Computer Science and Artificial Intelligence Laboratory

Technical Report

MIT-CSAIL-TR-2011-004
CBCL-296

January 24, 2011

Multi-Output Learning via Spectral Filtering

Luca Baldassarre, , Lorenzo Rosasco,, Annalisa Barla,, and Alessandro Verri

massachusetts institute of technology, cambridge, ma 02139 usa — www.csail.mit.edu

Multi-Output Learning via Spectral Filtering

Luca Baldassarre[†], Lorenzo Rosasco[⋆,+], Annalisa Barla[†], Alessandro Verri[†]
⋆ *CBCL, McGovern Institute, Massachusetts Institute of Technology, Cambridge, MA, USA*
\+ *Istituto Italiano di Tecnologia, Genova, Italy*
† *DISI, Università di Genova*
baldassarre@disi.unige.it, lrosasco@mit.edu, {barla,verri}@disi.unige.it

January 24, 2011

Abstract

In this paper we study a class of regularized kernel methods for vector-valued learning which are based on filtering the spectrum of the kernel matrix. The considered 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 finite sample bounds for the excess risk of the obtained estimator and, in turn, these results allow to prove consistency both for regression and multi-category classification. Finally, we present some promising results of the proposed algorithms on artificial and real data.

1 Introduction

In this paper we study theoretical and computational properties of learning a vector-valued function using kernel methods. This problem has been recently considered in (Micchelli and Pontil, 2005) where the framework of vector-valued reproducing kernel Hilbert spaces was adopted and the representer theorem for Tikhonov regularization was generalized to the vector-valued setting. Our work can be seen as an extension of the work in (Micchelli and Pontil, 2005) aimed in particular at:

- Investigating the application of spectral regularization schemes (Lo Gerfo et al, 2008) to multi-output learning problems.
- Establishing consistency and finite sample bounds for Tikhonov regularization as well as for the other methods in the setting of vector-valued learning.
- Discussing the problem of multi-category classification within the vector-valued framework as well as Bayes consistency of spectral regularization methods.

A main outcome of our study is a general finite sample bound for spectral methods that leads to consistency. Moreover, we show in theory and practice how iterative methods can be computationally much more efficient than Tikhonov regularization. As a byproduct of our analysis we discuss theoretical and practical differences among vector valued learning, multi-task learning and multi-category classification.

Classical supervised learning focuses on the problem of estimating functions with scalar outputs: a real number in regression and one between two possible labels in binary classification. The starting point of our investigation is the observation that in many practical problems it is convenient to model the object of interest as a function with multiple outputs. In machine learning this problem typically goes under the name of multi-task or multi-output learning and has recently attracted a certain attention. It is interesting to recognize at least two classes of problems with multiple output functions. The first class, that we might call multi-task learning, corresponds to the situation in which we have to solve several standard scalar learning problems that we assume to be related, so that we can

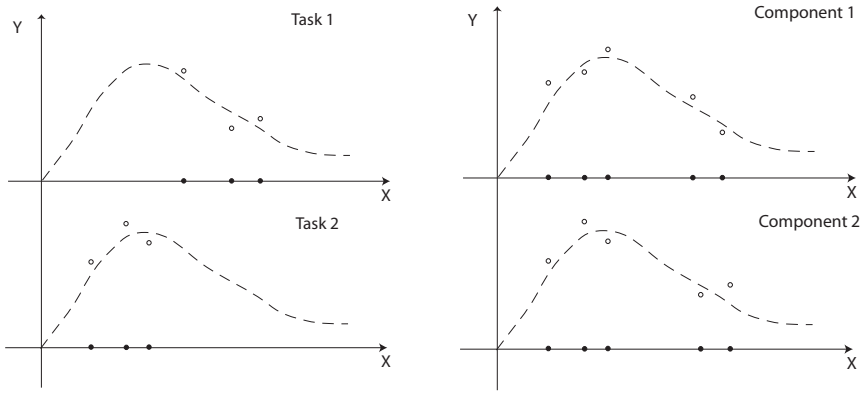

Figure 1: Comparison of a multi-task and a vector-valued learning problem. We consider a simplified situation in which there are only two tasks/components and they are the same function. In the multi-task case, the tasks can be sampled at different input points, whereas in the vector-valued case it is natural to assume all the components to be sampled at the same input points.

expect to obtain a better solution if we attempt to solve them simultaneously. A practical example is the problem of modeling buying preferences of several people based on previous purchases (Evgeniou et al, 2005). People with similar tastes tend to buy similar items and their buying histories are probably related. The idea is then to predict the consumer preferences for all individuals simultaneously by solving a multi-output learning problem. Each consumer is modelled as a task and its previous preferences are the corresponding training set. The second class of problems corresponds to learning vector-valued functions. This situation is better described as a supervised learning problem where the outputs are vector-valued. For example, a practical problem is that of estimating the velocity field of an incompressible fluid from scattered spatial measurements (see experiments section).

The two problems are clearly related. Indeed, we can view tasks as components of a vector valued function or equivalently learning each component of a vector-valued function as one of many scalar tasks. Nonetheless, there are also some differences that make the two problems different both from a practical and a theoretical point of view. For example, in multi-task learning the input points for each task (component) can be different, they can be represented by different features and the sample size might vary from one task to the other. In particular, each task can be sampled in a different way so that, in some situations, we can essentially augment the number of effective points available for each individual task by assuming that the tasks are highly correlated. This effect does not occur while learning vector fields - see Figure 1 - where each component is sampled at the same input points. 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, 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.

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 in (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). Classical approaches to learning vector values function in artificial intelligence include neural networks algorithms (Bishop, 2006). More recently 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, 2006). 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 in (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.

In this paper we focus primarily on vector-valued learning as a natural extension of the classical scalar setting. Indeed, many of the computational ideas we discuss apply to general multi-output problems, but some of the theoretical results are specific for vector-valued functions. The main contribution of this paper is a complete analysis of a class of regularized kernel methods for vector-valued learning. The description and motivation of the considered algorithms differ from those of penalized empirical risk algorithms. Each algorithm has a natural definition in terms of spectral filtering of the kernel matrix, designed to suppress contributions corresponding to small eigenvalues. This justifies calling these methods spectral regularization. The rationale behind them is the connection between learning theory and regularization of ill-posed problems (De Vito et al, 2005) and, more generally, the results showing the relation between stability and generalization (Poggio et al, 2004; Bousquet and Elisseeff, 2002). Indeed, one of our results is an excess risk bound that ensures generalization properties and consistency of spectral regularization. Though the analysis can be done in a unified framework, the specific form of the filter enters the bounds.

The various methods have different computational properties. As we show, both in theory and practice, iterative algorithms, that can be seen as extensions of L2 boosting (Bühlmann and Yu, 2002), can outperform Tikhonov regularization from the computational point of view while preserving its good learning performance. The complexity analysis we provide takes into account the specific form of the kernel as well as the regularization parameter choice step. The empirical performance of spectral filtering methods are tested in multi-task and vector-valued learning both for toy and real data.

Finally, we give a theoretical discussion on the application of vector field learning techniques in the context of multi-category classification. We show how to formulate a multi-class problem as a vector-valued learning problem and discuss the role played by the coding strategy. The difference between a one-versus-all approach and an approach where the correlation among classes is taken into account is clear within the vector-valued framework. Bayes consistency of spectral filtering methods easily follow from the aforementioned excess risk bounds. Some of the material in this paper has been presented in (Baldassarre et al, 2010). The conference paper contains only the discussion on vector fields learning with no proofs and limited experimental analysis.

The plan of the paper follows: in Sect. 2 we recall some basic concepts, in Sect. 3 we present the class of algorithms under study and the finite sample bound on the excess risk. In Sect. 4 we discuss examples of kernels and computational issues, in Sect. 5 we illustrate applications to multi-class classification, while in Sect. 6 we discuss multi-task learning. Experimental analysis is conducted in Sect. 7 and we conclude in Sect. 8 proposing some future work.

2 Learning Vector-valued Functions with Kernels: Basic Concepts

We start by setting the notation and recalling some elementary facts. We consider vector-valued learning and present the setup of the problem, as well as the basic notions behind the theory of vector-valued reproducing kernels.

2.1 Supervised Learning as Function approximation

The problem of supervised learning amounts to inferring an unknown functional relation given a finite *training set* of input-output pairs $\mathbf{z} = \{(x_i, y_i)\}_{i=1}^n$ that are randomly sampled and noisy. More precisely, the training points are assumed to be identically and independently distributed according to a fixed, but unknown probability measure $\rho(x, y) = \rho_X(x)\rho(y|x)$ on $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, where usually $\mathcal{X} \subseteq \mathbb{R}^p$ and $\mathcal{Y} \subseteq \mathbb{R}$. Here we are interested in vector-valued learning where $\mathcal{Y} \subseteq \mathbb{R}^d$. A learning algorithm is a map from a training set $\mathbf{z}$ to an estimator $f_{\mathbf{z}} : \mathcal{X} \to \mathcal{Y}$. A good estimator should generalize to future examples and, if we choose the square loss, this translates into the requirement of having small *expected risk* (or error)

$$\mathcal{E}(f) = \int_{\mathcal{X}\times\mathcal{Y}} \|y - f(x)\|_d^2 d\rho(x, y),$$

where $\|\cdot\|_d$ denotes the euclidean norm in $\mathbb{R}^d$. In this framework the ideal solution is the minimizer of the expected error, that is the regression function $f_\rho(x) = \int_{\mathcal{Y}} y\rho(y|x)$, but cannot be directly calculated since ρ is unknown. Further, the search for a solution is often restricted to some space of hypotheses $\mathcal{H}$. In this case the best attainable error is $\mathcal{E}(f_{\mathcal{H}}) = \inf_{f\in\mathcal{H}} \mathcal{E}(f)$. The quality of an estimator can be assessed considering the distribution of the *excess risk*, $\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_{\mathcal{H}})$, and in particular, we say that an estimator is consistent if

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

for all positive ε, where $\mathrm{P}[A]$ is the probability of the event A. A more quantitative result is given by finite sample bounds,

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

We add two remarks on related problems that we discuss in the following. The first is multi-task learning.

Remark 1. *In multi-task learning (MTL) (Evgeniou et al, 2005; Caponnetto et al, 2008; Micchelli and Pontil, 2004) the goal is to learn several correlated scalar problems simultaneously. For each task $j = 1, \ldots, d$ we are given a training set of examples $S_j = \{(x_{ij}, y_{ij})\}_{i=1}^{n_j}$. The examples are often assumed to belong to the same space $\mathcal{X} \times \mathcal{Y}$ and, if this is the case, vector-valued learning corresponds to the case where the inputs are the same for all tasks.*

The second problem is multi-category classification.

Remark 2. *It is well known that binary classification can be seen as a regression problem where the output values are only ± 1. In the multi-class case the naïve idea of assigning a label $y \in \{1, 2, \ldots, d\}$ to each class introduces an artificial ordering among the classes. A possible way to solve this issue is to assign a "code" to each class, for example class 1 can be $(1, 0 \ldots, 0)$, class 2 $(0, 1 \ldots, 0)$ etc. In this case, we can see the problem as a vector-valued regression problem. As we discuss in Sect. 5, this point of view allows to show that the spectral regularization algorithms we consider can be used as consistent multi-class algorithms.*

2.2 Vector-valued RKHS

In the following we are interested in the theoretical and computational properties of a class of vector-valued kernel methods, that is, methods where the hypotheses space is chosen to be a reproducing kernel Hilbert space (RKHS). This motivates recalling the basic theory of vector-valued RKHS.

The development of the theory in the vector case is essentially the same as in the scalar case. We refer to (Micchelli and Pontil, 2005; Carmeli et al, 2006) for further details and references. We consider functions having values in some euclidean space $\mathcal{Y}$ with scalar product (norm) $\langle\cdot,\cdot\rangle_{\mathcal{Y}}$, ($\|\cdot\|_{\mathcal{Y}}$), for example $\mathcal{Y} \subset \mathbb{R}^d$. A RKH space, $\mathcal{H}$, is a Hilbert space of functions $f : \mathcal{X} \to \mathcal{Y}$, with scalar product (norm) denoted by $\langle\cdot,\cdot\rangle_\Gamma$ ($\|\cdot\|_\Gamma$), such that the evaluation maps $ev_x : \mathcal{H} \to \mathcal{Y}$ are linear and bounded, that is

$$\|f(x)\|_{\mathcal{Y}} = \|ev_x f\|_{\mathcal{Y}} \leq C_x \|f\|_\Gamma. \tag{1}$$

A reproducing kernel Γ is then defined as:

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

so that $\Gamma : \mathcal{X} \times \mathcal{X} \to \mathcal{B}(\mathcal{Y})$, where $\mathcal{B}(\mathcal{Y})$ is the space of bounded operators on $\mathcal{Y}$ and ev_x^* is the adjoint[1] of ev_x. Note that for $\mathcal{Y} \subset \mathbb{R}^d$ the space $\mathcal{B}(\mathcal{Y})$ is simply the space of $d \times d$ matrices.
By definition, the kernel Γ has the following reproducing property, for all $c \in \mathcal{Y}$ and $x \in \mathcal{X}$

$$\langle f(x), c\rangle_{\mathcal{Y}} = \langle f, ev_x^* c\rangle_{\Gamma} = \langle f, \Gamma_x c\rangle_{\Gamma}\,, \tag{2}$$

where $ev_x^* y = \Gamma_x y = \Gamma(\cdot, x)y$. It follows that in (1) we have $C_x \leq \sup_{x\in\mathcal{X}} \|\Gamma(x,x)\|_{\mathcal{Y},\mathcal{Y}}$, where $\|\cdot\|_{\mathcal{Y},\mathcal{Y}}$ is the operator norm. We assume throughout that

$$\sup_{x\in\mathcal{X}} ||\Gamma(x,x)||_{\mathcal{Y},\mathcal{Y}} = \kappa < \infty. \tag{3}$$

Similarly to the scalar case, it can be shown that for any given reproducing kernel Γ, a unique RKHS can be defined by considering the completion of the space

$$\mathcal{H}_N = \{\sum_{i=1}^{N} \Gamma(\cdot, x_i)c_i \mid c_i \in \mathcal{Y}, x_i \in \mathcal{X}, i = 1, \ldots, N\}$$

with respect to the norm induced by the inner product

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

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

In Sect. 4.1 we discuss several examples of kernels corresponding to vector-valued RKHS (see also (Micchelli and Pontil, 2005; Evgeniou et al, 2005)). To avoid confusion in the following we denote with K scalar kernels and with Γ reproducing kernels for vector-valued RKHS.

Remark 3. *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 $\Gamma(x, x')_{\ell q} = Q((x, \ell), (x', q))$. See (Hein and Bousquet, 2004) for more details.*

3 Learning Vector-valued Functions with Spectral Regularization

In this section we present the class of algorithms under study. First, we briefly recall the main features of Tikhonov regularization for scalar and vector problems. On the one hand, this allows us to point out the role played by vector-valued RKHS and, on the other hand, it will help us introducing the spectral regularization methods of which Tikhonov is a special case. Second, we discuss the general framework of spectral methods as well as several examples of algorithms. Third, we state a finite sample bound on the excess risk.

3.1 Tikhonov Regularization from the Scalar to the Vector Case

In this section we start from the Tikhonov regularization 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.

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

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

Its solution is given by

$$f_{\mathbf{z}}^{\lambda}(\cdot) = \sum_{i=1}^{n} c_i K(x_i, \cdot), \quad c_i \in \mathbb{R} \quad \forall i = 1, \ldots, n \tag{4}$$

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

where the coefficients $\mathbf{c} = (c_1, \ldots, c_n)^\top$, satisfy

$$(\mathbf{K} + \lambda n I)\mathbf{c} = \mathbf{y}, \tag{5}$$

with $\mathbf{K}_{ij} = K(x_i, x_j)$, $\mathbf{y} = (y_1, \ldots, y_n)^\top$ and I is the $n \times n$ identity matrix.

The final estimator $f_{\mathbf{z}}$ is determined by a parameter choice $\lambda_n = \lambda(n, \mathbf{z})$, so that $f_{\mathbf{z}} = f_{\mathbf{z}}^{\lambda}$.
In the case of vector-valued output, i.e. $\mathcal{Y} \subset \mathbb{R}^d$, 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

$$\mathcal{H} = \mathcal{H}^1 \times \mathcal{H}^2 \cdots \times \mathcal{H}^d, \tag{6}$$

where the spaces $\mathcal{H}^1, \mathcal{H}^2, \ldots, \mathcal{H}^d$ are endowed with norms $\|\cdot\|_{\mathcal{H}^1}, \ldots, \|\cdot\|_{\mathcal{H}^d}$. Then $f = (f^1, \ldots, f^d)$ and $\|f\|_{\Gamma}^2 = \sum_{j=1}^d \|f^j\|_{\mathcal{H}^j}^2$. Tikhonov regularization amounts to solving the following problem

$$\min_{f \in \mathcal{H}} \{\frac{1}{n} \sum_{i=1}^n \|y_i - f(x_i)\|_d^2 + \lambda \|f\|_{\Gamma}^2\} \tag{7}$$

that can be rewritten as

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

From the above expression it is clear that the solution of the problem is equivalent to solving d *independent* scalar problems. Within the framework of vector-valued kernels, assumption (6) corresponds to a special choice of a matrix valued kernel, namely a kernel of the form

$$\Gamma(x, x') = diag(K_1(x, x'), \ldots, K_d(x, x')).$$

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 (7) 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 result in (Micchelli and Pontil, 2005) shows that the regularized solution can be written as

$$f_{\mathbf{z}}^{\lambda}(\cdot) = \sum_{i=1}^n \Gamma(\cdot, x_i) c_i, \quad c_i \in \mathbb{R}^d \quad \forall i = 1, \ldots, n. \tag{8}$$

The coefficients can be concatenated in a nd vector $\mathbf{C} = (c_1^\top, \ldots, c_n^\top)^\top$ and satisfy

$$(\mathbf{\Gamma} + \lambda n I)\mathbf{C} = \mathbf{Y} \tag{9}$$

where $\mathbf{Y} = (y_1^\top, \ldots, y_n^\top)^\top$ is the nd vector where we concatenated the outputs and the kernel matrix $\mathbf{\Gamma}$ is a $d \times d$ block matrix, where each block is a $n \times n$ scalar matrix, so that $\mathbf{\Gamma}$ is a $nd \times nd$ scalar matrix, while I is the $nd \times nd$ identity matrix.

Remark 4. *We observe that the kernel matrix corresponding to (9) has a block diagonal structure. Indeed, the presence of off-diagonal terms reflects the dependence among the components.*

We will now present a class of spectral regularization methods that contains Tikhonov regularization as a special case.

3.2 Beyond Tikhonov: Regularization via Spectral Filtering

In this section we present the class of regularized kernel methods under study, referring to (Lo Gerfo et al, 2008; Bauer et al, 2007) for the scalar case. 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

corresponding 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. More precisely, the solution of (unpenalized) empirical risk minimization can be written as in (8), but the coefficients are given by

$$\mathbf{\Gamma C} = \mathbf{Y}. \tag{10}$$

Comparing the above expression to (9) 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 corresponding 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, 2005) motivates considering spectral regularization techniques, where the spectrum is defined as the set of the eigenvalues of a square matrix. 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 $(\mathbf{\Gamma} + \lambda nI)^{-1}$ 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

$$\mathbf{\Gamma} = \mathbf{USU}^*$$

is the eigendecomposition of $\mathbf{\Gamma}$ with $\mathbf{S} = diag(\sigma_1, \ldots, \sigma_n)$ containing its eigenvalues, then

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

and

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

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 (8) with coefficients given by

$$\mathbf{C} = g_\lambda(\mathbf{\Gamma})\mathbf{Y}. \tag{11}$$

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 5. *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 following, 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 Sect. 3.4 we provide a formal definition. The latter will be the key to give an error analysis for the different algorithms within a unified framework.

3.3 Examples of Spectral Regularization algorithms

We now describe several examples of algorithms that can be cast in the above framework.

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, hence the regularization parameter is the number of iterations.

The coefficients (11) can be found by setting $\mathbf{C}^0 = 0$ and considering for $i = 1, \ldots, t$ the following iteration

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

where the step size η can be chosen to make the iterations converge to the minimizer of the empirical risk - see (12) below. It is easy to see that this is simply gradient descent if we use (8) to write the empirical risk as

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

The corresponding filter function can be found noting that

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

and indeed one can prove by induction that the solution at the $t-$th iteration is given by

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

Then, the filter function is $G_\lambda(\sigma) = \eta \sum_{i=0}^{t-1}(I - \eta\sigma)^i$. Interestingly, this filter function has another interpretation that can be seen recalling that $\sum_{i=0}^{\infty} x^i = (1-x)^{-1}$, for $0 < x < 1$. In fact, a similar relation holds if we consider matrices rather than scalars, so that, if we replace x with $1 - \eta\mathbf{\Gamma}$, we get

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

The filter function of L2 boosting corresponds to the truncated power series expansion of $\mathbf{\Gamma}^{-1}$. The last reasoning also shows a possible way to choose the step-size. In fact we should choose η so that

$$\|I - \eta\mathbf{\Gamma}\| < 1, \tag{12}$$

where we use the operator norm.

Next, we briefly discuss three other methods.

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

$$\begin{aligned}
\mathbf{C}^i &= \mathbf{C}^{i-1} + u_i(\mathbf{C}^{i-1} - \mathbf{C}^{i-2}) + \frac{\omega_i}{n}(\mathbf{Y} - \mathbf{\Gamma}\mathbf{C}^{i-1}) \\
u_i &= \frac{(i-1)(2i-3)(2i+2\nu-1)}{(i+2\nu-1)(2i+4\nu-1)(2i+2\nu-3)} \\
\omega_i &= 4\frac{(2i+2\nu-1)(i+\nu-1)}{(i+2\nu-1)(2i+4\nu-1)}.
\end{aligned}$$

The parameter ν is usually set to 1. The 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). This method can be proved to be faster than L2 boosting since the regularization parameter is the *square root* of the iteration number rather than the iteration number itself. In other words the ν-method can find in $\sqrt{t}$ steps the same solution found by L2 boosting after t iterations.

Iterated Tikhonov This method can be seen as a combination of Tikhonov regularization and L2 boosting where we set $\mathbf{C}^0 = 0$ and consider for $i = 0, \ldots, t-1$ the iteration $(\mathbf{\Gamma} + n\lambda I)\mathbf{C}^i = \mathbf{Y} + n\lambda \mathbf{C}^{i-1}$. The filter function is:

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

This methods is motivated by the desire to circumvent some of the limitations of Tikhonov regularization, namely a saturation effect that prevents exploiting the smoothness of the target function beyond a given critical value, see (Engl et al, 1996; Lo Gerfo et al, 2008) for further details.

Truncated Singular Values Decomposition. This method is akin to a projection onto the first principal components in a vector-valued setting. The number of components depends on the regularizing parameter. The filter function is defined by $G_\lambda(\sigma) = 1/\sigma$ if $\sigma \geq \lambda/n$ and 0 otherwise.

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 computational cost of tuning the regularization parameter. Some considerations along this line are given at the end of Sect. 4, whereas theoretical aspects are discussed in the next section.

3.4 Excess Risk for Spectral Regularization

The main result of this section is a finite sample bound on the excess risk that immediately leads to consistency. In order to prove such a result for the various algorithms in a unified framework 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).

We also need to make some preliminary assumptions. More precisely, the reproducing kernel is assumed to be bounded (3). The input space is a separable metric space (not necessarily compact). The output space is a bounded set in $\mathbb{R}^d$, that is $\sup_{y \in \mathcal{Y}} ||y||_d = M < \infty$. For the sake of simplicity we also assume that a minimizer of the expected risk on $\mathcal{H}$ exists and denote it with $f_{\mathcal{H}}$. Given the above assumptions, the definition of 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*
$$\sup_{0 < \sigma \leq \kappa^2} |\sigma g_\lambda(\sigma)| \leq D \tag{13}$$
- *There exists a constant B such that*
$$\sup_{0 < \sigma \leq \kappa^2} |g_\lambda(\sigma)| \leq \frac{B}{\lambda} \tag{14}$$
- *There exists a constant γ such that*
$$\sup_{0 < \sigma \leq \kappa^2} |1 - g_\lambda(\sigma)\sigma| \leq \gamma \tag{15}$$
- *There exists a constant $\overline{\nu} > 0$, namely the qualification of the filter g_λ such that*
$$\sup_{0 < \sigma \leq \kappa^2} |1 - g_\lambda(\sigma)\sigma|\sigma^\nu \leq \gamma_\nu \lambda^\nu, \qquad \forall 0 < \nu \leq \overline{\nu} \tag{16}$$

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

The above conditions are well known in the context of regularization for ill-posed problems, Roughly speaking, the first two conditions ensure the regularization operator induced by a filter to be bounded and with condition number controlled by the regularization parameter λ. The last two conditions are more technical and govern the approximation properties of each filter. We refer the interested reader to (Lo Gerfo et al, 2008; Bauer et al, 2007) for a more thorough discussion and further details.

Given the above definition we can state our main theorem.

Theorem 1. *Assume $\overline{\nu} \geq \frac{1}{2}$ and $\|f_{\mathcal{H}}\|_{\Gamma} \leq R$. Choose the regularization parameter $\lambda_n = \lambda(n)$ as*

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

so that when n goes to ∞, λ_n goes to zero. If we let $f_{\mathbf{z}} = f_{\mathbf{z}}^{\lambda_n}$, then with probability $1-\eta$

$$\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_{\mathcal{H}}) \leq \frac{C \log 4/\eta}{\sqrt{n}}, \tag{17}$$

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

The above result generalizes the analysis in (Bauer et al, 2007; Caponnetto and De Vito, 2006), to which we refer for the computation of the constants corresponding to each algorithm. We give the proof in the appendix and add three remarks. First, the above result immediately leads to consistency. Even when the expected risk does not achieve a minimum in $\mathcal{H}$, one can still show that there is a parameter choice ensuring convergence to $\inf_{f\in\mathcal{H}} \mathcal{E}(f)$ – see (Caponnetto, 2006). If the kernel is universal (Steinwart, 2002; Caponnetto et al, 2008), then universal consistency (Devroye et al, 1996) is ensured. Second, if we strengthen the assumptions on the problem we can obtain faster convergence rates. If $L_\Gamma f(s) = \int_{\mathcal{X}} \Gamma(x,s) f(s) d\rho_X(s)$ is the integral operator with kernel Γ, we can consider the assumption[2] $f_\rho = L_\Gamma^r u$ for some $u \in L^2(X,\rho)$ (the space of square integrable functions). In this case by choosing $\lambda_n = n^{-\frac{1}{2r+1}}$ we can replace the rate $n^{-1/2}$ with $n^{-\frac{2r}{2r+1}}$, which is optimal in a minimax sense (Caponnetto and De Vito, 2006). 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, namely adaptively. Indeed, this is the case since we can directly apply the results in (De Vito et al, 2008).

4 Kernels and Computational Aspects

In this section we discuss the computational properties of the various algorithms on the basis of the parameter strategy considered and the kernel used. Towards this end we begin by recalling some examples of kernels defining vector-valued RKHS and their connection to the regularizer choice.

4.1 Decomposable Kernels and Regularizers

A crucial practical question is which kernel to use in a given problem. 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, it is known that choosing an appropriate penalty function, a *regularizer*, corresponds to choosing a kernel function (Smola et al, 1998). 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 ones presented in the previous section. Also, this 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. The material in this section is largely drawn from previous results in (Evgeniou et al, 2005; Sheldon, 2008; Caponnetto et al, 2008).

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,

$$\Gamma_\omega(x,x') = K(x,x')(\omega \mathbf{1} + (1-\omega)\mathbf{I}) \tag{18}$$

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

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 explained 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:

$$\Gamma(x, x') = K(x, x')A \tag{19}$$

where K is a scalar kernel and A a positive semidefinite $d \times d$ 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 data.

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

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

with $c_i^t \in \mathbb{R}$. Each component is thus a different linear combination of the same coefficients $\{c_i\}_{i=1}^n$ and 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 component f^ℓ is independent to the others. The norm of the vector-valued function can also be expressed in terms of the coefficients c_i and the matrix A,

$$\begin{aligned} ||f||_\Gamma^2 &= \langle f, f\rangle_\Gamma = \sum_{ij} \langle c_i, \Gamma(x_i, x_j)c_j\rangle_\mathcal{Y} \\ &= \sum_{ij} \langle c_i, K(x_i, x_j)Ac_j\rangle_\mathcal{Y} \\ &= \sum_{ij} \sum_{\ell q} K(x_i, x_j)c_i^\ell A_{\ell q}c_j^q. \end{aligned}$$

Now for the considered kernels, the similarity between the components can be evaluated by their pairwise scalar products:

$$\langle f^\ell, f^q\rangle_K = \sum_{ij} \sum_{ts} K(x_i, x_j)A_{\ell t}c_i^t A_{qs}c_j^s. \tag{20}$$

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 (19). Then the norm of any function in the corresponding RKHS can be written as*

$$||f||_\Gamma^2 = \sum_{\ell,q=1}^{d} A_{\ell q}^\dagger \langle f^\ell, f^q\rangle_K, \tag{21}$$

where $A^\dagger$ is the pseudoinverse of A.

The above result immediately leads to a RKHS interpretation of many regularizers. We illustrate this recalling some examples.

Graph regularization. Following (Sheldon, 2008; Micchelli and Pontil, 2004), we can 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,

$$J(f) = \frac{1}{2} \sum_{\ell,q=1}^{d} ||f^\ell - f^q||_K^2 M_{\ell q} + \sum_{\ell=1}^{d} ||f^\ell||_K^2 M_{\ell\ell}. \tag{22}$$

The regularizer $J(f)$ can be rewritten as:

$$\sum_{\ell,q=1}^{d} \left(||f^\ell||_K^2 M_{\ell q} - \langle f^\ell, f^q \rangle_K M_{\ell q} \right) + \sum_{\ell=1}^{d} ||f^\ell||_K^2 M_{\ell\ell} = \sum_{\ell=1}^{d} ||f^\ell||_K^2 \sum_{q=1}^{d} (1 + \delta_{\ell q}) M_{\ell q} - \sum_{\ell,q=1}^{d} \langle f^\ell, f^q \rangle_K M_{\ell q} = \sum_{\ell,q=1}^{d} \langle f^\ell, f^q \rangle_K L_{\ell q} \tag{23}$$

where $L = D - M$, with $D_{\ell q} = \delta_{\ell q} \left(\sum_{h=1}^{d} M_{\ell h} + M_{\ell q} \right)$. Eq. (23) is of the form defined in Prop. 1, therefore the resulting kernel will be $\Gamma(x, x') = K(x, x')L^\dagger$, with $K(x, x')$ a scalar kernel to be chosen according to the problem at hand.

Output components clustering. Another example of regularizer, proposed in (Evgeniou et al, 2005), is based on the idea of grouping the components into r clusters and enforcing the components in each cluster to be similar. Following (Jacob et al, 2008), let us define the matrix E as the $d \times r$ matrix, where r is the number of clusters, such that $E_{\ell c} = 1$ if the component ℓ belongs to cluster c and 0 otherwise. Then we can compute the $d \times d$ matrix $M = E(E^T E)^{-1} E^T$ such that $M_{\ell q} = \frac{1}{m_c}$ if components ℓ and q belong to the same cluster c, and m_c is its cardinality, $M_{\ell q} = 0$ otherwise. Furthermore let $I(c)$ be the index set of the components that belong to cluster c. Then we can consider the following regularizer that forces components belonging to the same cluster to be close to each other:

$$J(f) = \epsilon_1 \sum_{c=1}^{r} \sum_{\ell \in I(c)} ||f^\ell - \overline{f}_c||_K^2 + \epsilon_2 \sum_{c=1}^{r} m_c ||\overline{f}_c||_K^2, \tag{24}$$

where $\overline{f}_c$ is the mean of the components in cluster c and ϵ_1, ϵ_2 are parameters balancing the two terms. Straightforward calculations show that the previous regularizer can be rewritten as $J(f) = \sum_{\ell q} G_{\ell q} \langle f^\ell, f^q \rangle_K$, where $G_{\ell q} = \epsilon_1 \delta_{\ell q} + (\epsilon_2 - \epsilon_1) M_{\ell q}$. Therefore the corresponding matrix valued kernel is $\Gamma(x, x') = K(x, x')G^\dagger$.

Common similarity. The simple matrix valued kernel (18), that imposes a common similarity between the output components, can be viewed as a particular regularizer. In fact a simple calculation shows that, letting $\gamma = \frac{1}{1-\omega+\omega d}$, the corresponding regularizer is

$$J(f) = \gamma \sum_{\ell=1}^{d} ||f^\ell||_K^2 + \gamma \frac{\omega d}{1-\omega} \sum_{\ell=1}^{d} ||f^\ell - \frac{1}{d} \sum_{q=1}^{d} f^q||_K^2. \tag{25}$$

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, $\overline{f} = \frac{1}{d} \sum_{q=1}^{d} f^q$.

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 via the SVR method, with ad-hoc matrix valued kernels based on matrix valued radial basis functions (RBF) (Narcowich and Ward, 1994). These kernels induce a similarity between the vector field components that depends on the input points, and therefore cannot be reduced to the form $\Gamma(x, x') = K(x, x')A$.

The divergence-free matrix valued kernels can be obtained as

$$\Gamma(x, x')_{df} = \Phi(x - x')_{df} = \Phi(u)_{df} = (\nabla \nabla^T - \nabla^T \nabla I)\phi(u) = H\phi(u) - tr(H\phi(u))I \ ,$$

where we defined $u = x - x'$, H the Hessian operator and ϕ a scalar RBF.

The columns of the matrix valued RBF Φ are divergence-free. In fact, computing the divergence of a linear combination of its columns, $\nabla^T(\Phi(u)_{df}c)$, with $c \in \mathbb{R}^d$, we get

$$\begin{aligned}\nabla^T(\Phi(u)_{df}c) &= \nabla^T(\nabla\nabla^T\phi(u))c - \nabla^T(\nabla^T\nabla\phi(u))c \\ &= (\nabla^T\nabla\nabla^T\phi(u))c - (\nabla^T\nabla^T\nabla\phi(u))c = 0\ ,\end{aligned}$$

where the last equality holds applying the product rule of the gradient, the fact that the coefficient vector c does not depend upon u and the equality $a^T a a^T = a^T a^T a$. Choosing a Gaussian RBF, we obtain the divergence-free kernel

$$\Gamma_{df}(x, x') = \frac{1}{\sigma^2} e^{-\frac{||x-x'||^2}{2\sigma^2}} A_{x,x'} \tag{26}$$

where

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

The curl-free matrix valued kernels are obtained as

$$\Gamma(x, x')_{cf} = \Phi(x - x')_{cf} = \Phi(u)_{cf} = -\nabla\nabla^T\phi(u) = -H\phi(u)\ ,$$

where ϕ is a scalar RBF.

It is easy to show that the columns of Φ_{cf} are curl-free. The j-th column of Φ_{cf} is given by $\Phi_{cf}e_j$, where e_j is the standard basis vector with a one in the j-th position. This gives us

$$\Phi_{cf}e_j = -\nabla\nabla^T\Phi_{cf}e_j = \nabla(-\nabla^T\Phi_{cf}e_j) = \nabla g\ ,$$

where $g = -\partial\phi/\partial x_j$. g is a scalar function and the curl of the gradient of a scalar function is always zero.

Choosing a Gaussian RBF, we obtain the following curl-free kernel

$$\Gamma_{cf}(x, x') = \frac{1}{\sigma^2} e^{-\frac{||x-x'||^2}{2\sigma^2}} \left(\mathbf{I} - \left(\frac{x-x'}{\sigma}\right)\left(\frac{x-x'}{\sigma}\right)^T\right)\ . \tag{27}$$

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 the divergence-free and curl-free parts separately (see (Macêdo and Castro, 2008) and the experiments in Sect. 7 for more details). The interested reader can refer to (Narcowich and Ward, 1994; Lowitzsch, 2005; Fuselier Jr, 2006) for further details on matrix valued RBF and the properties of divergence-free and curl-free kernels.

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 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 if $\mathcal{Y} \subseteq \mathbb{R}^d$) are introduced and their universality discussed. One such example is given by kernels defined by transformations. For the purpose of our discussion, let $\mathcal{Y} = \mathbb{R}^d$, $\mathcal{X}_0$ be a Hausdorff space and T_p be a map (not necessarily linear) from $\mathcal{X}$ to $\mathcal{X}_0$ for $p = \{1, \ldots, d\}$. 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 $x, x' \in \mathcal{X}$

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

An example of this type of kernel is used in (Vazquez and Walter, 2003), where $\mathcal{X} = \mathcal{X}_0 = \mathbb{R}$ and $T_p(x) = x + \tau_p$, with $\tau_p \in \mathbb{R}$. This kernel models "delays" between the components of the vector-valued function and can be used for system identification.

4.2 Eigen-decomposition for Matrix-valued Kernels

Before discussing complexity issues, we describe some specific properties of kernels of the form $\Gamma(x,y) = K(x,y)A$. 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 an easier way.

We start observing that if we denote with $u_1, \ldots, u_d$ the eigenvectors of A we can write the vector $\mathbf{C} = (c_1, \ldots, c_n)$, with $c_i \in \mathbb{R}^d$, as

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

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

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

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 $\mathbf{K}$ and with σ_j $(j = 1, \ldots, d)$ the eigenvalues of A, we have the following equalities

$$\begin{aligned}
\mathbf{C} &= g_\lambda(\mathbf{\Gamma})\mathbf{Y} \\
&= \sum_{j=1}^{d} g_\lambda(\mathbf{K} \otimes A)\tilde{y}^j \otimes u_j \\
&= \sum_{j=1}^{d} \sum_{i=1}^{n} g_\lambda(\sigma_j \lambda_i) \langle \tilde{y}^j, v_i \rangle v_i \otimes u_j \\
&= \sum_{j=1}^{d} g_\lambda(\sigma_j \mathbf{K})\tilde{y}^j \otimes u_j
\end{aligned}$$

Since the eigenvectors u_j are orthonormal, it follows that:

$$\tilde{c}^j = g_\lambda(\sigma_j \mathbf{K})\tilde{y}^j, \qquad \text{for all } j = 1, \ldots, d. \tag{28}$$

The above equation shows that in the new coordinate system 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

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

and we see that the scaling term is essentially changing the scale of the regularization parameter (and 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.

4.3 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 $O(N(nd)^3)$ complexity for Tikhonov regularization and $O(N(nd)^2)$ for iterative methods.

In the special case of kernels of the form $\Gamma(x, x') = K(x, x')A$ 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. The latter can be chosen as $2/\sigma_{max}$ 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.

5 Multi-category Classification as Learning a Vector-valued Function

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

Multiclass, 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 this section we show that approaches based on vector-valued learning are natural and help understanding multi-class problems. In particular, we show how spectral regularization methods for vector-valued learning can be used to build multi-classification rules that are Bayes consistent.

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, or all versus all (also called all pairs). 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 an interesting, exhaustive experimental comparison 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 (Zhang, 2004; Tewari and Bartlett, 2005). 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 section 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.

To this end, we need to fix some basic concepts and notation. In multi-category classification the examples belong to either one of d classes, that is we can set $Y = \{1, 2, \ldots, d\}$ and let $\rho(k|x)$, with $k = 1, \ldots, d$, denote the conditional probability for each class. A classifier is a function $c : \mathcal{X} \to Y$, assigning each input point to one of the d classes. The classification performance can be measured via the misclassification error

$$R(c) = \mathrm{P}\left[c(x) \neq y\right].$$

It is easy to check that the minimizer of the misclassification error is the Bayes rule, defined as

$$b(x) = \operatorname*{argmax}_{k=1,\ldots,d} \rho(k|x). \tag{29}$$

A standard approach for the binary case is based on viewing classification as a regression problem with binary values. Following this idea we might consider real-valued functions to fit the labels $Y = \{1, 2, \ldots, d\}$ but, we would force an unnatural ordering among the classes. Another possibility is to define a *coding*, that is a one-to-one map $C : Y \to \mathcal{Y}$ where $\mathcal{Y} = \{\bar{\ell}_1, \ldots, \bar{\ell}_d\}$ is a set of d distinct coding vectors $\bar{\ell}_k \in \mathbb{R}^d$ for $j = 1, \ldots, d$. For example $\bar{\ell}_1 = (1, 0, 0, \ldots, 0)$, $\bar{\ell}_2 = (0, 1, 0, \ldots, 0)$, ..., $\bar{\ell}_d = (0, 0, 0, \ldots, 1)$. Once we fix a coding we can use algorithms for vector regression to fit the data where the outputs are given by the coding. In practice the algorithm will return an estimator that takes values in the whole space $\mathbb{R}^d$, rather then in the set of coding vectors, and we need to define a classification rule. 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 squared loss, a simple calculation guides us through some of the above choices.

We use upper indexes to indicate vector components, so that the squared 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 one-to-one, the probability for each coding vector $\bar{\ell}_k$ is given by $\rho(k|x)$ for all $k = 1, \ldots, d$. The expected risk

$$\mathcal{E}(f) = \int_{\mathcal{X}\times Y} \|y - f(x)\|_d^2 d\rho(y|x) d\rho(x) = \int_{\mathcal{X}} \sum_{k=1}^d \|\bar{\ell}_k - f(x)\|_d^2 \rho(k|x) d\rho(x)$$

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

$$f_\rho(x) = (f_\rho^1(x), f_\rho^2(x) \ldots, f_\rho^d(x)) = \int_Y y d\rho(y|x) = \sum_{k=1}^d \bar{\ell}_k \rho(k|x).$$

Given a general coding

$$\bar{\ell}_1 = (a, b, \ldots, b),\ \bar{\ell}_2 = (b, a, \ldots, b), \ldots,\ \bar{\ell}_d = (b, b, \ldots, a), \quad a > b, \tag{30}$$

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

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

since $\sum_{k=1}^d \rho(k|x) = 1$. 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

$$b(x) = \operatorname*{argmax}_{j=1,\ldots,d} f_\rho^j(x). \tag{31}$$

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 (30). Third, once we obtained an estimator for the regression function, Equation (31) shows that the natural way to define a classification rule is to take the argmax of the components of the estimator.

For loss-functions other than the squared loss the above conclusions are not straightforward and a detailed discussion can be found in (Tewari and Bartlett, 2005; Zhang, 2004). Here we only note that one case where similar results are available is the variation of the hinge loss studied in (Lee et al, 2004): in this case the target function is the Bayes rule itself. It may be interesting to note that the results in (Lee et al, 2004) requires a sum-to-zero coding, e.g., $a = 1$ and $b = -1/(d-1)$, which is not needed in our setting. It is also interesting to see that multicategory SVM has a straightforward interpretation in terms of vector-valued regression. In fact, in our notation the 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.

It is clear that we are considering a reproducing kernel Hilbert space of vector valued functions with *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 squared loss we have the following problem

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

For a general coding of the form (30), 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.

In the case of the squared 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 Sect. 4.1 are a step towards this direction. In particular, for kernels of the form (19) the matrix A can be viewed as encoding the class correlations. The choice of A is therefore crucial. In certain problems the matrix A can be defined using a priori information. For example in the object recognition dataset Caltech-356 (Griffin et al, 2007), there are 256 object categories and a hand-made taxonomy relating the categories is available which can be exploited to design the matrix A. In general, empirically estimating the matrix A is much harder in multi-category classification than in vector-valued regression since the covariance structure of the coding vectors does not yield any useful information. We note that different strategies to exploit prior information in multi-class classification have been proposed in different, but not unrelated settings. In particular, we mention structured learning and error correcting code strategies (Dietterich and Bakiri, 1995; Szedmak et al, 2005; Tsochantaridis et al, 2005).

Remark 6. *In structured learning the empirical risk minimization approach is extended to a large class of problems where the outputs are structured objects. In particular, in the case of multi-category classification, the authors in (Tsochantaridis et al, 2005) propose to use a joint feature map on input and output which is reminding of the decomposable kernels discussed in Sect. 4.1. The analysis in (Szedmak et al, 2005) does not explicitly use the machinery of RKHS and it would be interesting to investigate more formally the connections with our study. Error bounds in the context of structured learning can be found in (McAllester, 2007).*

Remark 7. *Error correcting code strategies, see for example (Dietterich and Bakiri, 1995), differ from the approach we described here in the way in which the correlation among tasks is exploited. More precisely, instead of simply considering the argmax of the one-versus-all output, more sophisticated strategies are considered. This kind of approaches are interesting as they try to take advantage of the full information contained in the different binary classifiers. On the other hand, they are hard to compare to our study and, more generally, to analyze within the framework of statistical learning theory.*

The above discussion shows that, provided a coding of the form (30), we can use spectral regularization methods to solve multi-class problems and use (31) to obtain a classification rule. Also, Equation (31) and the fact that spectral regularization algorithms estimate the regression function, suggest that Bayes consistency can be achieved by spectral regularization methods. Indeed, this can be easily proved using the results in Sect. 3.4 and results in (Tewari and Bartlett, 2005) and (Zhang, 2004).

Theorem 2. *Assume $\overline{\nu} \geq \frac{1}{2}$ and $f_\rho \in \mathcal{H}$. Choose the regularization parameter $\lambda_n = \lambda(n)$ so that, when n goes to ∞, λ_n goes to zero and $\lambda_n \sqrt{n}$ goes to ∞. If we let $f_{\mathbf{z}} = f_{\mathbf{z}}^{\lambda_n}$ and $c_{\mathbf{z}} = \mathrm{argmax}_{j=1\ldots,d} f_{\mathbf{z}}^j$, then, for all $\varepsilon > 0$,*

$$\lim_{n \to \infty} \mathrm{P}\left[R(c_{\mathbf{z}}) - R(b) > \varepsilon \right] = 0, \tag{32}$$

where b is the Bayes rule (29).

We add three comments. First, the proof of the above result is given in the Appendix and is based on the bound given in Theorem 1 together with a so-called *comparison* result relating expected risk and misclassification error. More precisely, we use a result given in Corollary 26 in (Zhang, 2004) (see also (Tewari and Bartlett, 2005)) to show that for the squared loss

$$R(c) - R(b) \leq \psi(\mathcal{E}(f) - \mathcal{E}(f_\rho)),$$

where ψ is a decreasing function that goes to zero in the origin. 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 investigation 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).

6 Spectral Regularization in Multi-task Learning

In this section, we briefly discuss the use of spectral regularization methods for a general multi-task problem. As we mentioned in the introduction, the latter can be seen as a generalization of vector-valued learning where, in particular, each output coordinate might have samples of different cardinalities. Among many references, we mention the original paper by (Caruana, 1997), the works using regularization approaches (see references in the introduction and Sect. 4.1) and also Bayesian techniques using Gaussian processes (Bonilla et al, 2007; Chai et al, 2009).

In the following we use the notation introduced in Remark 1. 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 3 turns out to be useful, since it allows to work component-wise.

Recall that according to Remark 3 we can view vector-valued RKHS as defined by a (joint) kernel $Q : (X, \Pi) \times (X, \Pi) \to \mathbb{R}$, where $\Pi = 1, \ldots d$ is the index set of the output components. A function in this RKHS is

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

with norm

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

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 to write the empirical error

$$\sum_{j=1}^{d} \frac{1}{n_j} \sum_{i=1}^{n_j} (y_i^j - f^j(x_i))^2$$

as

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

where we consider a training set $(x_1, y_1, t_1), \ldots, (x_{n_d}, y_{n_d}, t_{n_d})$ with $n_d = \sum_{j=1}^{d} n_j$.

The representer theorem ensures that the solution of 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 paper 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 Sect. 3.3.

We conclude this section observing that in general, differently to vector-valued regression, the matrix $\mathbf{\Gamma}$ is not a block matrix. In particular, in the case 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 reduce the complexity of the problem using the technique described in the end of Sect. 4.3. Therefore iterative methods might be considerably more efficient than Tikhonov regularization, as we will show with some experiments in the next section.

7 Empirical Analysis

In this section we present some empirical analysis using spectral regularization algorithms. We first consider an academic example aimed at showing a computational comparison of the various spectral filters while illustrating the difference between multi-task and vector-valued learning. Secondly, we present some artificial examples of $2D$ vector fields for which our approach outperforms regressing on each component independently with the same filter function. On these fields, we also compare the proposed approach with a state-of-the-art sparsity-enforcing method proposed by (Obozinski et al, 2007) and discuss its drawbacks. Finally, we consider a real-world case where our methods perform faster than standard Tikhonov regularization, while achieving a performance comparable to the best in the literature. Note that 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 Sect. 4.1, we show how this problem can be effectively solved.

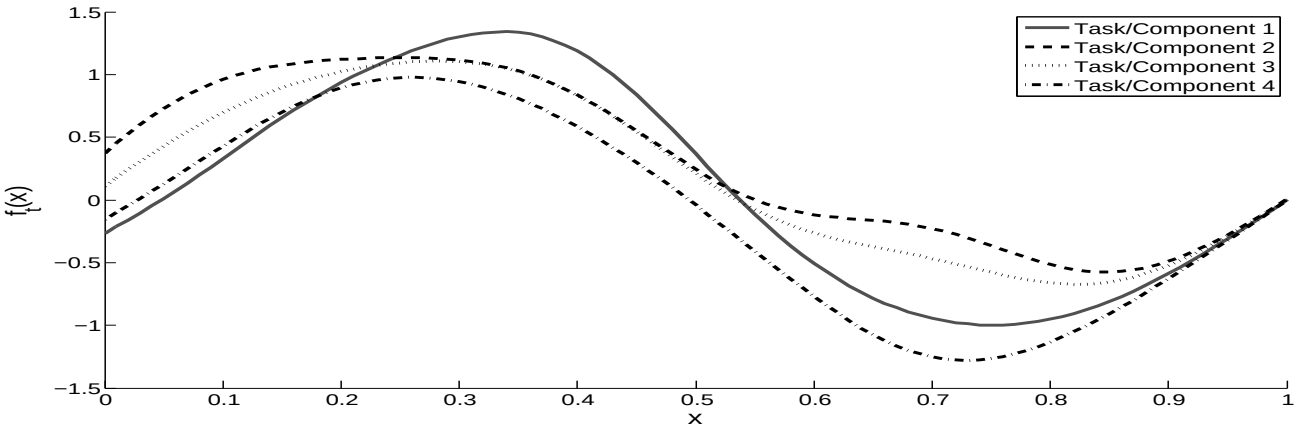

Figure 2: The four tasks/components (before being affected by Gaussian noise of variance 0.01) used to compare multi-task and vector-valued learning. The tasks are generated perturbating the common task (a sine function) with three Gaussians of width 0.1, centered in $x_1 = 0.05$, $x_2 = 0.4$ and $x_3 = 0.7$. The Gaussians are multiplied with task specific coefficients.

7.1 Simulated Data

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 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 a weighted sum of three Gaussians of width $\sigma = 0.1$ centered at $x_1 = 0.05$, $x_2 = 0.4$ and $x_3 = 0.7$. We have designed the task-specific weights of the perturbation in order to yield tasks that are still strongly related by the common target function, but also present local differences, as shown in Figure 2. It may appear that the tasks are simply shifted versions of the common sine function and that an approach based on computing the sample covariance might be able to estimate the phase differences. This is indeed not the case, since the perturbations added to each task are local and defined by Gaussian functions. Notwithstanding the simplicity of this example, we believe it is illustrative of the different behaviors of the multi-task and vector-valued settings and it allows us to compare the computational properties of three spectral filters in a controlled setting.

Since the performance of the spectral filters is very similar, we chose the Accelerated L2 Boosting (see Sect. 3.3, in the following referred to as ν-method) to illustrate the typical behavior. In the multi-task case each task is sampled in different input points, whereas in the vector-valued case the input points are the same for all the components. We used the kernel (18) that imposes a common similarity among all components, adopting a Gaussian kernel for its scalar part. 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.

We show the results for multi-task learning case in Figure 3 (left panel), where we compare the error obtained with the matrix valued kernel and the error of learning each task independently with a Gaussian kernel of the same width. We observe that exploiting 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 3 (right panel), where we see that there is no gain in using a non-diagonal kernel.

In Figure 4 (left panel) we report the time required to select the optimal regularization parameter on the val-

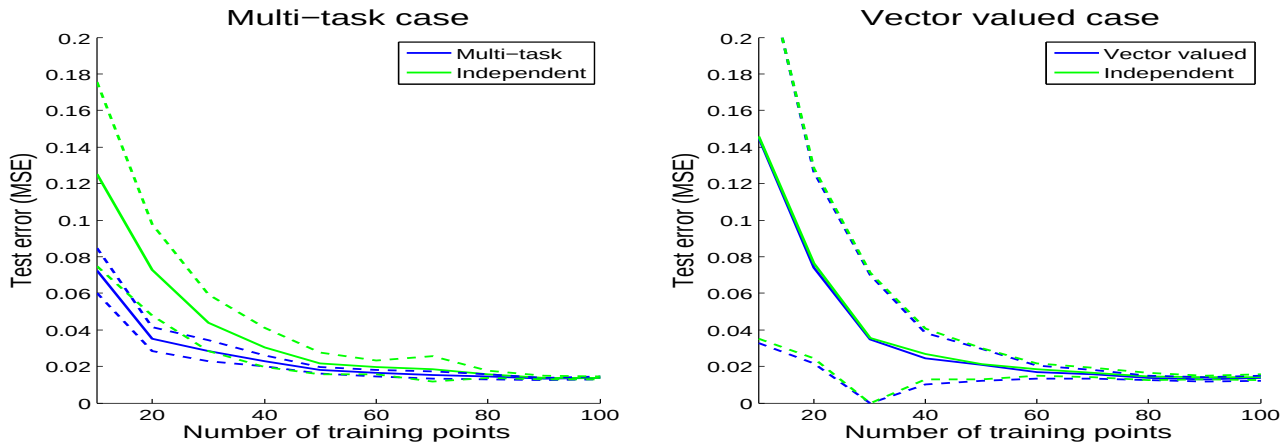

Figure 3: Results for the multi-task case (left) and for the vector-valued case (right) using the ν-method with a maximum of 200 iterations. Solid lines represent average test error, while dotted lines show the average test error plus/minus one standard deviation of the corresponding error. The test 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.

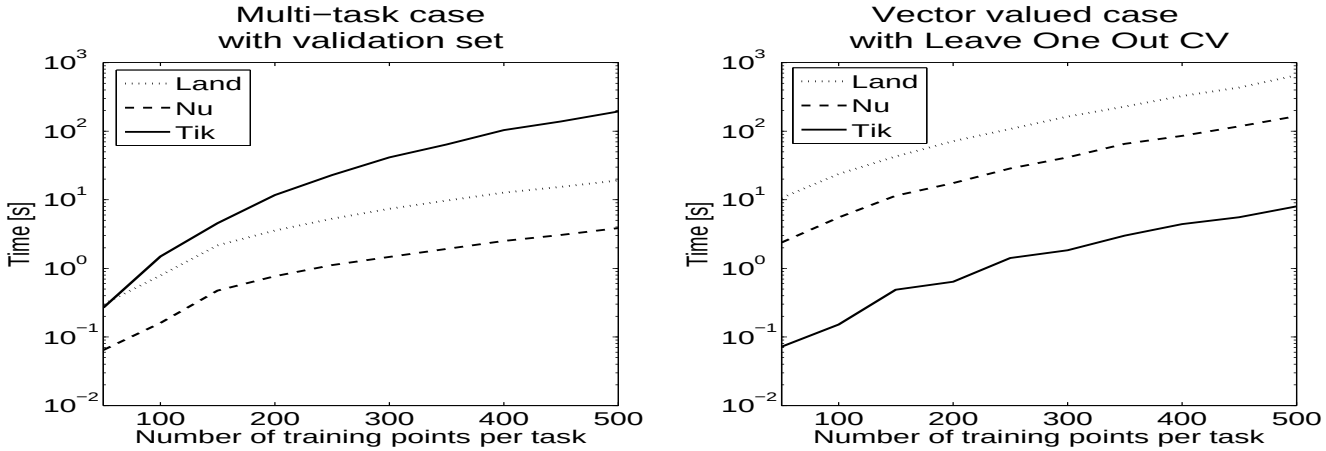

Figure 4: Time needed to select the best regularization parameter for different algorithms and settings. In the left panel the times required to select the regularization parameter for the multi-task setting with respect to the number of training examples are reported. The regularization parameter is 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 Sect.4.2 and the closed form solution to compute the LOO errors for Tikhonov. The range of the parameters evaluated is 25 values for Tikhonov and a maximum of 1000 iterations for Landweber and 200 iterations for the ν-method. The computations were performed with MATLAB on a notebook with 2GB of RAM and a 2GHz Intel Core 2 Duo Processor.

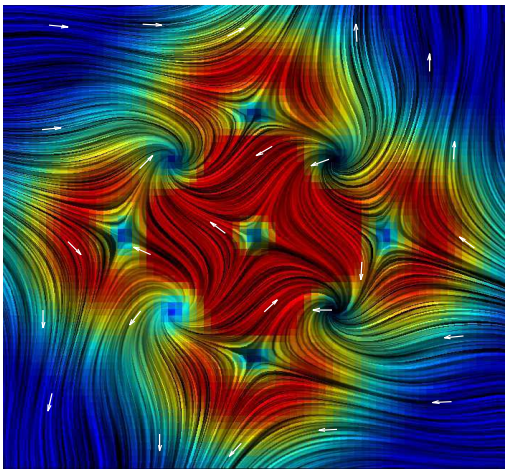

Figure 5: Visualization of the first artificial 2-dimensional vector field for $\gamma = 0.5$

idation set in the multi-task case. The vector-valued case presented the same behavior (graph not shown). The algorithms are Tikhonov with 25 regularization parameter values, Landweber 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 Sect. 4.3, the ν-method is outperforming the other methods. In Figure 4 (right panel) 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 Sect. 4.2 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 entire regularization path for each LOO loop.

2D Vector Field - 1. The following set of simulations are aimed at showing the advantages of using the divergence and curl-free kernels, (26) and (27) respectively, for the estimation of a general 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 $0.45\mathbf{I}$ ($\mathbf{I}$ is the 2×2 identity matrix). By computing the gradient of the scalar field, we obtain an irrotational (curl-free) field. The vector field perpendicular to the latter (computed applying a $\pi/2$ rotation) is a solenoidal (divergence-free) field. We consider a convex combination of these two vector fields, controlled by a parameter γ. One instance of the resulting field, for which $\gamma = 0.5$, is shown in Fig. 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.

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 200 and for each cardinality of the training set, the training and prediction process is repeated 10 times with different randomizations 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 $v \to \tilde{v} = \frac{1}{||(v^1, v^2, 1)||}(v^1, v^2, 1)$.

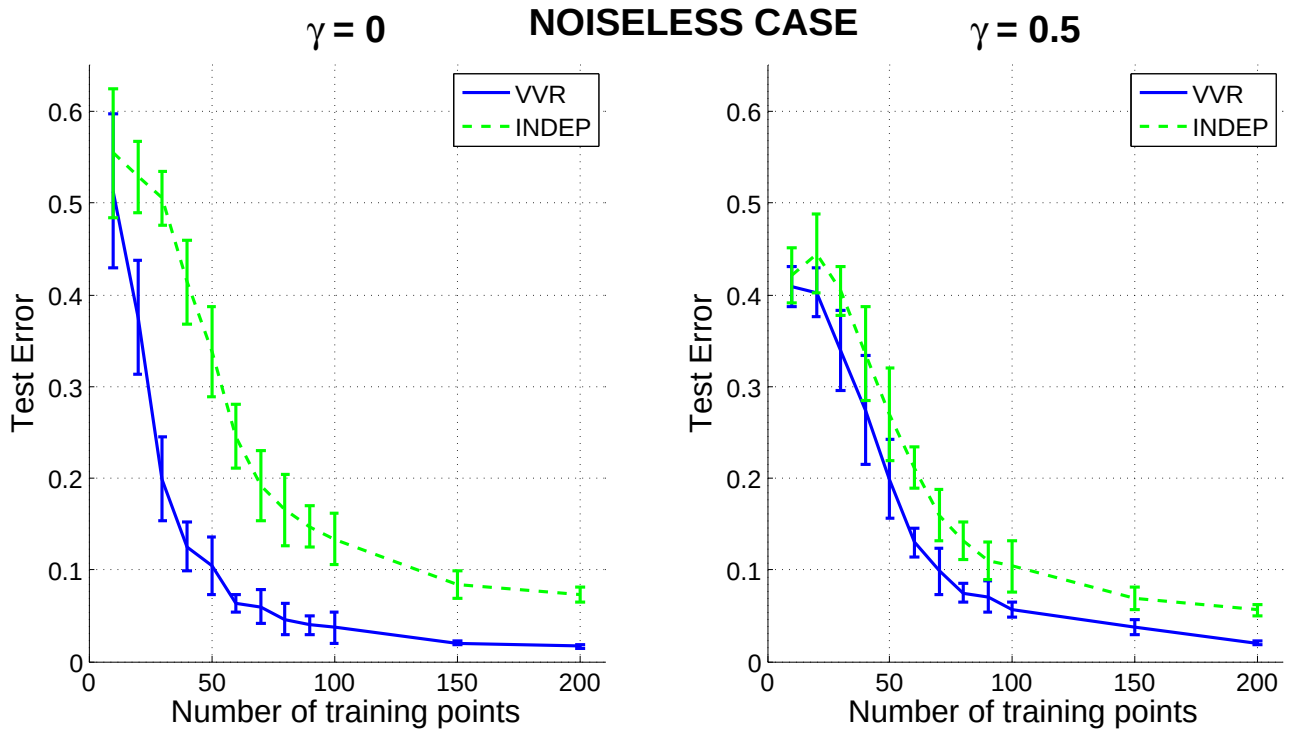

Figure 6: Vector field 1 - noiseless case. Test errors for the proposed vector-valued approach and for learning each component of the field independently as a function of the number of training points used for learning. Solid lines represent average test error, while dotted lines show the average test error plus/minus one standard deviation of the corresponding error. The test error is evaluated according to the error measure (33).

The error measure is then

$$err = \arccos(\tilde{v}_e \cdot \tilde{v}_o). \tag{33}$$

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 Fig. 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 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 quite large. The average value of the estimated parameter $\tilde{\gamma}$ 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, see Fig.7.

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 case. The results are reported in Fig. 8 and indicate that also in the presence of noise the proposed approach consistently outperforms regressing on each component independently.

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 satisfy the hypotheses of Helmholtz decomposition of a vector field.

2D Vector Field - 2. 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

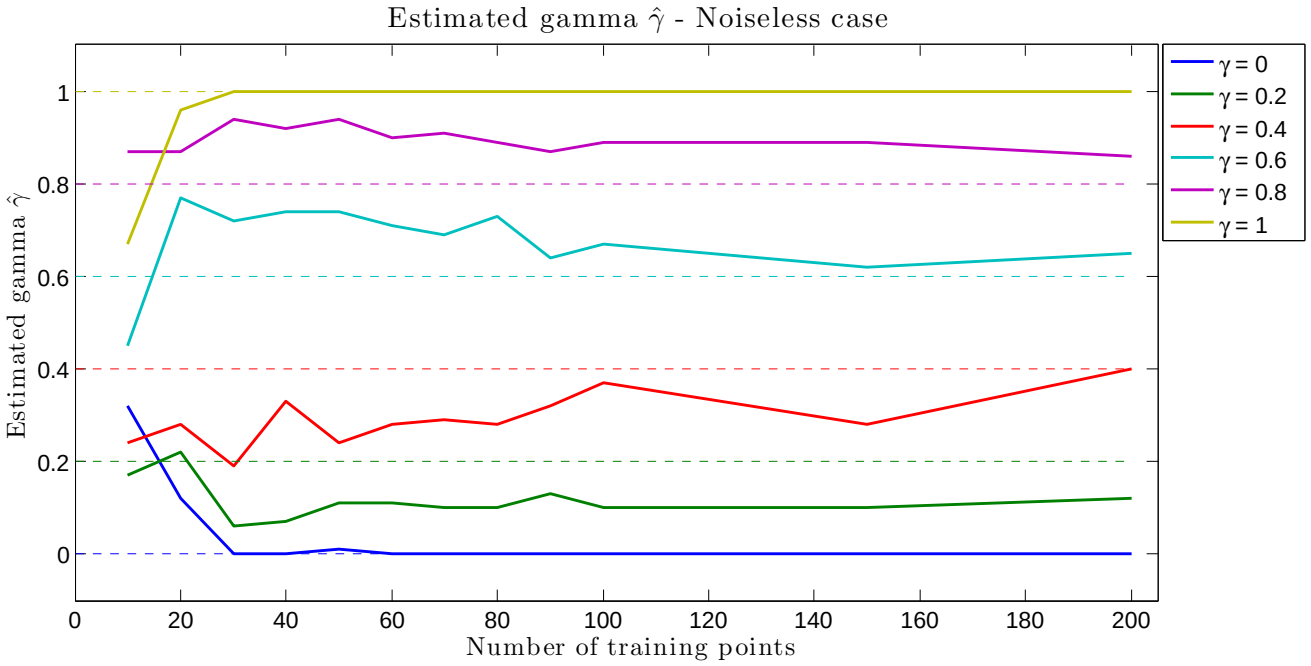

Figure 7: Vector field 1 - noiseless case. The solid lines represent the averages of the estimated kernel parameter $\hat{\gamma}$ that governs the balance between the divergence-free and the curl-free matrix valued kernels. The dotted lines represent the values of the parameter γ that was used to design the vector field. The learning algorithm estimates this values correctly, allowing to separately reconstruct the irrotational and solenoidal parts of the field.

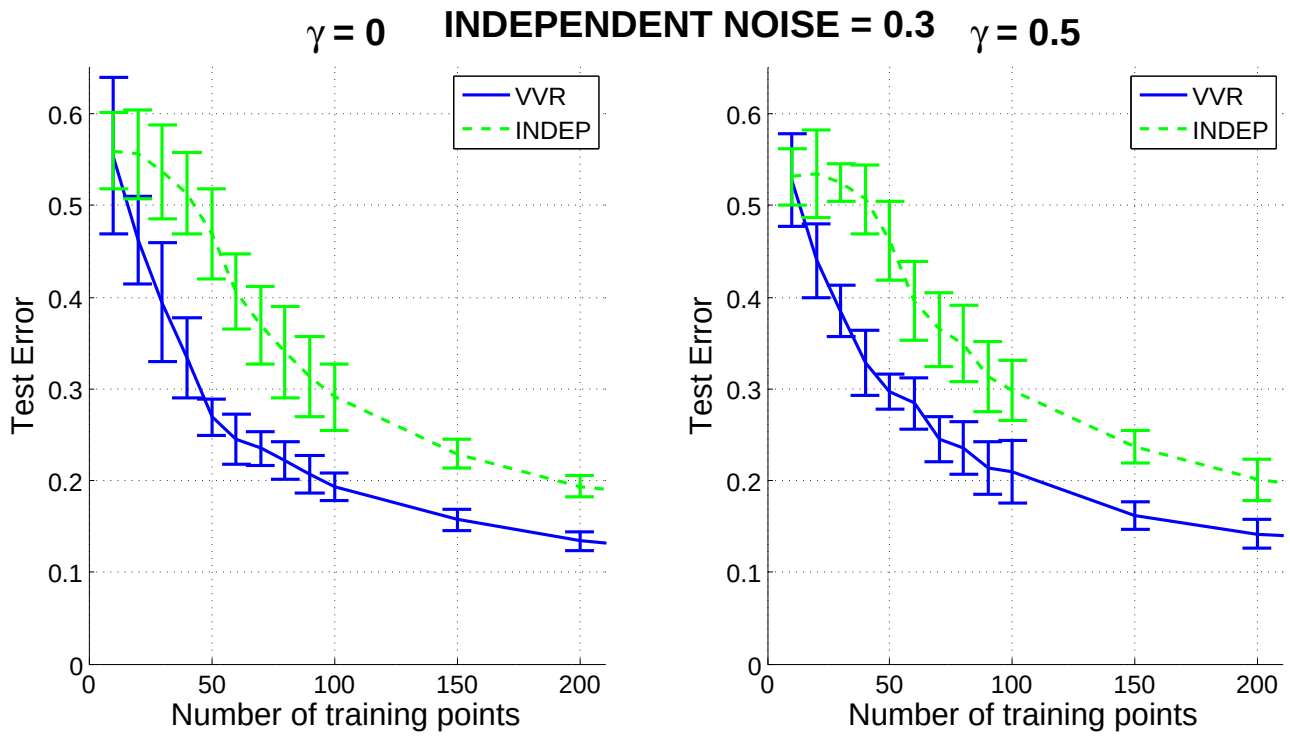

Figure 8: Vector field 1 - noise with standard deviation 0.3. Test errors for the proposed vector-valued approach and for learning each component of the field independently as a function of the number of training points used for learning. Solid lines represent average test error, while dotted lines show the average test error plus/minus one standard deviation of the corresponding error. The test error is evaluated according to the error measure (33).

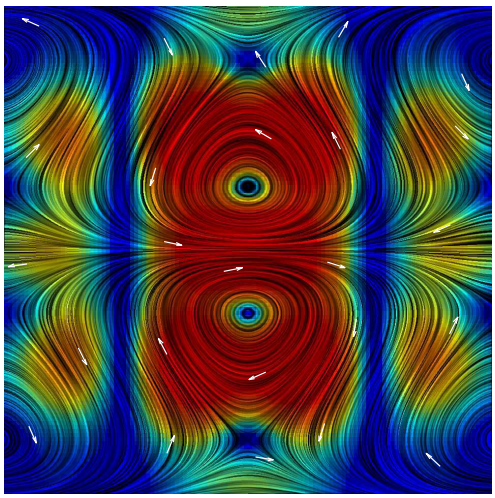

Figure 9: Visualization of the second artificial vector field without noise.

a divergence-free part and a curl-free part. 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 it is indeed possible to obtain a better estimate using these kernels when the vector field satisfies the assumptions of the Helmholtz theorem. Furthermore, we compare our approach with a state-of-the-art sparsity-enforcing method (Obozinski et al, 2007) and show that the latter is computationally much slower and critically depends on the choice of the feature map.

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

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

In order to enforce the decay at infinity the field is multiplied with a Gaussian function centered at the origin and of width 1.2. The field without noise is shown in Fig. 9.

We followed an experimental protocol similar to the one adopted for the previous artificial experiment. 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 model parameters are selected on a validation set of the same size of the training set, instead of performing the costly 5-fold cross-validation. Our approach consists in using the ν-method with a convex combination of the divergence-free and curl-free kernels, (26) and (27) respectively, controlled by a parameter γ, which is selected on the validation set alongside the optimal number of iterations. For the weight balancing the two kernels, we explored 11 values, equally spaced between 0 and 1, and we set the maximum number of iterations to 700, which was also used for regressing on each field component independently. We set the width of the Gaussian part of the matrix-valued kernels and the width of the Gaussian kernel for scalar regression to 0.8.

For comparison, we use the algorithm proposed by Mosci et al (2008) for minimizing the functional of the sparsity-enforcing method of Obozinski et al (2007). The algorithm considers a linear multi-task model and performs a two-step procedure. The first step consists in the selection of features uniformly across tasks, which is followed by a regularized least squares step for optimally estimating the coefficients for the selected features. The algorithm depends on two regularization parameters, τ and λ. The first weighs the ℓ_1 penalty on the norms of the coefficient vectors for each task and is responsible for obtaining sparse solutions. The second parameter is the ℓ_2 penalty on the coefficients of the regularized least squares step on the selected features. Both these parameters

were selected on the validation set among a geometric series of 30 values between 10^{-8} and 1. Since the vector field is obviously non-linear, we consider two feature maps from $\mathbb{R}^2$ to a higher dimensional Hilbert space, where we can treat the estimation problem as linear. These feature maps are based on dictionaries of basis functions.

The first dictionary contains basis vector fields with null-divergence or null-curl, centered on the nodes of a grid of $L = 17 \times 17$ lines spaced $\Delta = 0.25$ in either direction. Following (Mussa-Ivaldi, 1992), the curl-free field basis are obtained as the gradient of Gaussian potential functions centered on the nodes of the grid, in our case, $\phi(x, x_j) = -2(x - x_j)G(x, x_j)/\sigma^2$, where $G(x, x_j) = \exp(-||x - x_i||^2/\sigma^2)$. To ensure a significant overlap between neighboring fields, we set $\sigma^2 = 2\Delta$. The divergence-free basis, $\varphi(x, x_j)$ are obtained from the curl-free ones by a $\pi/2$ rotation, so that $\varphi^1(x, x_j) = -\phi^2(x, x_j)$ and $\varphi^2(x, x_j) = \phi^1(x, x_j)$. In (Mussa-Ivaldi, 1992), the estimated vector field is a linear combination of the field bases, $f(x) = \sum_{j=1}^{L} c_j \phi(x, x_j) + \sum_{j=1}^{L} d_j \varphi(x, x_j)$, so that each component f^t of the field depends on the same coefficients c_j and d_j. Conversely, we allow each component to depend on a different set of coefficients c_j^t and d_j^t

$$f^t(x) = \sum_{j=1}^{L} c_j^t \phi^t(x, x_j) + \sum_{j=1}^{L} d_j^t \varphi^t(x, x_j) \qquad t = 1, 2 \; .$$

This approach no longer permits to consider the vector field as the linear combination of field bases, because we are in fact adopting different scalar bases for each component. In other words, each task is given by a linear model on a different set of features. However, since the sparsity pattern is enforced to be the same for each task, the method discards entire bases from the expression of the reconstructed field. Obviously, it is no longer possible to decompose the estimated field in its divergence-free and curl-free parts.

The second dictionary we consider is the one proposed and used in (Haufe et al, 2009) for the estimation of electric currents in the brain from scattered EEG/MEG measurements. In this case, the vector field is modeled as a combination of field bases, $\mathbf{c}_j$ (to be estimated), with weights given by spherical Gaussians $b_{j,s}(x)$ centered on L points x_j and characterized by S widths σ_s,

$$f(x) = \sum_{j=1}^{L} \sum_{s=1}^{S} \mathbf{c}_{j,s} b_{j,s}(x) \; .$$

We can consider the Gaussians as determining feature map $\phi : \mathbb{R}^2 \to \mathbb{R}^{LS}$,

$$\phi(x) = [b_{1,1}(x)\; b_{1,2}(x) \ldots b_{1,S}(x) \ldots b_{L,1}(x) \ldots b_{L,S}(x)]^T \; ,$$

allowing to write the field as $f(x) = C\phi(x)$ where $C = [\mathbf{c}_{1,1}\; \mathbf{c}_{1,2} \ldots \mathbf{c}_{1,S} \ldots \mathbf{c}_{L,1} \ldots \mathbf{c}_{L,S}]$ is the coefficient matrix. We computed the spherical Gaussians on the same grid of 17×17 points used for the previous dictionary, and chose four different values of standard deviation, $\sigma_s = 0.2 \times 2^{s-1}, s = 1, \ldots, 4$. All these choices were made arbitrarily balancing the number and locality of the basis functions. In order to keep things simple, we kept the dictionary fixed as we varied the number of examples available for training and model selection. One could argue that a data-driven dictionary could allow for increased accuracy, but this analysis was beyond the scope of our experimental assessment.

In Fig. 10 (solid line) we report the test errors obtained using the proposed ν-method spectral filter with the convex combination of divergence-free and curl-free matrix valued kernels. The dotted line shows the test errors achieved by regressing on each component of the field independently, with the same spectral filter and with a Gaussian kernel of the same width used for the matrix valued kernels. It is evident that for estimating this general vector field, a vector-valued approach is still advantageous, even though the gain in performance deteriorates with the amount of noise. In fact, the noise disrupts the smoothness of the field, which no longer can be exactly decomposed as the sum of a divergence-free and a curl-free part. Computationally, the ν-method applied to vector-valued regression is just slightly more expensive than for scalar regression, as shown in Fig. 12.

In Fig. 11 we compare the test errors of the proposed vector-valued approach with the sparsity-enforcing multi-task method using the two dictionaries described above. We observe that when we use a dictionary consisting of field bases with null-divergence or null-curl, we obtain similar results as using the corresponding matrix valued

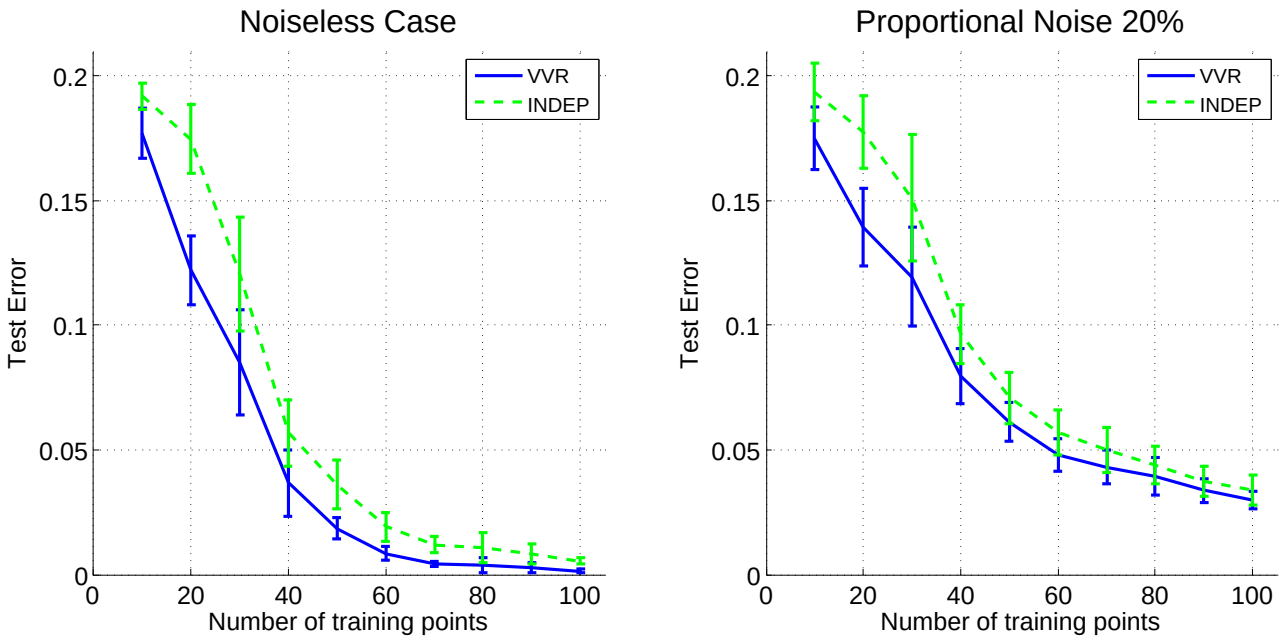

Figure 10: Vector field 2. Test errors for the proposed vector-valued approach and for learning each component of the field independently with the ν-method spectral filter, in the noiseless case (left) and when the standard deviation of the noise is equal to 20% of the field magnitude (right). The test error is evaluated according to the error measure (33).

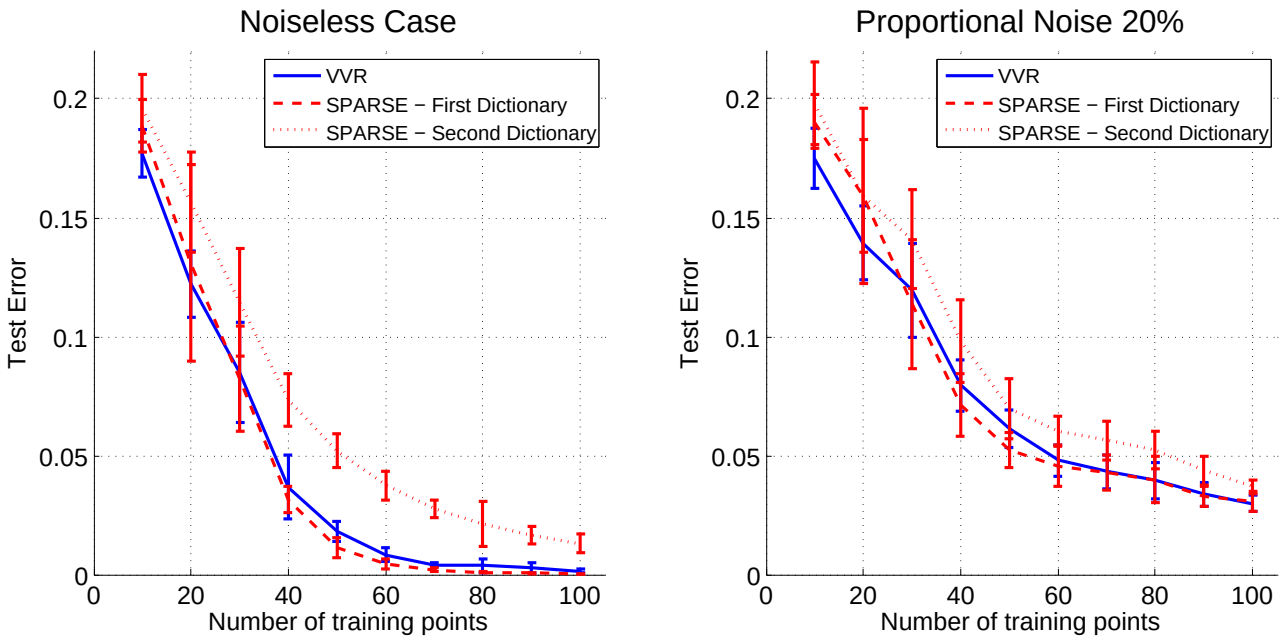

Figure 11: Vector field 2. Test errors for the proposed vector-valued approach and for the sparsity-enforcing method (Obozinski et al, 2007) in the noiseless case (left) and when the standard deviation of the noise is equal to 20% of the field magnitude (right). The first dictionary is the one proposed in (Mussa-Ivaldi, 1992), while the second is adapted from (Haufe et al, 2009). The test error is evaluated according to the error measure (33).

kernels. On the other hand, if a more general dictionary is used, the results are slightly worse and this may be due to a more critical dependence on the tuning of the dictionary parameters, e.g., number of nodes and standard deviations of the Gaussians.

Fig. 12 reports the computation times for training and model selection for all the different methods assessed for each replicate of the experiment. We observe that the proposed vector-valued approach using the ν-method spectral filter is significantly faster than the sparsity-enforcing multi-task method and comparable to regressing on each component independently. The differences in computation time between the two dictionaries can be partly explained by the different size of the feature maps, which in the second case is twice as big, since we consider $L \times S = 17 \times 17 \times 4 = 1156$ Gaussians, instead of L divergence-free and L curl-free bases.

7.2 Real Data

School data. This dataset from the Inner London Education Authority[3] 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. Hence, there are 139 tasks, 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 values 1, 2 or 3) and ethnic group (among 13 possible values). Following the literature, we converted the categorical attributes using one binary variable

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

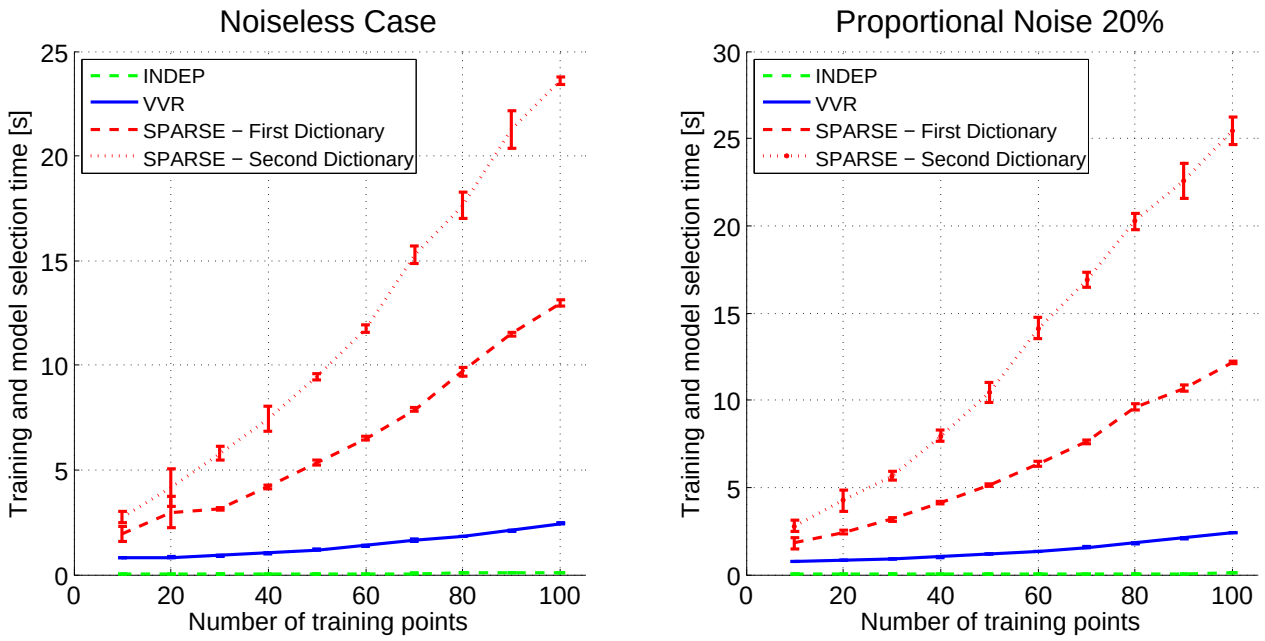

Figure 12: Vector field 2. Computation time for the proposed vector-valued approach, for the multi-task feature selection method and for learning each component of the field independently in the noiseless case. The computations were performed with MATLAB on a notebook with 2GB of RAM and a 2GHz Intel Core 2 Duo Processor.

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 direct Tikhonov algorithms using the simple common similarity kernel (18).

We randomly selected only 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 regularizing parameter and the value of the parameter ω for the kernel (18). 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 of the k nearest neighbors to each training point, where k is set to be 20% of the cardinality of the training set. We repeat this procedure ten times, with different random sampling of students from each school, to evaluate the stability of the error estimates obtained with each filter.

In Table 1 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 since these are the same for all algorithms). The range of the parameter ω is $[0, 1]$ and was sampled at 0.1 steps. The three algorithms perform consistently and improve on the results reported in (Argyriou et al, 2008a) - which obtain a performance of $26.4\% \pm 1.9\%$ - despite being trained only on 20% of the available data, plus an additional 20% for parameter selection. 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 almost two 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 assess-

ment, we opted for a maximum of 3000 iterations while for the ν-method a maximum of only 150 iterations. For Tikhonov, we choose 30 values sampled geometrically in the interval $[10^{-5}, 10^{-2}]$ and we performed a Singular Value Decomposition of the kernel matrix to more efficiently compute the regularized inverse with different regularization parameters.

Table 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 with MATLAB on a notebook with 2GB of RAM and a 2GHz Intel Core 2 Duo Processor.

Algorithm	**Performance**	**Model Selection Time [s]**
ν-method	$31 \pm 3\%$	106 ± 4
Landweber	$32 \pm 4\%$	2015 ± 45
Tikhonov	$32 \pm 3\%$	5900 ± 70

8 Conclusion

In this paper we studied the problem of learning vector-valued functions using a class of regularized kernel methods called spectral regularization. Tikhonov regularization and (vector-valued) L2 boosting are examples of methods falling in our framework. Computational issues comparing the implementations of the different algorithms on the basis of the kernel are discussed. Some of the algorithms, in particular iterative methods, provide interesting computational alternatives to Tikhonov regularization and in the experiments were shown to be much faster. A finite sample bound for all the methods was proven in a unified framework that shows their different theoretical properties. Finally, we analyzed the problem of multi-class classification from a vector learning perspective, discussing the role played by the coding strategy and Bayes consistency.

One outcome from the experiments is that the kernels proposed so far seem to be interesting in the context of multi-task regularization and for vector fields that satisfy the assumptions of the Helmholtz theorem, but potentially unable to capture the functional relations describing real vector-valued functions.

Future work will focus on the problem of defining new kernels for learning vector-valued functions and their role in exploiting the correlations among classes in multi-category classification.

Acknowledgments
We would like to thank Ernesto De Vito for many useful discussions. This work has been partially supported by the EU Integrated Project Health-e-Child IST-2004-027749.

Appendix: Proofs

In this section we give the proofs of the results in Sect. 3.4. Towards this end we first recall the definitions of some operators based on the kernel.

Appendix A: Kernel Matrix and Extension Operators

For any $\mathbf{x} \in \mathcal{X}^n$, we introduce the sampling operator $S_{\mathbf{x}} : \mathcal{H} \to \mathbb{R}^{nd}$ defined by $(S_{\mathbf{x}} f) = (f(x_1), \ldots, f(x_n))$ whose adjoint $S^*_{\mathbf{x}} : \mathbb{R}^{nd} \to \mathcal{H}$ is

$$S^*_{\mathbf{x}} = \frac{1}{n} \sum_{i=1}^{n} \Gamma_{x_i}.$$

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), it can be written as $\mathbf{\Gamma} = S_{\mathbf{x}} S^*_{\mathbf{x}}$.

If we define $T_x = \Gamma_x \Gamma_x^*$ and $T_{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^n T_{x_i} = S_{\mathbf{x}}^* S_{\mathbf{x}}$, then the operator $T_{\mathbf{x}} : \mathcal{H} \to \mathcal{H}$ can be seen as a natural out of sample extension of the kernel matrix since

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

Indeed, $T_{\mathbf{x}}$ and $\mathbf{\Gamma}$ are positive Hilbert-Schmidt operators with the same eigenvalues (Caponnetto and De Vito, 2006). The corresponding eigenfunctions v and eigenvectors $u = (u^1, \ldots, u^n)$ (associated to some eigenvalue σ) are related by the equations

$$v = \frac{1}{\sigma} S_{\mathbf{x}}^* u = \frac{1}{n\sigma}\sum_{i=1}^n \Gamma_{x_i} u^i, \qquad u = S_{\mathbf{x}} v.$$

The operator $T_{\mathbf{x}}$ can be viewed as a discretized version of $T = \int_{\mathcal{X}} T_x d\rho_{\mathcal{X}}(x)$, which is a positive and Hilbert-Schmidt operator. We conclude noting that

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

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

Appendix B: Proofs

Before proceeding to the proofs, we need some 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. It is easy to see that $f_{\mathbf{z}}^{\lambda} = S_{\mathbf{x}}^* g_\lambda(\mathbf{\Gamma})\mathbf{Y} = S_{\mathbf{x}}^* g_\lambda(S_{\mathbf{x}} S_{\mathbf{x}}^*)\mathbf{Y}$. Then, recalling the singular value decomposition $S_{\mathbf{x}} = U D_{\mathbf{x}} V^*$ (consequently $S_{\mathbf{x}}^* = V D_{\mathbf{x}}^* U^*$) and the spectral property of the filters $g_\lambda(S_{\mathbf{x}}) = U g_\lambda(D_{\mathbf{x}}) V^*$, we have that:

$$\begin{aligned}
S_{\mathbf{x}}^* g_\lambda(S_{\mathbf{x}} S_{\mathbf{x}}^*)\mathbf{Y} &= V D_{\mathbf{x}}^* U^* g_\lambda(U D_{\mathbf{x}} V^* V D_{\mathbf{x}}^* U^*)\mathbf{Y} \\
&= V D_{\mathbf{x}}^* g_\lambda(D_{\mathbf{x}} D_{\mathbf{x}}^*) U^* \mathbf{Y} \\
&= V g_\lambda(D_{\mathbf{x}}^* D_{\mathbf{x}}) D_{\mathbf{x}}^* U^* \mathbf{Y} \\
&= g_\lambda(S_{\mathbf{x}}^* S_{\mathbf{x}}) S_{\mathbf{x}}^* \mathbf{Y} \\
&= g_\lambda(T_{\mathbf{x}}) h_{\mathbf{z}},
\end{aligned}$$

where $D_{\mathbf{x}}^* g_\lambda(D_{\mathbf{x}} D_{\mathbf{x}}^*) = g_\lambda(D_{\mathbf{x}}^* D_{\mathbf{x}}) D_{\mathbf{x}}^*$ since $D_{\mathbf{x}}$ and $D_{\mathbf{x}}^*$ are both multiplicative operators. □ □

Let us introduce the operator $h = T_{\mathbf{x}} f_{\mathcal{H}}$ and recall the following lemma from (Caponnetto and De Vito, 2006).

Lemma 2. *Let* $M = sup_{y \in \mathcal{Y}} ||y||_d$ *and* $||f_{\mathcal{H}}||_\Gamma \leq R$. *For* $0 < \eta \leq 1$ *and* $n \in \mathbb{N}$ *let*

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

with

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

then we have that

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

We are ready to state the following theorem.

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

$$\lambda \geq \frac{1}{\sqrt{n}} 2\sqrt{2}\kappa^2 \log\frac{4}{\eta}. \tag{34}$$

Moreover, we assume that $f_{\mathcal{H}} \in \mathcal{H}$ and $||f_{\mathcal{H}}||_\Gamma \leq R$. Then with probability at least $1 - \eta$ we have

$$\mathcal{E}(f_{\mathbf{z}}^\lambda) - \mathcal{E}(f_{\mathcal{H}}) \leq 2\left((\gamma + \gamma_{\frac{1}{2}})^2 \lambda R^2 + \frac{C}{\lambda n}\right), \tag{35}$$

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

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

$$\mathcal{E}(f) - \mathcal{E}(f_{\mathcal{H}}) = \|\sqrt{T}(f - f_{\mathcal{H}})\|_\Gamma^2, \tag{36}$$

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

We assume throughout that $\mathbf{z} \in G_\eta$ as given in the above lemma so that the above inequalities holds true with probability at least $1 - \eta$ with $0 < \eta \leq 1$. We consider the following error decomposition:

$$\begin{aligned} ||\sqrt{T}(f_{\mathbf{z}}^\lambda - f_{\mathcal{H}})||_\Gamma^2 \leq \\ 2||\sqrt{T}(f_{\mathbf{z}}^\lambda - f^\lambda)||_\Gamma^2 + 2||\sqrt{T}(f^\lambda - f_{\mathcal{H}})||_\Gamma^2, \end{aligned} \tag{37}$$

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

$$\begin{aligned} \sqrt{T}(f_{\mathbf{z}}^\lambda - f^\lambda) = \sqrt{T} g_\lambda(T_{\mathbf{x}})(h_{\mathbf{z}} - h) = \\ = \sqrt{T_{\mathbf{x}}} g_\lambda(T_{\mathbf{x}})(h_{\mathbf{z}} - h) + (\sqrt{T} - \sqrt{T_{\mathbf{x}}}) g_\lambda(T_{\mathbf{x}})(h_{\mathbf{z}} - h). \end{aligned} \tag{38}$$

The inequality

$$||\sqrt{T} - \sqrt{T_{\mathbf{x}}}|| \leq \sqrt{||T - T_{\mathbf{x}}||} \leq \sqrt{\delta_2} \leq \sqrt{\lambda} \tag{39}$$

follows from Theorem 1 in (Mathé and Pereverzev, 2002), Lemma 2 and assumption (34). Furthermore, using the properties (13) and (14) of an admissible filter and standard results of spectral theory, it is easy to show that

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

If we take the norm of (38) we get

$$||\sqrt{T}(f_{\mathbf{z}}^\lambda - f^\lambda)||_\Gamma \leq \frac{1}{\sqrt{\lambda}}(B + \sqrt{BD})\delta_1. \tag{40}$$

We now deal with the second term in the r.h.s of (37). We can write

$$\begin{aligned} \sqrt{T}(f_{\mathcal{H}} - f^\lambda) &= \sqrt{T}(I - g_\lambda(T_{\mathbf{x}})T_{\mathbf{x}})f_{\mathcal{H}} \\ &= (\sqrt{T} - \sqrt{T_{\mathbf{x}}})(I - g_\lambda(T_{\mathbf{x}})T_{\mathbf{x}})f_{\mathcal{H}} + \\ &\quad \sqrt{T_{\mathbf{x}}}(I - g_\lambda(T_{\mathbf{x}})T_{\mathbf{x}})f_{\mathcal{H}}. \end{aligned} \tag{41}$$

We can bound this term using (39) and the following properties of admissible filters, that can be derived from (15) and (16),

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

to obtain

$$||\sqrt{T}(f^\lambda - f_{\mathcal{H}})||_\Gamma \leq (\gamma + \gamma_{\frac{1}{2}})\sqrt{\lambda}R. \tag{42}$$

The estimate in (35) follows plugging (42) and (40) into (37) and using the definition of δ_1. □

□

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

Proof. Since the sample error increases with λ while the approximation error decreases, in order to get the best error we should take the value of λ which gives a good trade-off between the two terms. To this end we set the two terms to be of the same order

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

Then, in order to be consistent with condition (34) 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 (35) we get the rate (17). □ □

Finally we prove Theorem 2.

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 $f_\rho \in \mathcal{H}$, we have that $R(c_{\mathbf{z}}) - R(b) \leq \psi(\mathcal{E}(f_{\mathbf{z}}) - \mathcal{E}(f_\rho))$, and using Theorem 1

$$R(c_{\mathbf{z}}) - R(b) \leq \psi(\frac{C \log 4/\eta}{\sqrt{n}}).$$

The proof follows since ψ is a decreasing function that goes to zero at the origin. □ □

References

Abernethy J, Bartlett PL, Rakhlin A (2007) Multitask learning with expert advice. In: COLT, pp 484–498

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

Alvarez M, Luengo D, Lawrence N (2009) Latent force models. In: Twelfth International Conference on Artificial Intelligence and Statistics

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

Argyriou A, Maurer A, Pontil M (2008b) An algorithm for transfer learning in a heterogeneous environment. In: ECML/PKDD, pp 71–85

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

Baldassarre L, Rosasco L, Barla A, Verri A (2010) Learning vector fields with spectral filtering. In: The European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), (to appear)

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

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

Bishop C (2006) Pattern Recognition and machine learning. Springer-Verlag, New York

Bonilla EV, Chai KM, Williams C (2007) Multi-task gaussian process prediction. In: Advances in Neural Information Processing Systems (NIPS), Curran Associates, Inc

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

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

Breiman L, Friedman JH (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, pp 1562–1569

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

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

Caponnetto A, De Vito E (2006) Optimal rates for regularized least-squares algorithm. Foundations of Computational Mathematics

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

Carmeli C, De Vito E, 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

Chai KMA, Williams CKI, Klanke S, 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, Schölkopf B (2003) Cluster kernels for semi-supervised learning. In: Advances in Neural Information Processing Systems (NIPS), MIT Press, pp 585–592

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

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

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

Devroye L, Györfi L, Lugosi G (1996) A Probabilistic Theory of Pattern Recognition. No. 31 in Applications of mathematics, Springer, New York

Dietterich TG, Bakiri G (1995) Solving multiclass learning problems via error-correcting output codes. Journal of Artificial Intelligence Research 2:263–286, URL citeseer.ist.psu.edu/dietterich95solving.html

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

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

Fuselier Jr E (2006) Refined error estimates for matrix-valued radial basis functions. PhD thesis, Texas A&M University

Griffin G, Holub A, Perona P (2007) Caltech-256 object category dataset. Tech. Rep. 7694, California Institute of Technology, URL http://authors.library.caltech.edu/7694

Hastie T, Tibshirani R, Friedman J (2001) The Elements of Statistical Learning. Springer, New York

Haufe S, Nikulin VV, Ziehe A, Müller KR, Nolte G (2009) Estimating vector fields using sparse basis field expansions. In: Koller D, Schuurmans D, Bengio Y, Bottou L (eds) Advances in Neural Information Processing Systems 21, pp 617–624

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

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

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

Lee Y, Lin Y, 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, Verri A (2008) Spectral algorithms for supervised learning. Neural Computation

Lowitzsch S (2005) A density theorem for matrix-valued radial basis functions. Numerical Algorithms 39(1):253–256

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

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

McAllester D (2007) Predicting Structured Data, MIT press, chap Generalization Bounds and Consistency for Structured Learning

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

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

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

Mosci S, Santoro M, Verri A, Villa S, Rosasco L (2008) Simple algorithms to solve sparsity based regularization via fenchel duality. In: OPT 2008 Optimization for Machine Learning, NIPS 2008 Workshop.

Mussa-Ivaldi F (1992) From basis functions to basis fields: vector field approximation from sparse data. Biological Cybernetics 67(6):479–489

Narcowich F, Ward J (1994) Generalized hermite interpolation via matrix-valued conditionally positive definite functions. Mathematics of Computation 63(208):661–687

Obozinski G, Taskar B, Jordan MI (2007) Multi-task feature selection. Tech. rep., Department of Statistics, UC Berkeley

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

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

Rifkin R, Lippert RA (2007) Notes on regularized least squares. Tech. Rep. CBCL Paper 268, MIT

Sheldon D (2008) Graphical multi-task learning. Tech. rep., Cornell University, URL http://www.cs.cornell.edu/ dsheldon/, preprint

Smola A, Kondor R (2003) Kernels and regularization on graphs. In: COLT

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

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

Steinwart I (2002) On the influence of the kernel on the consistency of support vector machines. Journal of Machine Learning Research 2:67–93

Szedmak S, Shawe-Taylor J, Group I (2005) Learning via linear operators: Maximum margin regression. Tech. rep., In Proceedings of 2001 IEEE International Conference on Data Mining

Tewari A, Bartlett PL (2005) On the consistency of multiclass classification methods. In: Proceedings of the 18th Annual Conference on Learning Theory, Springer, vol 3559, pp 143–157

Tsochantaridis I, Joachims T, Hofmann T, Altun Y (2005) Large margin methods for structured and interdependent output variables. Journal of Machine Learning Research 6(2):1453–1484

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

Wold S, Ruhe H, Wold H, 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

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