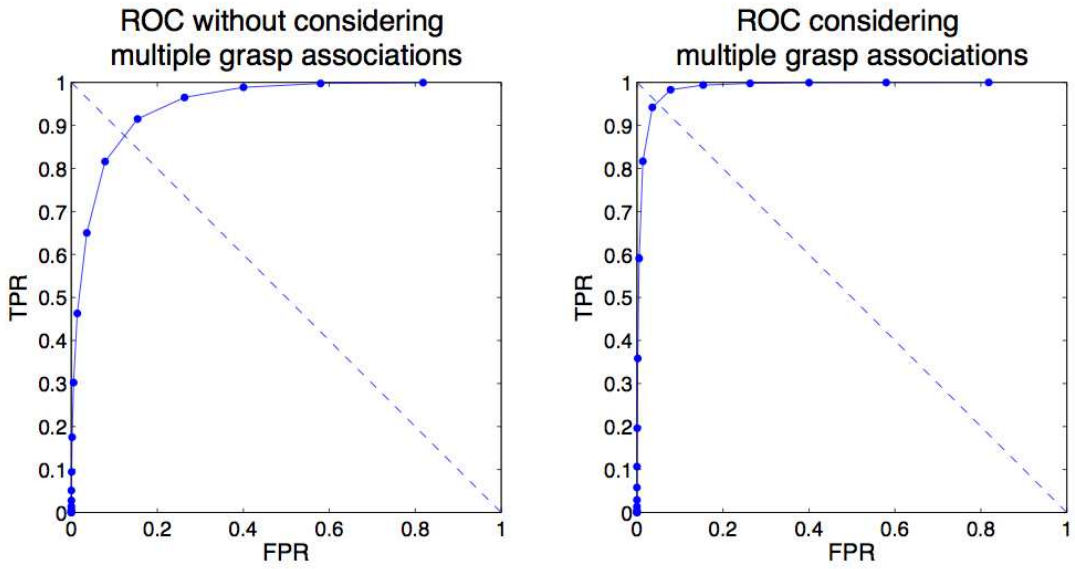


Figure 8.8: R.O.C. curves evaluating the grasp classifier performance. FPR stands for False Positive Rate and TPR for True Positive Rate. The higher the TPR for corresponding FPR the better the model.

In order to give a preliminary assessment of the overall system performance, we used a simple Nearest-Neighbor (NN) procedure. For each example in the test set, we computed its NN in the training set according to the Euclidean distance between the estimated and the true hand postures. We then compared the object and grasp classes of the NN with the true classes of the test example under consideration. We obtain an object *recall rate* of $52.6\% \pm 4.2\%$ and a grasp *recall rate* of $50.8\% \pm 1.8\%$. These results are better than a random guess (14% and 20%, respectively), but suggest that a more appropriate measure is needed to evaluate the regression module or perhaps that the inter-subjects variations disrupt the correlation between objects and hand posture (see the discussion in the previous section on inter-subject variability).

Chapter 9

Conclusions

This thesis presented a unifying framework for dealing with multi-output problems using matrix valued kernels and spectral filters. Spectral filters are algorithms for supervised learning that yield a regularized estimator by filtering out the unstable components of the kernel matrix. Some of the filters can be implemented as iterative algorithms that achieve very good computational performance, while retaining excellent generalizing abilities, as we demonstrated both theoretically and with numerical experiments.

We discussed the main differences between vector valued, multi-task and multi-class learning problems and showed how to use the spectral filters in each case. Our main theoretical result for vector valued learning is a finite bound on the excess risk of the estimator obtained with the spectral filters. The bound directly leads to consistency for all the algorithms. In other words, we are guaranteed that the solutions obtained with any spectral filter will approximate the best solution in the hypothesis space better and better as the number of training examples increases. Bayes consistency was proven for the multi-class extension that transforms a multi-category problem into the equivalent vector valued one of estimating a coding vector for each class and then applying a classification rule. This result guarantees that, as the number of training examples increases, the spectral filters yield a classifier that tends to the Bayes rule, that is the classification rule that assigns a data point to the class with the highest conditional probability. In the multi-task case, the error analysis is complicated by the fact that the training sets for each task might be different and of different cardinality and one could want to assess the risk of each task and not overall. We reserve to future work the investigation of these questions.

We reviewed several matrix valued kernels proposed recently in the literature, focusing on kernels that can be decomposed as the product of a scalar kernel on the input space and a positive semi-definite matrix of size equal to the number of dimensions of the output space. We established a connection between these kernels and regularizers on the components of the vector valued function. This result allows to apply spectral filters for the minimization of functionals that contain such regularizers, because the spectral filters act only on the kernel matrix computed on the training examples.

For the same class of kernels, we proposed a decomposition of the vector valued regression problem, that allows to dramatically reduce the computational burden of the algorithms. The experiments on synthetic and real data indeed show the computational advantage of using the

iterative spectral filters, especially the ν-method. Only when one chooses a Leave-One-Out procedure to select the optimal regularization parameter, Tikhonov regularization is again the fastest due to the results of Rifkin and Lippert (2007).

Our experiments on synthetic data showed that, for vector valued learning, exploiting the correlation among the outputs is much less straightforward than in multi-task learning, where we can essentially pool information from the training sets of each task. Only in the case of very strong prior information, like imposing a vector field to be divergence-free or curl-free or be a combination of two parts with those properties, it was possible to obtain a tangible advantage in using a vector valued approach. This may be due to the fact than when prior information is not available or strong enough, regressing on each component independently allows for a much more accurate fine tuning of the models.

In Chapter 7 we tackled the problem of estimating the liver iron overload from measurements obtained with the Magnetic Iron Detector, a non-invasive room-temperature biosusceptometer that is currently in use at the "E.O. Galliera" hospital in Genoa, Italy. A vector valued regression model is trained in order to estimate the background signal of the patients, that is the signal that would be obtained in absence of iron overload. From the difference between the measured and the estimated signal it is possible to recover the iron overload. The accuracy we obtained with a custom matrix valued kernel must be improved in order to be able to detect slight iron overloads and the experimental analysis shows that imposing a quadratic relationship among the measures might be unsatisfactory. In fact, we obtained the same precision with a model trained independently on each component. Future work will be focused on how to better translate into a proper matrix valued kernel the correlations that obviously exist among the measures, with the aim of increasing the prediction accuracy.

The framework for multi-modal learning that we proposed in Chapter 8 is an effective tool for augmenting the input space of learning models with reconstructed patterns from active modalities. For our paradigmatic case of grasping actions, we showed how we can deal with a many-to-many perception-to-action map, decomposing the estimation problem into simpler steps that use either multi-category classification or vector valued regression. We also presented effective evaluation schemes in order to realistically assess the performance of the entire workflow pipeline. Despite there exists much correlation among the outputs, that is among the relative positions of the fingers during the grasping actions, we observed that this correlation is disrupted by the inter-subject variability. However one cannot escape from this issue if the goal is to create a model that is not specific for a unique volunteer.

The main issue that emerged from the research presented in this thesis is the choice of the kernel. Differently than in the scalar case, where the gaussian kernel is the first choice for almost any problem, in the multi-output case there are no natural off-the-shelf kernels. One has to resort to his/her own intuition or knowledge of the problem in order to choose a kernel and, even in this case, the chosen kernel might depend on so many parameters, that it becomes unrealistic to select all of them in an unbiased data-driven manner. Despite the point of view of the regularizers offers slightly more insight on the regularization that is imposed on the components of the vector valued function, it remains difficult to assess to which degree such a regularization really encodes for the true underlying relationship among the components. Therefore, we believe that much research must be devoted to devise a procedure to either properly incorporate prior information

in the kernel, *kernel design*, or estimate a kernel from the data, *kernel learning*. Although quite few works recently addressed the kernel learning problem for scalar outputs (see (Bach et al., 2004; Micchelli and Pontil, 2007) and references therein), its extension to the multi-output case is still at its beginning (Zien and Ong, 2007; Lampert and Blaschko, 2008).

This challenge will greatly benefit from the speed with which iterative spectral filters can compute the solution to multi-output problems for any given kernel.

As a final note, we would like to suggest that it could be possible to extend the spectral filters to deal with matrix completion problems (Candes and Plan, 2009; Candes and Recht, 2009), where the aim is to reconstruct a matrix with missing values. Spectral filters could be used to obtained regularized matrices with specific properties that are commonly exploited for these problems, such as having low rank or small trace norm.

Bibliography

Abernethy, J., Bach, F., Evgeniou, T., and Vert, J.-P. (2009). A new approach to collaborative filtering: Operator estimation with spectral regularization. *Journal of Machine Learning Research*, 10:803–826.

Abernethy, J., Bartlett, P., and Rakhlin, A. (2007). Multitask learning with expert advice. *Lecture Notes in Computer Science*, 4539:484–498.

Alvarez, M., Luengo, D., and Lawrence, N. (2009). Latent force models. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, volume 5, pages 9–16.

Arfken, G. B. and Weber, H.-J. (2005). *Mathematical methods for physicists*. Elsevier.

Argyriou, A., Evgeniou, T., and Pontil, M. (2008a). Convex multi-task feature learning. *Machine Learning*, 73.

Argyriou, A., Maurer, A., and Pontil, M. (2008b). An algorithm for transfer learning in a heterogeneous environment. In *Proceedings of the 2008 European Conference on Machine Learning and Knowledge Discovery in Databases-Part I*, pages 71–85. Springer.

Aronszajn, N. (1950). Theory of reproducing kernels. *Transactions of the American Mathematical Society*, 68(3):337–404.

Bach, F., Lanckriet, G., and Jordan, M. (2004). Multiple kernel learning, conic duality, and the smo algorithm. In *Proceedings of the twenty-first International Conference on Machine Learning*. ACM New York, NY, USA.

Bakker, B. and Heskes, T. (2003). Task clustering and gating for bayesian multitask learning. *Journal of Machine Learning Research*, 4:83–99.

Baldassarre, L., Barla, A., Noceti, N., and Odone, F. (2010). Learning how to grasp objects. In *Proceedings of the 18th European Symposium on Artificial Neural Networks*.

Barron, J., Fleet, D., and Beauchemin, S. (1994). Performance of optical flow techniques. *International Journal of Computer Vision*, 12(1):43–77.

Bauer, F., Pereverzev, S., and Rosasco, L. (2007). On regularization algorithms in learning theory. *Journal of Complexity*, 23(1):52–72.

Bauer, F., Pereverzev, S. V., and Rosasco, L. (2006). A note on adaptive learning in reproducing kernel hilbert spaces. Technical Report DISI-TR-06-04, DISI Università di Genova.

Bonilla, E. V., Chai, K. M., and Williams, C. (2007). Multi-task gaussian process prediction. In *Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, Inc.

Bousquet, O. and Elisseeff, A. (2002). Stability and generalization. *Journal of Machine Learning Research*, 2:499–526.

Boyle, P. and Frean, M. (2005). Dependent gaussian processes. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press.

Breiman, L. and Friedman, J. H. (1997). Predicting multivariate responses in multiple linear regression. *Journal of the Royal Statistical Society*, 59(1):3–54.

Brudnak, M. (2006). Vector-valued support vector regression. In *International Joint Conference on Neural Networks*, pages 1562–1569.

Bühlmann, P. and Yu, B. (2002). Boosting with the l_2-loss: Regression and classification. *Journal of American Statistical Association*, 98:324–340.

Candes, E. and Plan, Y. (2009). Matrix completion with noise. *Proceedings of the IEEE*.

Candes, E. and Recht, B. (2009). Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 9:717–772.

Caponnetto, A. (2006). Optimal rates for regularization operators in learning theory. Technical report, CBCL Paper #264/ CSAIL-TR #2006-062, MIT.

Caponnetto, A. and De Vito, E. (2007). Optimal rates for regularized least-squares algorithm. *Foundations of Computational Mathematics*, 7(3):331–368.

Caponnetto, A., Micchelli, C., Pontil, M., and Ying, Y. (2008). Universal kernels for multi-task learning. *Journal of Machine Learning Research*, 9:1615–1646.

Carmeli, C., De Vito, E., and Toigo, A. (2006). Vector valued reproducing kernel Hilbert spaces of integrable functions and Mercer theorem. *Anal. Appl. (Singap.)*, 4(4):377–408.

Caruana, R. (1997). Multitask learning. *Machine Learning*, 28:41–75.

Castellini, C., Orabona, F., Metta, G., and Sandini, G. (2007). Internal models of reaching and grasping. *Advanced Robotics*, 21.

Chai, K. M. A., Williams, C. K. I., Klanke, S., and Vijayakumar, S. (2009). Multi-task gaussian process learning of robot inverse dynamics. In *Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, Inc.

Chapelle, O., Weston, J., and Scholkopf, B. (2003). Cluster kernels for semi-supervised learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 585–592. MIT Press.

Csurka, G., Dance, C., Fan, L., and Bray, C. (2004). Visual categorization with bag of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, volume 1, page 22. Citeseer.

Cutkosky, M. (1989). On grasp choice, grasp models and the design of hands for manufacturing tasks. *IEEE Transactions on Robotics and Automation*.

De Vito, E. and Caponnetto, A. (2005). Risk bounds for regularized least-squares algorithm with operator-valued kernels. Technical report, Massachusetts Institute of Technology - Computer Science and Artificial Intelligence Laboratory.

De Vito, E., Pereverzev, S., and Rosasco, L. (2008). Adaptive kernel methods via the balancing principle. Technical Report CBCL paper 275/CSAILTR-2008-062, MIT.

De Vito, E., Rosasco, L., and Caponnetto, A. (2005a). Discretization error analysis for tikhonov regularization. *Analysis and Applications*.

De Vito, E., Rosasco, L., Caponnetto, A., De Giovannini, U., and Odone, F. (2005b). Learning from examples as an inverse problem. *Journal of Machine Learning Research*, 6:883–904.

Ekvall, S. and Kragic, D. (2004). Interactive grasp learning based on human demonstration. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 3519–3524. Citeseer.

Engl, H. W., Hanke, M., and Neubauer, A. (1996). *Regularization of inverse problems*, volume 375 of *Mathematics and its Applications*. Kluwer Academic Publishers Group, Dordrecht.

Evgeniou, T., Micchelli, C. A., and Pontil, M. (2005). Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637.

Evgeniou, T., Pontil, M., and Poggio, T. (2000). Regularization networks and support vector machines. *Advances in Computational Mathematics*, 13(1):1–50.

Ferrari, V., Tuytelaars, T., and Van Gool, L. (2006). Simultaneous object recognition and segmentation from single or multiple model views. *Internation Journal of Computer Vision*, 67(2):159–188.

Gallese, V., Fadiga, L., Fogassi, L., and Rizzolatti, G. (1996). Action recognition in the premotor cortex. *Brain*, 119:593–609.

Girosi, F., Jones, M., and Poggio, T. (1995). Regularization theory and neural networks architectures. *Neural computation*, 7(2):219–269.

Griffin, G., Holub, A., and Perona, P. (2007). Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology.

Groetsch, C. W. (1984). *The theory of Tikhonov regularization for Fredholm equations of the first kind*, volume 105 of *Research Notes in Mathematics*. Pitman (Advanced Publishing Program), Boston, MA.

Hadamard, J. (1902). Sur les problemes aux derivees partielles et leur signification physique. *Princeton University Bulletin*, 13(1):49–52.

Harris, C. and Stephens, M. (1988). A combined corner and edge detector. In *Proceedings of The Fourth Alvey Vision Conference*.

Hartigan, J. A. and Wong, M. A. (1979). A k-means clustering algorithm. *Applied Statistics*, 28.

Hastie, T., Tibshirani, R., and Friedmann, J. (2001). *The Elements of Statistical Learning*. Springer.

Hein, M. and Bousquet, O. (2004). Kernels, associated structures and generalizations. Technical Report 127, Max Planck Institute for Biological Cybernetics.

Hjorth, J. (1994). *Computer intensive statistical methods: Validation model selection and bootstrap*. Chapman & Hall/CRC.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge regression: biased estimation for nonorthogonal problems. *Technometrics*, 8:27–51.

Ivanov, V. (1976). *The theory of approximate methods and their application to the numerical solution of singular integral equations*. Springer.

Izenman, A. J. (1975). Reduced-rank regression for the multivariate linear model. *Journal of Multivariate Analysis*, 5:248–264.

Jacob, L., Bach, F., and Vert, J. (2008). Clustered multi-task learning: A convex formulation. In *Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, Inc.

Kimeldorf, G. and Wahba, G. (1971). Some results on tchebycheffian spline functions. *J. Math. Anal. Applic*, 33(82-95):38.

Lampert, C. and Blaschko, M. (2008). A multiple kernel learning approach to joint multi-class object detection. *Lecture Notes in Computer Science*, 5096:31–40.

Lee, Y., Lin, Y., and Wahba, G. (2004). Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99(465):67–82.

Lo Gerfo, L., Rosasco, L., Odone, F., De Vito, E., and Verri, A. (2008). Spectral algorithms for supervised learning. *Neural Computation*, 20(7):1873–1897.

Lounici, K., Pontil, M., Tsybakov, A., and van de Geer, S. (2009). Taking advantage of sparsity in multi-task learning. In *Proceedings of the 22nd Conference On Learning Theory (COLT)*.

Lowe, D. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110.

Macêdo, I. and Castro, R. (2008). Learning divergence-free and curl-free vector fields with matrix-valued kernels. Technical report, Instituto Nacional de Matematica Pura e Aplicada.

Marinelli, M., Cuneo, S., Gianesin, B., Lavagetto, A., Lamagna, M., Oliveri, E., Sobrero, G., Terenzani, L., and Forni, G. (2006). Non-invasive measurement of iron overload in the human body. *IEEE Transactions on Applied Superconductivity*, 16(2).

Marinelli, M., Gianesin, B., Lamagna, M., Lavagetto, A., Oliveri, E., Saccone, M., Sobrero, G., Terenzani, L., and Forni, G. (2007). Whole liver iron overload measurement by a non-cryogenic magnetic susceptometer. In *Proceedings of New Frontiers in Biomagnetism*, Vancouver, Canada.

Mathé, P. and Pereverzev, S. V. (2002). Moduli of continuity for operator valued functions. *Numerical Functional Analysis and Optimization. An International Journal*, 23(5-6):623–631.

Maurer, A. (2006a). Bounds for linear multi-task learning. *Journal of Machine Learning Research*, 7:139.

Maurer, A. (2006b). The rademacher complexity of linear transformation classes. *Lecture Notes in Computer Science*, 4005:65.

Metta, G., Sandini, G., Natale, L., Craighero, L., and Fadiga, L. (2006). Understanding mirror neurons: A bio-robotic approach. *Interaction Studies*, 7:197–232.

Micchelli, C. and Pontil, M. (2005). On learning vector–valued functions. *Neural Computation*, 17:177–204.

Micchelli, C. and Pontil, M. (2007). Feature space perspectives for learning the kernel. *Machine Learning*, 66(2):297–319.

Micchelli, C. A. and Pontil, M. (2004). Kernels for multi-task learning. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press.

Mikolajczyk, K. and Schmid, C. (2004). Scale and affine invariant interest point detectors. *Internation Journal of Computer Vision*, 60:63–86.

Mikolajczyk, K. and Schmid, C. (2005). A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27:1615–1630.

Mukherjee, S., Niyogi, P., Poggio, T., and Rifkin, R. (2006). Learning theory: stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization. *Advances in Computational Mathematics*, 25(1):161–193.

Noceti, N., Caputo, B., Castellini, C., Baldassarre, L., Barla, A., Rosasco, L., Odone, F., and Sandini, G. (2009a). Towards a theoretical framework for learning multi-modal patterns for embodied agents. In *Proceedings of the 15th International Conference on Image Analysis and Processing*, page 248. Springer.

Noceti, N., Castellini, C., Caputo, B., and Odone, F. (2009b). Vmgdb –the contact visuo motor grasping database. preprint - http://slipguru.disi.unige.it/Research/VMGdB.

O'Leary, D., Stewart, G., and Vandergraft, J. (1979). Estimating the largest eigenvalue of a positive definite matrix. *Mathematics of Computation*, 33(148):1289–1292.

Pelossof, R., Miller, A., Allen, P., and Jebara, T. (2004). A svm learning approach to robotic grasping. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, pages 3215–8. Citeseer.

Penrose, R. (1955). A generalized inverse for matrices. In *Proc. Cambridge Philos. Soc*, volume 51, pages 406–413.

Piater, J. H. (2000). Learning visual features to predict hand orientations. In *Proceedings of ICML - Workshop in spatial knowledge*.

Poggio, T., Rifkin, R., Mukherjee, S., and Niyogi, P. (2004). General conditions for predictivity in learning theory. *Nature*, 428:419–422.

Rifkin, R. and Klautau, A. (2004). In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141.

Rifkin, R. and Lippert, R. A. (2007). Notes on regularized least squares. Technical report, MIT Dspace.

Rifkin, R., Yeo, G., and Poggio, T. (2003). *Advances in Learning Theory: Methods, Models and Applications*, chapter Regularized Least-Squares Classification. IOS Press, Amsterdam.

Rizzolatti, G. and Craighero, L. (2004). The mirror-neuron system. *Annual Review of Neuroscience*, 27:169–192.

Rosasco, L., Caponnetto, A., Vito, E. D., Piana, M., and Verri, A. (2004). Are loss functions all the same? *Neural Computation*, 16:1063–1076.

Saxena, A., Driemeyer, J., Kearns, J., and Ng, A. Y. (2006). Robotic grasping of novel objects. In *Advances in Neural Information Processing Systems (NIPS)*.

Schölkopf, B. and Smola, A. J. (2001). *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA.

Schwartz, L. (1964). Sous-espaces hilbertiens despaces vectoriels topologiques et noyaux associés (noyaux reproduisants). *Journal d'analyse mathématique*, 13(1):115–256.

Sheldon, D. (2008). Graphical multi-task learning. Technical report, Cornell University. Preprint.

Smola, A., Chaussee, R., and Sch Olkopf, B. (1998). From regularization operators to support vector kernels. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press.

Smola, A. and Kondor, R. (2003). Kernels and regularization on graphs. In *Proceedings of the 16th Conference On Learning Theory (COLT)*.

Stein, M. L. (1999). *Interpolation of spatial data*. Springer Series in Statistics. Springer-Verlag, New York.

Tewari, A. and Bartlett, P. L. (2005). On the consistency of multiclass classification methods. In *Proceedings of the 18th Annual Conference on Learning Theory*, volume 3559, pages 143–157. Springer.

Tikhonov, A. N. and Arsenin, V. Y. (1977). *Solutions of Ill-posed Problems*. John Wiley.

van der Merwe, A. and Zidek, J. V. (1980). Multivariate regression analysis and canonical variates. *Canadian Journal of Statistics*, 8:27–39.

Vapnik, V. (1998). *Statistical learning theory*. Wiley, New York.

Vapnik, V. (2000). *The nature of statistical learning theory*. Springer Verlag.

Vapnik, V. and Chervonenkis, A. (1974). *Theory of pattern recognition*. Nauka, Moscow.

Vazquez, E. and Walter, E. (2003). Multi output support vector regression. In *13th IFAC Symposium on System Identification, SYSID 2003*, pages 1820–1825, Rotterdam. IFAC.

Wahba, G. (1990). *Spline models for observational data*. Society for Industrial Mathematics.

Wold, S., Ruhe, H., Wold, H., and Dunn III, W. (1984). The collinearity problem in linear regression. the partial least squares (pls) approach to generalizes inverses. *SIAM Journal of Scientific and Statistical Computations*, 5:735–743.

Yao, Y., Rosasco, L., and Caponnetto, A. (2007). On early stopping in gradient descent learning. *Constructive Approximation*, 26(2):289–315.

Zhang, T. (2004). Statistical analysis of some multi-category large margin classification methods. *Journal of Machine Learning Research*, 5:1225–1251.

Zien, A. and Ong, C. (2007). Multiclass multiple kernel learning. In *Proceedings of the 24th International Conference on Machine Learning*, page 1198. ACM.