Spectral Algorithms for Supervised Learning

L. Lo Gerfo[*], L. Rosasco[†], F. Odone[‡], E. De Vito[§], A. Verri[¶]

October 26, 2007

Abstract

We discuss how a large class of regularization methods, collectively known as spectral regularization and originally designed for solving ill-posed inverse problems, gives rise to regularized learning algorithms. All these algorithms are consistent kernel methods which can be easily implemented. The intuition behind their derivation is that the same principle allowing to numerically stabilize a matrix inversion problem

[*]DISI, Università di Genova, v. Dodecaneso 35, 16146 Genova, Italy, LOGERFO@DISI.UNIGE.IT

[†]DISI, Università di Genova, v. Dodecaneso 35, 16146 Genova, Italy, ROSASCO@DISI.UNIGE.IT

[‡]DISI, Università di Genova, v. Dodecaneso 35, 16146 Genova, Italy, ODONE@DISI.UNIGE.IT

[§]DSA, Università di Genova, Stradone S.Agostino, 37 and INFN, Sezione di Genova, Via Dodecaneso, 33, Italy, DEVITO@DIMA.UNIGE.IT

[¶]DISI, Università di Genova, v. Dodecaneso 35, 16146 Genova, Italy, VERRI@DISI.UNIGE.IT

is crucial to avoid over-fitting. The various methods have a common derivation, but different computational and theoretical properties. We describe examples of such algorithms, analyzing their classification performance on several datasets and discussing their applicability to real world problems.

1 Introduction

A large amount of literature had pointed out the connection between algorithms in learning theory and regularization methods in inverse problems, see for example (V. Vapnik, 1982; Poggio & Girosi, 1992; V. N. Vapnik, 1998; Evgeniou, Pontil, & Poggio, 2000; T. Hastie, Tibshirani, & Friedman, 2001; Schölkopf & Smola, 2002; De Vito, Rosasco, Caponnetto, De Giovannini, & Odone, 2005). The main message is that regularization techniques provide stability with respect to noise and sampling, therefore ensuring good generalization properties to the corresponding learning algorithms. Usually, regularization in learning is based on the minimization of a functional in a suitable hypothesis space - for example, the penalized empirical error on a reproducing kernel Hilbert space. Hence theoretical analysis mainly focuses on the choice of the loss function, the penalty term and the kernel (V. N. Vapnik,

1998; Evgeniou et al., 2000; Schölkopf & Smola, 2002; T. Hastie et al., 2001).

From the seminal work of Tikhonov and others (Tikhonov & Arsenin, 1977) regularization has been rigorously defined in the theory of ill-posed inverse problems. In this context the problem is to invert a linear operator (or a matrix) that might have unbounded inverse (or a bad condition number). Regularization amounts to replace the original operator with a bounded operator, namely the *regularization operator* (Engl, Hanke, & Neubauer, 1996), whose condition number is controlled by a regularization parameter. The regularization parameter should be chosen according to the noise level in order to ensure stability. Many regularization algorithms are known, Tikhonov and truncated singular value decomposition (TSVD) being probably the most commonly used.

As discussed in (Bertero & Boccacci, 1998) one can also regard regularization from a signal processing perspective introducing the notion of *filter*. This last point of view gives a way to look constructively at regularization, indeed each regularization operator can be defined using spectral calculus as a suitable filter on the eigendecomposition of the operator defining the problem. The filter is designed to suppress the oscillatory behavior corresponding

to small eigenvalues. In this view it is known, for example, that Tikhonov regularization can be related to Wiener filter (Bertero & Boccacci, 1998).

As we mentioned, regularization has a long history in learning and our starting point is the theoretical analysis proposed in (Bauer, Pereverzev, , & Rosasco, 2006; De Vito, Rosasco, & Verri, 2005; Caponnetto & De Vito, 2006; Caponnetto, 2006) showing that many regularization methods originally proposed in the context of inverse problems give rise to consistent kernel methods with optimal minimax learning rates. The analysis we propose in this paper focuses on three points.

First, differently from (Bauer et al., 2006), we propose a more intuitive derivation of regularization based on the notion of spectral filter. We start introducing the notion of filter functions and explain why, besides ensuring numerical stability, they can also provide a way to learn with generalization guarantees. This requires, in particular, to discuss the interplay between filtering and random sampling. Our analysis is complementary to the theory developed in (Bauer et al., 2006; De Vito, Rosasco, & Verri, 2005; Yao, Rosasco, & Caponnetto, 2007; Caponnetto, 2006). Note that the fact that algorithms ensuring numerical stability can also learn is not obvious but confirms the deep connection between stability and generalization (see (Bousquet

& Elisseeff, 2002; Poggio, Rifkin, Mukherjee, & Niyogi, 2004) and (Rakhlin, Mukherjee, & Poggio, 2005) for references).

Second, we present and discuss several example of filters inducing spectral algorithms for supervised learning. The filter function perspective provides a unifying framework for discussing similarities and differences between the various methods. Some of these algorithms, such as the ν-method and iterated Tikhonov, are new to learning. Other algorithms are well known: spectral cut-off (TSVD) is related to Principal Component Regression (PCR) and its kernel version; Landweber iteration is known as L2-boosting (Bühlmann & Yu, 2002) and Tikhonov regularization is also known as regularized least squares or ridge regression. Our analysis highlights the common regularization principle underlying algorithms originally motivated by seemingly unrelated ideas: *penalized empirical risk minimization* – like the regularized least-squares, *early stopping of iterative procedures* – like the gradient descent, and *(kernel) dimensionality reduction methods* – like (kernel) Principal Component Analysis.

Despite these similarities, spectral algorithms have differences from both the computational and theoretical point of view. One of the main difference regards the so called *saturation* effect affecting some regularization schemes.

This phenomenon, which is well known in inverse problem theory, amounts to the impossibility, for some algorithms, to exploit the regularity of the target function beyond a certain critical value, referred to as the *qualification* of the method. We try to shed light on this aspect, which is usually not discussed in literature of learning rates, via some theoretical considerations and numerical simulations.

Another point that differentiates spectral algorithms concerns algorithmic complexity. An interesting aspect is the built-in property of iterative methods to recover solutions corresponding to the whole regularization path (S. Hastie T. andRosset, Tibshirani, & Zhu, 2004).

Third, we evaluate the effectiveness of the proposed algorithms in learning tasks via an extensive experimental study. The performance of these spectral algorithms against state-of-the-art techniques, such as SVMs, is assessed on various datasets. Interestingly, algorithms the implementation of which amounts to few lines of code match (and sometime improve) state-of the art results and have interesting computational properties. Optimization issues that might be interesting starting point for future work are beyond the scope of this research (see the recent work (Li, Lee, & Leung, 2007) for appropriate references).

The plan of the paper is the following. Section 2 discusses previous work in the context of filtering and learning, Section 3 reviews the regularized least-squares algorithm from a filter function perspective, while Section 4 is devoted to extending the filter point of view to a large class kernel methods. In Section 5 we give several examples of such algorithms and we discuss their properties and complexity in Section 6. Section 7 reports results obtained with an experimental analysis on various datasets, while Section 8 is left to a final discussion.

2 Previous Works on Learning, Regularization and Spectral Filtering

The idea of using regularization in statistics and machine learning has been explored since a long time - see for example (Wahba, 1990; Poggio & Girosi, 1992) and references therein - and the connection between large margin kernel methods such as Support Vector Machines and regularization is well known – see (V. N. Vapnik, 1998; Evgeniou et al., 2000; Schölkopf & Smola, 2002) and reference therein. Ideas coming from inverse problems regarded mostly the use of Tikhonov regularization and were extended to several error measures

other then the quadratic loss function. The gradient descent learning algorithm in (Yao et al., 2007) can be seen as an instance of Landweber iteration (Engl et al., 1996) and is related to the $L2$ boosting algorithm (Bühlmann & Yu, 2002). For iterative methods some partial results, which do not take into account the random sampling, are presented in (Ong & Canu, 2004; Ong, X., Canu, & Smola, 2004). The interplay between ill-posedness, stability and generalization is not new to learning (Poggio & Girosi, 1992; Evgeniou et al., 2000; Bousquet & Elisseeff, 2002; Rakhlin et al., 2005; Poggio et al., 2004; De Vito, Rosasco, Caponnetto, et al., 2005).

The notion of filter function was previously studied in machine learning and gives a connection to the literature of function approximation in signal processing and approximation theory. The pioneering work of (Poggio & Girosi, 1992) established the relation between neural networks, radial basis functions and regularization. Though the notion of reproducing kernel is not explicitly advocated, Green functions of second order differential operator are used to define penalties for penalized empirical risk minimization. Filtering and reweighing of the Fourier transform of Green functions are used de facto to design new kernels (see also for the relation between kernel and penalty term in Tikhonov regularization). The aforementioned paper, as

well as (Girosi, Jones, & Poggio, 1995) are suitable sources for references and discussions. An important aspect that we would like to stress is that, from a technical point of view, these works (implicitly) assume the data to be sampled according to a uniform distribution and make an extensive use of Fourier theory. Indeed the extension to general probability distribution is not straightforward and this is crucial since it is standard in learning theory to assume the point to be drawn according to a *general, unknown* distribution. A mathematical connection between sampling theory and learning theory has been recently proposed in (Smale & Zhou, 2004, 2005b) whereas (De Vito, Rosasco, Caponnetto, et al., 2005; De Vito, Rosasco, & Caponnetto, 2006) gave an inverse problem perspective on learning. The analysis we present can be seen as a further step towards a deeper understanding of learning as a function approximation problem.

Recently, filtering of the kernel matrix have been considered in the context of graph regularization – see (T. Hastie et al., 2001; Chapelle, Weston, & Scholkopf, 2003; Zhu, Kandola, Ghahramani, , & Lafferty, 2005; Smola & Kondor, 2003; Zhang & Ando, 2006). In this case reweighing of a kernel matrix (filters) on a set of labeled and unlabeled input points is used to define new penalty terms replacing the square of the norm in the adopted hypothesis

space. It has been shown – see for example (Zhang & Ando, 2006) – that this is equivalent to standard regularized least square with data dependent kernels. Note that in graph regularization no sampling is considered and the problem is truly a problem of transductive learning.

Our analysis relies on a different use of filter functions to define *new algorithms* rather the *new kernels*. In fact in our setting the kernel is fixed and each rescaling of the kernel matrix leads to a learning algorithm which is not necessarily a penalized minimization. The dependency of the rescaling on the regularization parameter allows us to derive consistency results in a natural way.

3 Regularized Least-Squares as a Spectral Filter

In this section we review how the generalization property of the regularized least-squares algorithm is a consequence of the algorithm being seen as a filter on the eigenvalues of the kernel matrix. This point of view naturally suggests a new class of learning algorithms defined in terms of filter functions, the properties of which are discussed in the next section.

In the framework of supervised learning, the regularized least-squares algorithm is based on the choice of a Mercer kernel[1] $K(x,t)$ on the input space X and of a regularization parameter $\lambda > 0$. Hence, for each training set $\mathbf{z} = (\mathbf{x}, \mathbf{y}) = \{(x_1, y_1), \cdots, (x_n, y_n)\}$ of n-examples $(x_i, y_i) \in X \times \mathbb{R}$, regularized least squares amounts to

$$f_{\mathbf{z}}^{\lambda}(x) = \sum_{i=1}^{n} \alpha_i K(x, x_i) \quad \text{with} \quad \alpha = (\mathbf{K} + n\lambda I)^{-1}\mathbf{y}. \tag{1}$$

where $\mathbf{K}$ is the $n \times n$-matrix $(\mathbf{K})_{ij} = K(x_i, x_j)$.

Since $\lambda > 0$, it is clear that we are numerically stabilizing a matrix inversion problem which is possibly ill-conditioned (that is numerically unstable). Before showing that regularized least squares can be seen as a suitable filtering of the kernel matrix, able to to ensure goog generalization properties of the estimator, we first need to recall some basic concepts of learning theory.

We assume that the examples (x_i, y_i) are drawn identically and independently distributed according to an unknown probability measure $\rho(x, y) = \rho(y|x)\rho_X(x)$. Moreover, we assume that X is a compact subset of $\mathbb{R}^d$, and the labels y_i belong into a bounded subset $Y \subset \mathbb{R}$ (for example in a binary

[1]This means that $K : X \times X \to \mathbb{R}$ is a symmetric continuous function, which is positive definite (Aronszajn, 1950).

classification problem $Y = \{-1, 1\}$). Finally, we assume that the kernel K is bounded by 1 and is universal (see (Micchelli, Xu, & H., 2006)and references therein), that is the set of functions

$$\mathcal{H} = \{\sum_{i=1}^{N} \alpha_i K(x, x_i) \mid x_i \in X,\ \alpha_i \in \mathbb{R}\}$$

is dense in $L^2(X)$, the Hilbert space of functions that are square-integrable with respect to ρ_X.

With the choice of the square loss, the generalization property of the estimator means that the estimator $f_{\mathbf{z}}^{\lambda}$ is a good approximation of the regression function

$$f_\rho(x) = \int_Y y \; d\rho(y|x),$$

with respect to the norm of $L^2(X)$. In particular the algorithm is (weakly) consistent (V. N. Vapnik, 1998) if, for a suitable choice of the parameter $\lambda = \lambda_n$ as a function of the examples,

$$\lim_{n\to\infty} \int_X (f_{\mathbf{z}}^{\lambda}(x) - f_\rho(x))^2 \; d\rho_X(x) = \lim_{n\to\infty} \left\| f_{\mathbf{z}}^{\lambda} - f_\rho \right\|_\rho^2 = 0$$

with high probability – see for example (V. N. Vapnik, 1998).

Notice that, in classification, the goal is to approximate the Bayes rule $\text{sign}(f_\rho) = \text{sign}(\rho(1|x) - 1/2)$ with the plug-in estimator $\text{sign}(f_{\mathbf{z}}^\lambda)$ with respect to the classification error $R(f_{\mathbf{z}}^\lambda) = P(y f_{\mathbf{z}}^\lambda(x) < 0)$. In any case the following bound holds

$$\mathcal{R}(f_{\mathbf{z}}^\lambda) - \mathcal{R}(f_\rho) \leq \left\| f_{\mathbf{z}}^\lambda - f_\rho \right\|_\rho , \tag{2}$$

see for example (Bartlett, Jordan, & McAuliffe, 2006), so in the following discussion we only consider the square loss.

We start re-writing the equation (1) in a slightly different way

$$(f_{\mathbf{z}}^\lambda(x_1), \ldots, f_{\mathbf{z}}^\lambda(x_n)) = \frac{\mathbf{K}}{n}(\frac{\mathbf{K}}{n} + \lambda)^{-1}\mathbf{y}. \tag{3}$$

Observe that, if v is an eigenvector of $\mathbf{K}/n$ with eigenvalue σ, then we have $\frac{\mathbf{K}}{n}(\frac{\mathbf{K}}{n} + \lambda)^{-1}v = \frac{\sigma}{\sigma+\lambda}v$, so that the regularized least-squares algorithm is in fact a filter on the eigenvalues of the kernel matrix.

The filter $\frac{\sigma}{\sigma+\lambda}$ not only ensures numerical stability, but also the generalization properties of the estimator. To obtain a deep insight on this point, consider the population case when we have knowledge of the probability distribution ρ generating the data. In this setting, the kernel matrix $\mathbf{K}/n$ is replaced by

the integral operator L_K with kernel K

$$L_K f(x) = \int_X K(x,s)\, f(s) d\rho_X(s) \qquad f \in L^2(X), \tag{4}$$

the data $\mathbf{y}$ is replaced by the regression function f_ρ, so that (3) becomes

$$f^\lambda = L_K (L_K + \lambda I)^{-1} f_\rho. \tag{5}$$

More explicitly, since L_K is a positive compact operator bounded by 1 and $\mathcal{H}$ is dense in $L^2(X)$, there is basis $(u_i)_{i\geq 1}$ in $L^2(X)$ such that $L_K u_i = \sigma_i u_i$ with $0 < \sigma_i \leq 1$ and $\lim_{i\to\infty} \sigma_i = 0$. Hence

$$\begin{aligned} f_\rho &= \sum_{i=1}^{\infty} \langle f_\rho, u_i \rangle_\rho \, u_i \\ f^\lambda &= \sum_{i=1}^{\infty} \frac{\sigma_i}{\sigma_i + \lambda} \langle f_\rho, u_i \rangle_\rho \, u_i. \end{aligned}$$

By comparing the two equations, one has that f^λ is a good approximation of f_ρ, provided that λ is small enough. For such λ, the filter $\frac{\sigma}{\sigma+\lambda}$ selects only the components of the f_ρ corresponding to large eigenvalues, which are a finite number since the sequence of eigenvalues goes to zero. Hence, if we slightly perturb both L_K and f_ρ, the corresponding solution of (4) is close

to f_ρ, provided that the perturbation is small. The key idea is that now we can regard the sample case $\mathbf{K}$, $\mathbf{y}$ and the corresponding estimator $f^\lambda_{\mathbf{z}}$, as perturbation of L_K, f_ρ and f^λ, respectively. A mathematical proof of the above intuition requires some work and we refer to (De Vito, Rosasco, & Verri, 2005; Bauer et al., 2006) for the technical details. The basic idea is that law of large numbers ensures that the perturbation is small, provided that the number of example is large enough and, as a consequence, $f^\lambda_{\mathbf{z}}$ is close to f^λ and, hence, to f_ρ.

The above discussion suggests that one can replace $\frac{\sigma}{\sigma+\lambda}$ with other functions $\sigma g_\lambda(\sigma)$ which are filters on the eigenvalues of L_K and obtain different regularization algorithms, as shown in the next section.

4 Kernel Methods from Spectral Filtering

In this section we discuss the properties of kernel methods based on spectral filtering. Our approach is inspired by inverse problems. A complete theoretical discussion of our approach can be found in (De Vito, Rosasco, & Verri, 2005; Bauer et al., 2006; Caponnetto, 2006).

Let K be a Mercer kernel as in the above section, by looking at (1), this

suggests to define a new class of learning algorithm by letting

$$f_{\mathbf{z}}^{\lambda} = \sum_{i=1}^{n} \alpha_i K(x, x_i) \qquad \text{with} \qquad \alpha = \frac{1}{n} g_\lambda(\frac{\mathbf{K}}{n})\mathbf{y}, \tag{6}$$

where $g_\lambda : [0,1] \to \mathbb{R}$ is a suitable function and $g_\lambda(\frac{\mathbf{K}}{n})$ is defined by spectral calculus, that is, if v is an eigenvector of $\mathbf{K}/n$ with eigenvalue σ (since K is a Mercer kernel bounded by 1, $0 \leq \sigma \leq 1$), then $g_\lambda(\frac{\mathbf{K}}{n})v = g_\lambda(\sigma)v$. In particular, on the given data, one has

$$(f_{\mathbf{z}}^{\lambda}(x_1), \ldots, f_{\mathbf{z}}^{\lambda}(x_n)) = \frac{\mathbf{K}}{n}\ g_\lambda(\frac{\mathbf{K}}{n})\mathbf{y}. \tag{7}$$

We note that, unlike regularized least squares, such an estimator is not necessarily the solution of penalized empirical minimization. Clearly, to ensure both numerical stability as well as consistency, we need to make some assumptions on g_λ. Following (De Vito, Rosasco, & Verri, 2005; Bauer et al., 2006) we say that a function $g_\lambda : [0,1] \to \mathbb{R}$ parameterized by $0 < \lambda \leq 1$ is an admissible filter function if:

1. There exists a constant B such that

$$\sup_{0<\sigma\leq 1} |g_\lambda(\sigma)| \leq \frac{B}{\lambda} \qquad \forall \lambda \in [0,1]. \tag{8}$$

2. There exists a constant D such that

$$\begin{aligned} \lim_{\lambda \to 0} \sigma g_\lambda(\sigma) &= 1 \quad \forall \sigma \in]0,1] \\ \sup_{0<\sigma\leq 1} |\sigma g_\lambda(\sigma)| &\leq D \quad \forall \lambda \in [0,1]. \end{aligned} \tag{9}$$

3. There is a constant $\overline{\nu} > 0$, namely *the qualification of the regularization* g_λ such that

$$\sup_{0<\sigma\leq 1} |1 - g_\lambda(\sigma)\sigma|\sigma^\nu \leq \gamma_\nu \lambda^\nu, \qquad \forall\ 0 < \nu \leq \overline{\nu}, \tag{10}$$

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

A simple computation shows that $g_\lambda(\sigma) = \frac{1}{\sigma+\lambda}$ is an admissible filter function, indeed Eqs. (8) and (9) hold with $B = D = 1$, condition (10) is verified with $\gamma_\nu = 1$ for $0 < \nu \leq 1$ and hence the qualification equals to 1. Other examples are discussed in the next section. Here we give a heuristic motivation of the above conditions having in mind the discussion in the previous section. First, observe that population version of (6) becomes

$$f^\lambda = \sum_i \sigma_i g_\lambda(\sigma_i) \left\langle f_\rho, u_i \right\rangle_\rho u_i. \tag{11}$$

We can make the following observations.

1. Eq. (8) ensures that eigenvalues of $g_\lambda(\mathbf{K})$ are bounded by $\frac{B}{\lambda}$, so that (6) is numerically stable. Moreover, looking at (11), we see that it also implies that, if σ_i is much smaller than λ, the corresponding Fourier coefficient $\left\langle f^\lambda, u_i \right\rangle$ of f^λ is small. Hence, f^λ has essentially only a finite number of non-zero Fourier coefficients on the basis $(u_i)_{i\geq 1}$ and we can argue that, by the law of large numbers, $f^\lambda_{\mathbf{z}}$ is a good approximation of f^λ when n is large enough.

2. Assumption (9) implies that f^λ converges to f_ρ if λ goes to zero. In terms of the kernel matrix, such a condition means that $g_\lambda(\mathbf{K})$ converges to $\mathbf{K}^{-1}$ when λ goes to zero, avoiding over-smoothing.

3. Condition (10) is related to the convergence rates of the algorithm. These rates depend on how fast the Fourier coefficients $\langle f_\rho, u_i \rangle$ converge to 0 with respect to the eigenvalues σ_i (Bauer et al., 2006). This information is encoded by a priori assumptions on f_ρ of the form

$$\sum_{i=1}^{\infty} \frac{\langle f_\rho, u_i \rangle_\rho^2}{\sigma_i^{2r}} < R, \tag{12}$$

where the parameter r encodes the regularity property of the regression

function. If $r = 1/2$ this corresponds to assuming $f_\rho \in \mathcal{H}$ and, more generally, the larger is r the smoother is the function. Condition (10) and the choice $\lambda_n = \frac{1}{n^{2r+1}}$ ensure that, if $r \leq \overline{\nu}$,

$$\left\| f_{\mathbf{z}}^{\lambda_n} - f_\rho \right\|_\rho \leq C n^{-\frac{r}{2r+1}} \quad \text{with high probability,} \tag{13}$$

whereas, if $r \geq \overline{\nu}$, the rate of convergence is always $n^{-\frac{\overline{\nu}}{2\overline{\nu}+1}}$; for a proof and a complete discussion see (Bauer et al., 2006; Caponnetto, 2006). Hence filter functions having a larger qualification $\overline{\nu}$ gives better rates, that is, the corresponding algorithms can better exploit the smoothness of f_ρ. This fact marks a big distinction among the various algorithms we consider as we discussed in the following.

Also, notice that in the classification setting, that we will consider in our experiments, if $r \leq \overline{\nu}$ bounds (2) and from (13) we have that

$$\mathcal{R}(f_{\mathbf{z}}^{\lambda}) - \inf_f \mathcal{R}(f) = O(n^{-\frac{r}{2r+1}})$$

with high probability.

Considering the decomposition $f_{\mathbf{z}}^{\lambda} - f_\rho = (f_{\mathbf{z}}^{\lambda} - f^{\lambda}) + (f^{\lambda} - f_\rho)$, from the above discussion we have that the consistency of this class of learning algorithms

depends on two opposite terms: the approximation error $\left\| f^{\lambda} - f_{\rho} \right\|$ and the sample error $\left\| f_{\mathbf{z}}^{\lambda} - f^{\lambda} \right\|$. The approximation error depends on the examples only trough $\lambda = \lambda_n$ and it decreases if λ goes to zero, whereas the sample error is of probabilistic nature and it increases if λ goes to zero. The optimal choice of the regularization parameter λ will be a trade-off between these two errors – see (De Vito, Caponnetto, & Rosasco, 2005; Caponnetto & De Vito, 2006; Smale & Zhou, 2005a; Wu, Ying, & Zhou, 2006) and references therein about the rates for regularized least-squares, and (De Vito, Rosasco, & Verri, 2005; Bauer et al., 2006; Caponnetto, 2006) for arbitrary filters.

Before giving several examples of algorithms fitting into the above general framework we observe that the considered algorithms can be regarded as filters on the expansion of the target function on a suitable basis. In principle, this basis can be obtained from the spectral decomposition of the integral operator L_K and, in practice, is approximated by considering the spectral decomposition of the kernel matrix $\mathbf{K}$. Interestingly the basis thus obtained has a natural interpretation: if the data are centered (in the feature space), then the elements of the basis are the principal components of the expected (and empirical) covariance matrix in the feature space. In this respect the spectral methods we discussed rely on the assumption that most

of the information is actually encoded in the first principal components.

5 The Proposed Algorithms

In this section we give some specific examples of kernel methods based on spectral regularization. All these algorithms are known in the context of regularization for linear inverse problems but only some of them have been used for statistical inference problems. These methods have many interesting features: from the algorithmic point of view they are simple to implement, usually they amount to a few lines of code. They are appealing for applications: their model selection is simple since they depend on few parameters, while over-fitting may be dealt in a very transparent way. Some of them represent a very good alternative to Regularized Least Squares as they are faster without compromising classification performance (see Section 7). Note that for regularized least squares the algorithm has the following variational formulation

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

which can be interpreted as an extension of empirical risk minimization. In general the class of regularization might not be described by a variational

problem so that filter point of view provides us with a suitable description.

More details on the derivation of these algorithms can be found in (Engl et al., 1996).

5.1 Iterative Landweber

Landweber iteration is characterized by the filter function

$$g_t(\sigma) = \tau \sum_{i=0}^{t-1} (1 - \tau\sigma)^i$$

where we identify $\lambda = t^{-1}, t \in \mathbb{N}$ and take $\tau = 1$ (since the kernel is bounded by 1). In this case we have $B = D = 1$ and the qualification is infinite since (10) holds with $\gamma_\nu = 1$ if $0 < \nu \leq 1$ and $\gamma_\nu = \nu^\nu$ otherwise. The above filter can be derived from a variational point of view. In fact, as shown in (Yao et al., 2007), this method corresponds to empirical risk minimization via gradient descent. If we denote with $\|\cdot\|_n$ the norm in $\mathbb{R}^n$, we can impose

$$\nabla \|\mathbf{K}\alpha - \mathbf{y}\|_n^2 = 0,$$

and by a simple calculation we see that the solution can be rewritten as the following iterative map

$$\alpha_i = \alpha_{i-1} + \frac{\tau}{n}(\mathbf{y} - \mathbf{K}\alpha_{i-1}), \qquad i = 1, \ldots, t$$

where τ determines the step-size. We may start from a very simple solution, $\alpha_0 = 0$. Clearly if we let the number of iterations grow we are simply minimizing the empirical risk and are bound to overfit. Early stopping of the iterative procedure allows us to avoid over-fitting, thus the iteration number plays the role of the regularization parameter. In (Yao et al., 2007) the fixed step-size $\tau = 1$ was shown to be the best choice among the variable step-size $\tau = \frac{1}{(t+1)^\theta}$, with $\theta \in [0, 1)$. This suggests that τ does not play any role for regularization. Landweber regularization was introduced under the name of L2-boosting for splines in a fixed design statistical model (Bühlmann & Yu, 2002) and eventually generalized to general RKH spaces and random design in (Yao et al., 2007).

5.2 Semi-iterative Regularization

An interesting class of algorithms are the so called semi-iterative regularization or accelerated Landweber iteration. These methods can be seen as a generalization of Landweber iteration where the regularization is now

$$g_t(\sigma) = p_t(\sigma)$$

with p_t a polynomial of degree $t-1$. In this case we can identify $\lambda = t^{-2}, t \in \mathbb{N}$. One can show that $D = 1$, $B = 2$ and the qualification of this class of methods is usually finite (Engl et al., 1996).

An example which turns out to be particularly interesting is the so called $\nu - method$. The derivation of this method is fairly complicated and relies on the use of orthogonal polynomials to obtain acceleration of the standard gradient descent algorithm (see chapter 10 in (Golub & Van Loan, 1996)). Such a derivation is beyond the scope of this presentation and we refer the interested reader to (Engl et al., 1996). In the $\nu - method$ the qualification is ν (fixed) with $\gamma_\nu = c$ for some positive constant c. The algorithm amounts

to solving (with $\alpha_0 = 0$) the following map

$$\alpha_i = \alpha_{i-1} + u_i(\alpha_{i-1} - \alpha_{i-2}) + \frac{\omega_i}{n}(\mathbf{y} - \mathbf{K}\alpha_{i-1}), \quad i = 1, \ldots, t$$

where

$$\begin{aligned} u_i &= \frac{(i-1)(2i-3)(2i+2\nu-1)}{(i+2\nu-1)(2i+4\nu-1)(2i+2\nu-3)} \\ \omega_i &= 4\frac{(2i+2\nu-1)(i+\nu-1)}{(i+2\nu-1)(2i+4\nu-1)} \qquad t > 1. \end{aligned}$$

The interest of this method lies in the fact that since the regularization parameter here is $\lambda = t^{-2}$, we just need the square root of the number of iterations needed by Landweber iteration. In inverse problems this method is known to be extremely fast and is often used as a valid alternative to conjugate gradient – see (Engl et al., 1996), Chapter 6 for details. To our knowledge semi-iterative regularization has not been previously in learning.

5.3 Spectral Cut-Off

This method, also known as truncated singular values decomposition (TSVD), is equivalent to the so called (kernel) principal component regression. The filter function is simply

$$g_\lambda(\sigma) = \begin{cases} \frac{1}{\sigma} & \sigma \geq \lambda \\ 0 & \sigma < \lambda \end{cases}$$

In this case, $B = D = 1$. The qualification of the method is arbitrary and $\gamma_\nu = 1$ for any $\nu > 0$. The corresponding algorithm is based on the following simple idea. Perform SVD of the kernel matrix $\mathbf{K} = \mathbf{U}\mathbf{S}\mathbf{U}^T$ where $\mathbf{U}$ is an orthogonal matrix and $\mathbf{S} = diag(\sigma_1, \ldots, \sigma_n)$ is diagonal with $\sigma_i \geq \sigma_{i+1}$. Then discard the singular values smaller than the threshold λ, replace them with 0. The algorithm is then given by

$$\alpha = \mathbf{K}_\lambda^{-1}\mathbf{y} \tag{14}$$

where $\mathbf{K}_\lambda^{-1} = \mathbf{U}^T\mathbf{S}_\lambda^{-1}\mathbf{U}$ and $\mathbf{S}_\lambda^{-1} = diag(1/\sigma_1, \ldots, 1/\sigma_m, 0, \ldots)$ where $\sigma_m \geq \lambda$ and $\sigma_{m+1} < \lambda$. The regularization parameter is the threshold λ or, equivalently, the number m of components that we keep.

Finally, notice that, if the data are centered in the feature space, then the columns of the matrix $\mathbf{U}$ are the principal components of the covariance matrix in the feature space and the spectral cut-off is a filter that discards the projection on the last principal components. The procedure is well known in literature as kernel principal component analysis – see for example (Schölkopf & Smola, 2002).

5.4 Iterated Tikhonov

We conclude this section mentioning a method which is a mixture between Landweber iteration and Tikhonov regularization. Unlike Tikhonov regularization which has finite qualification and cannot exploit the regularity of the solution beyond a certain regularity level, iterated Tikhonov overcomes this problem by means of the following regularization

$$g_\lambda(\sigma) = \frac{(\sigma+\lambda)^\nu - \lambda^\nu}{\sigma(\sigma+\lambda)^\nu}, \quad \nu \in \mathbb{N}.$$

In this case we have $D = 1$ and $B = t$ and the qualification of the method is now ν with $\gamma_\nu = 1$ for all $0 < \nu \leq t$. The algorithm is described by the

following iterative map

$$(\mathbf{K} + n\lambda I)\alpha_i = \mathbf{y} + n\lambda\alpha_{i-1} \qquad i = 1, \ldots, \nu$$

choosing $\alpha_0 = 0$. It is easy to see that for $\nu = 1$ we simply recover the standard Tikhonov regularization but as we let $\nu > 1$ we improve the qualification of the method with respect to standard Tikhonov. Moreover we note that by fixing λ we can think of the above algorithms as an iterative regularization with ν as the regularization parameter.

6 Different Properties of Spectral Algorithms

In this section we discuss the differences from the theoretical and computational viewpoints of the proposed algorithms.

6.1 Qualification and Saturation Effects in Learning

As we mentioned in Section 4 one of the main differences between the various spectral methods is their qualification. Each spectral regularization algorithm has a critical value (the qualification) beyond which learning rates no longer improve despite the regularity of the target function f_ρ. If this is

the case we say that methods saturate. In this section we recall the origin of this problem and illustrate it with some numerical simulations.

Saturation effects have their origin in analytical and geometrical properties rather than in statistical properties of the methods. To see this recall the error decomposition $f_{\mathbf{z}}^{\lambda} - f_{\rho} = (f_{\mathbf{z}}^{\lambda} - f^{\lambda}) + (f^{\lambda} - f_{\rho})$, where the latter term is the approximation error that, recalling (11), is related to the behavior of

$$\begin{aligned} f_{\rho} - f^{\lambda} &= \sum_i \langle f_{\rho}, \sigma_i \rangle_{\rho} u_i - \sum_i \sigma_i g_{\lambda}(\sigma_i) \langle f_{\rho}, \sigma_i \rangle_{\rho} u_i \\ &= \sum_i (1 - \sigma_i g_{\lambda}(\sigma_i)) \sigma_i^r \frac{\langle f_{\rho}, \sigma_i \rangle_{\rho}}{\sigma_i^r} u_i. \end{aligned} \tag{15}$$

If the regression function satisfies (12) we have

$$\left\| f_{\rho} - f^{\lambda} \right\|_{\rho} \leq R \sup_{0 < \sigma \leq 1} (|1 - g_{\lambda}(\sigma)\sigma| \sigma^r).$$

The above formula clearly motivates condition (10) and the definition of qualification. In fact it follows that if $r \leq \bar{\nu}$ then $\left\| f_{\rho} - f^{\lambda} \right\|_{\rho} = O(\lambda^r)$ whereas if $r > \bar{\nu}$ we have $\left\| f_{\rho} - f^{\lambda} \right\|_{\rho} = O(\lambda^{\bar{\nu}})$. To avoid confusion note that the index r in the above equations encodes a regularity property of the target function whereas $\bar{\nu}$ in (10) encodes a property of the given algorithm.

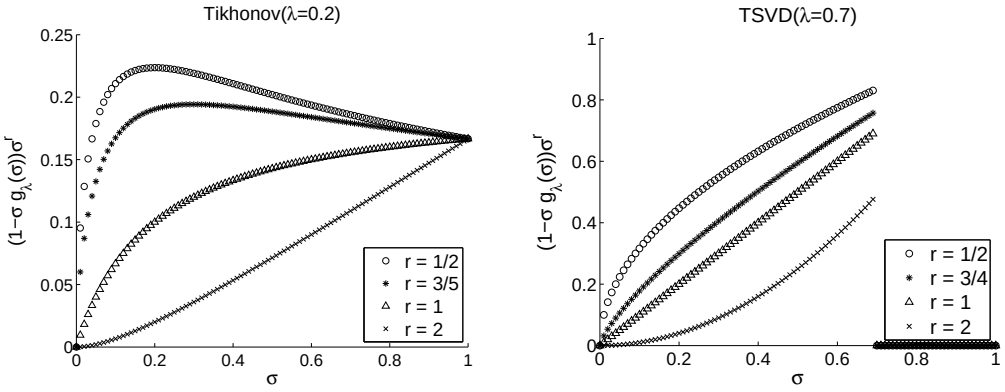

Figure 1: The behaviors of the residuals for Tikhonov regularization (left) and TSVD (right) as a function of σ for different values of r and fixed λ.

In Figure 1 we show the behaviors of the residual $(1-\sigma g_\lambda(\sigma))\sigma^r$ as a function of σ for different values of r and fixed λ. For Tikhonov regularization (Figure 1, left) in the two top plots - where $r < 1$ - the maximum of the residual changes and is achieved within the interval $0 < \sigma < 1$, whereas in the two bottom plots - where $r \geq 1$ - the maximum of the residual remains the same and is achieved for $\sigma = 1$. For TSVD (Figure 1, right) the maximum of the residual changes for all the values of the index r, and is always achieved at $\sigma = \lambda$. An easy calculation shows that the behavior of iterated Tikhonov is the same as Tikhonov but the critical value is now ν rather than 1. Similarly one can recover the behavior of ν-method and Landweber iteration.

In Figure 2 we show the corresponding behavior of the approximation error as a function of λ for different values of r. Again the difference between

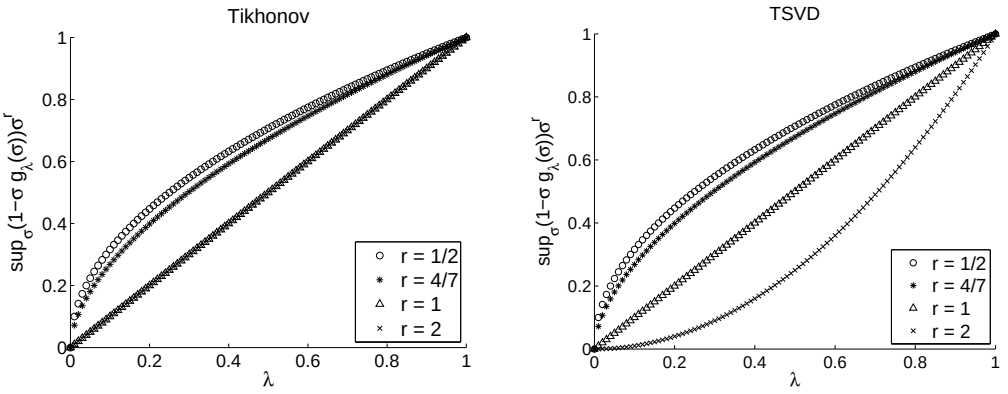

Figure 2: The behaviors of the approximation errors for Tikhonov regularization (left) and TSVD (right) as a function of λ for different values of r.

finite (Tikhonov) and infinite (TSVD) qualification is apparent. For Tikhnov regularization (Figure 2, left) the approximation error is $O(\lambda^r)$ for $r < 1$ (see the two top plots) and is $O(\lambda)$ for $r \geq 1$ (the plots for $r = 1$ and $r = 2$ overlap) since the qualification of the method is 1. For TSVD (Figure 2, right) the approximation error is always $O(\lambda^r)$ since the qualification is infinite. Again similar considerations can be done with iterated Tikhonov as well as for the other methods.

To further investigate the saturation effect we consider a regression toy problem and evaluate the effect of finite qualification on the expected error. Clearly this is more difficult since the effect of noise and sampling contributes to the error behavior through the sampling error as well. In our toy example

X is simply the interval $[0, 1]$ endowed with the uniform probability measure $d\rho_X(x) = dx$. As hypotheses space we choose the Sobolev space of absolutely continuous, with square integrable first derivative and boundary condition $f(0) = f(1) = 0$. This is a Hilbert space of function endowed with the norm

$$\|f\|^2_{\mathcal{H}} = \int_0^1 f'(x)dx$$

and can be shown to be a RKH space with kernel

$$K(x, s) = \Theta(x \geq s)(1 - x)s + \Theta(x \leq s)(1 - s)x$$

where Θ is the Heavyside step function. In this setting we compare the performance of spectral regularization methods in two different learning tasks. In both cases the output is corrupted by Gaussian noise. The first task is to recover the regression function given by $f_\rho(x) = K(x_0, x)$ for a fixed point x_0 given a priori, and the second task is to recover the regression function $f_\rho(x) = \sin(x)$. The two cases should correspond roughly to $r = 1/2$ and $r >> 1$. In Figure 3 we show the behavior, for various training set sizes, of

$$\Delta(n) = \min_\lambda \left\| f_\rho - f^\lambda_{\mathbf{z}} \right\|^2_\rho$$

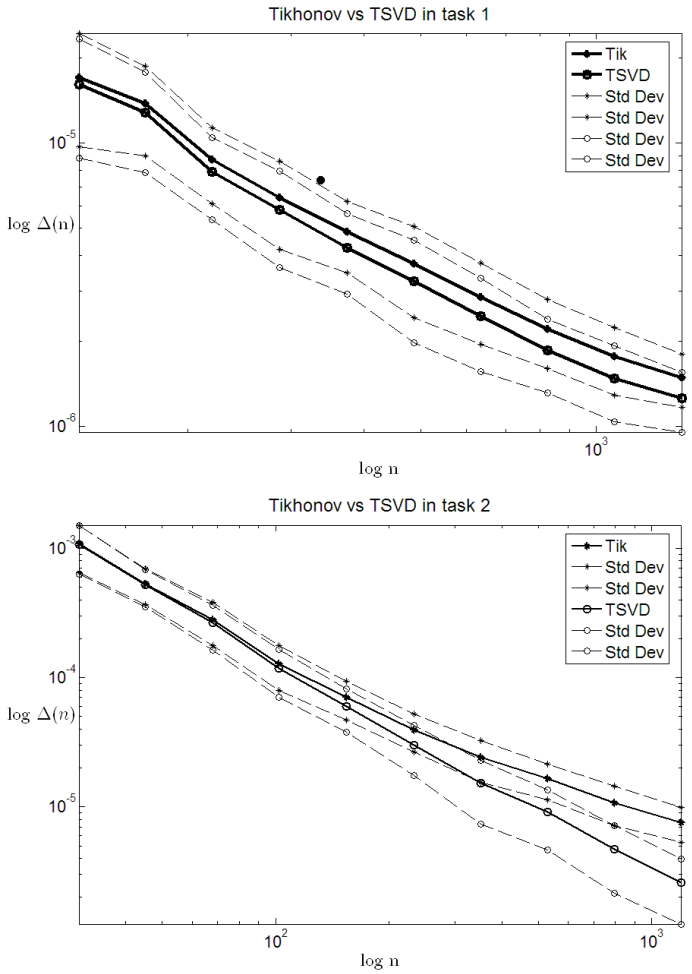

Figure 3: The comparisons of the learning rates for Tikhonov regularization and TSVD on two learning tasks with very different regularity indexes. In the first learning task (top plot) the regression function is less regular than in the second learning task (bottom plot). The continuous plots represent the average learning rates over 70 trials while the dashed plots represent the average learning rates plus and minus one standard deviation.

where we took a sample of cardinality $N >> n$ to approximate $\|f\|_\rho^2$ with $\frac{1}{N}\sum_{i=1}^{N} f(x_i)$. The plot is the average over 70 repeated trials and We considered 70 repeated trials and show the average learning rates plus and minus one standard deviation. The results in Figure 3 confirm the presence of a saturation effect. For the first learning task (top plot) the learning rates of Tikhonov and TSVD is essentially the same, but TSVD has better learning rates than Tikhonov in the second learning task (bottom plot) where the regularity is higher. We performed similar simulations, not reported here, comparing the learning rates for Tikhonov and iterated Tikhonov regularization recalling that the latter has higher qualification. As expected, iterated Tikhonov has better learning rates in the second learning tasks and essentially the same learning rates in the first task. Interestingly we found the real behavior of the error to be better than the one expected from the probabilistic bound, and we conjecture that this is due to pessimistic estimate of the sample error bounds.

6.2 Algorithmic Complexity and Regularization Path

In this section we will comment on the properties of spectral regularization algorithms in terms of algorithmic complexity.

Having in mind that each of the algorithms we discussed depends on at least one parameter[2] we are going to distinguish between: (1) the computational cost of each algorithm for one fixed parameter value and (2) the computational cost of each algorithm to find the solution corresponding to many parameter values. The first situation corresponds to the case when a correct value of the regularization parameter is given a priori or has been computed already. The complexity analysis in this case is fairly standard and we compute it in a worst case scenario, though for nicely structured kernel matrices (for example sparse or block structured) the complexity can be drastically reduced.

The second situation is more interesting in practice since one usually has to *find* a good parameter value, therefore the real computational cost includes the parameter selection procedure. Typically one computes solutions corresponding to different parameter values and then chooses the one minimizing some estimate of the generalization error, for example hold-out or leave-one-out estimates (T. Hastie et al., 2001). This procedure is related to the concept of *regularization path* (S. Hastie T. andRosset et al., 2004). Roughly speaking the regularization path is the sequence of solutions, cor-

[2]In general, besides the regularization parameter, there might be some kernel parameter. In our discussion we assume the kernel (and its parameter) to be fixed.

responding to different parameters, that we need to compute to select the best parameter estimate. Ideally one would like the cost of calculating the regularization path to be as close as possible to that of calculating the solution for a fixed parameter value. In general this is a strong requirement but, for example, SVM algorithm has a step-wise linear dependence on the regularization parameter (Pontil & Verri, 1998) and this can be exploited to find efficiently the regularization path (S. Hastie T. andRosset et al., 2004).

Given the above premises, analyzing spectral regularization algorithms we notice a substantial difference between iterative methods (*Landweber* and *ν-method*) and the others. At each iteration, iterative methods calculate a solution corresponding to t, which is both the iteration number and the regularization parameter (as mentioned above, equal to $1/\lambda$). In this view iterative methods have the built-in property of computing the *whole* regularization path. Landweber iteration at each step i performs a matrix-vector product between $\mathbf{K}$ and α_{i-1} so that at each iteration the complexity is $O(n^2)$. If we run t iteration the complexity is then $O(t * n^2)$. Similarly to Landweber iteration, the ν-method involves a matrix-vector product so that each iteration costs $O(n^2)$. However, as discussed in Section 5, the number of iteration required to obtain the same solution of Landweber iteration is the

square root of the number of iterations needed by Landweber (see also Table 2). Such rate of convergence can be shown to be optimal among iterative schemes (see (Engl et al., 1996)). In the case of *RLS* in general one needs to perform a matrix inversion for each parameter value that costs in the worst case $O(n^3)$. Similarly for spectral cut-off the cost is that of finding the singular value decomposition of the kernel matrix which is again $O(n^3)$. Finally we note that computing solution for different parameter values is in general very costly for a standard implementation of RLS, while for spectral cut-off one can perform only one singular value decomposition. This suggests the use of SVD decomposition also for solving RLS, in case a parameter tuning is needed.

7 Experimental analysis

This section reports experimental evidence of the effectiveness of the algorithms discussed in Section 5. We apply them to a number of classification problems, first considering a set of well known benchmark data and comparing the results we obtain with the ones reported in the literature; then we consider a more specific application, face detection, analyzing the results

obtained with a spectral regularization algorithm and comparing them with SVM, which has been applied with success in the past by many authors. For these experiments we consider both a benchmark dataset available on the web and a set of data acquired by a video-monitoring system designed in our lab.

7.1 Experiments on benchmark datasets

In this section we analyze the classification performance of the regularization algorithms on various benchmark datasets. In particular we consider the IDA benchmark, containing one toy dataset (`banana` — see Table 1), and several real datasets[3]. These datasets have been previously used to assess many learning algorithms, including Adaboost, RBF networks, SVMs, and Kernel Projection Machines. The benchmarks webpage reports the results obtained with these methods and which for our comparisons.

For each dataset, 100 resamplings into training and test sets are available from the website. The structure of our experiments follows the one reported on the benchmarks webpage: we perform parameter estimation with 5-fold cross validation on the first 5 partitions of the dataset, then we compute the

[3]This benchmark is available at the website:
`http://ida.first.fraunhofer.de/projects/bench/`.

median of the 5 estimated parameters and use it as an optimal parameter for all the resamplings. As for the choice of *parameter* σ (i.e., the standard deviation of the RBF kernel), at first we set the value to the average of square distances of training set points of two different resamplings: let it be σ_c. Then we compute the error on two randomly chosen partitions on on the range $[\sigma_c - \delta, \sigma_c + \delta]$ for a small δ, on several values of λ and choose the most appropriate σ. After selecting σ, the *parameter* t (corresponding to $1/\lambda$) is tuned with 5-CV on the range $[1, \infty]$ where κ is $\sup_{x \in X} K(x, x)$. Regarding the choice of the parameter ν for the $\nu - method$ and iterated Tikhonov (where ν is the number of iteration) we tried different values obtaining very similar results. The saturation effect on real data seemed much harder to spot and all the errors where very close. In the end we chose $\nu = 5$ for both methods.

Table 2 shows the average generalization performance (with standard deviation) over the data sets partitions. It also reports the parameters σ and t $(= 1/\lambda)$ chosen to find the best model. The results obtained with the five methods are very similar, with the exception of Landweber whose performances are less stable. The $\nu - method$ performs very well and converges to a solution in fewer iterations.

Table 1: The 13 benchmark datasets used:their size (training and test), the space dimension and the number of splits in training/test.

	#Train	#Test	Dim	#Resampl.
(1)Banana	400	4900	2	100
(2)B.Canc.	200	77	9	100
(3)Diabet.	468	300	8	100
(4)F.Solar	666	400	9	100
(5)German	700	300	20	100
(6)Heart	170	100	13	100
(7)Image	1300	1010	18	20
(8)Ringn.	400	7000	20	100
(9)Splice	1000	2175	60	20
(10)Thyroid	140	75	5	100
(11)Titanic	150	2051	3	100
(12)Twonorm	400	7000	20	100
(13)Wavef.	400	4600	21	100

From this analysis we conclude that the $\nu - method$ shows the best combination of generalization performance and computational efficiency among the four regularization methods analyzed. We choose it as a representative for comparisons with other approaches. Table 3 compares the results obtained with the ν-method, with an SVM with RBF kernel, and also, for each dataset, with the classifier performing best among the 7 methods considered on the benchmark page (including RBF networks, Adaboost and Regularized AdaBoost, Kernel Fisher Discriminant, and SVMs with RBF kernels). The results obtained with the $\nu - method$ compare favorably with the ones achieved by the other methods.

Table 2: Comparison of the 5 methods we discuss. The average and standard deviation of the generalization error on the 13 datasets (numbered as in the Table 1) is reported on top and the value of the regularization parameter and the gaussian width - (t/σ) - on the bottom of each row. The best result for each dataset is in bold face.

	LANDWEBER	ν-METH	RLS	TSVD	IT($\nu = 5$)
1	11.70 ± 0.68	$\mathbf{10.67 \pm 0.53}$	11.22 ± 0.61	11.74 ± 0.63	10.96 ± 0.56
	(116/1)	(70/1)	(350/1)	(301/1)	(141/1)
2	25.38 ± 4.21	25.35 ± 4.24	$\mathbf{25.12 \pm 4.32}$	26.81 ± 4.32	25.26 ± 4.14
	(5/2)	(5/2)	(41/2)	(120/2)	(4/2)
3	23.70 ± 1.80	$\mathbf{23.60 \pm 1.82}$	24.40 ± 1.79	24.29 ± 0.2	23.63 ± 1.88
	(18/2)	(11/2)	(400/2)	(300/2)	(10/2)
4	34.27 ± 1.57	$\mathbf{34.25 \pm 1.59}$	34.31 ± 1.607	32.43 ± 0.90	30.92 ± 10.47
	(25/1)	(8/1)	(51/1)	(140/1)	(6/1)
5	23.20 ± 2.28	$\mathbf{23.14 \pm 2.34}$	23.37 ± 2.11	24.67 ± 2.60	23.31 ± 2.24
	(119/3)	(16/3)	(600/3)	(1150/3)	(51/3)
6	15.94 ± 3.37	$\mathbf{15.48 \pm 3.25}$	15.71 ± 3.20	15.58 ± 3.41	15.60 ± 3.41
	(63/12)	(16/12)	(500/12)	(170/12)	(21/12)
7	6.42 ± 0.82	2.78 ± 0.56	$\mathbf{2.68 \pm 0.54}$	2.99 ± 0.48	2.72 ± 0.53
	(7109/1)	(447/2.6)	(179000/2.6)	(280000/2.6)	(20001/2.6)
8	9.09 ± 0.89	3.09 ± 0.42	4.68 ± 0.7	$\mathbf{2.85 \pm 0.33}$	3.83 ± 0.52
	(514/3)	(37/3)	(820/3)	(510/3)	(151/3)
9	14.71 ± 0.75	$\mathbf{10.79 \pm 0.67}$	11.43 ± 0.72	11.67 ± 0.68	10.92 ± 0.72
	(816/6)	(72/6)	(1250/6)	(1400/6)	(501/6)
10	4.53 ± 2.34	4.55 ± 2.35	$\mathbf{4.48 \pm 2.33}$	4.49 ± 2.21	4.59 ± 2.34
	(65/1)	(28/1)	(100/1)	(200/1)	(21/1)
11	23.53 ± 1.82	22.96 ± 1.21	22.82 ± 1.81	$\mathbf{21.28 \pm 0.67}$	20.20 ± 7.17
	(5/1)	(1/1)	(1.19/1)	(12/1)	(1/1)
12	2.39 ± 0.13	$\mathbf{2.36 \pm 0.13}$	2.42 ± 0.14	2.39 ± 0.13	2.56 ± 0.30
	(20/3)	(7/3)	(100/3)	(61/3)	(1/3)
13	9.53 ± 0.45	9.63 ± 0.49	9.53 ± 0.44	9.77 ± 0.35	$\mathbf{9.52 \pm 0.44}$
	(8/3.1)	(12/3.1)	(150/3.1)	(171/3.1)	(21/3.1)

Table 3: Comparison of the ν-method (right column) against the best of the 7 methods taken from the benchmark webpage (see text) on the 13 benchmark datasets. The middle column shows the results for SVM from the same webpage.

	Best of 7	SVM	ν-meth.
Banana	LP_Reg-Ada		
	10.73 ± 0.43	11.53 ± 0.66	10.67 ± 0.53
B.Canc.	KFD		
	24.77 ± 4.63	26.04 ± 4.74	25.35 ± 4.24
Diabet.	KFD		
	23.21 ± 1.63	23.53 ± 1.73	23.60 ± 1.82
F.Solar	SVM-RBF		
	32.43 ± 1.82	32.43 ± 1.82	34.25 ± 1.59
German	SVM-RBF		
	23.61 ± 2.07	23.61 ± 2.07	23.14 ± 2.08
Heart	SVM-RBF		
	15.95 ± 3.26	15.95 ± 3.26	15.48 ± 3.25
Image	ADA_Reg		
	2.67 ± 0.61	2.96 ± 0.6	2.78 ± 0.56
Ringn.	ADA_Reg		
	1.58 ± 0.12	1.66 ± 0.2	3.09 ± 0.42
Splice	ADA_Reg		
	9.50 ± 0.65	10.88 ± 0.66	10.79 ± 0.67
Thyroid	KFD		
	4.20 ± 2.07	4.80 ± 2.19	4.55 ± 2.35
Titanic	SVM-RBF		
	22.42 ± 1.02	22.42 ± 1.02	22.96 ± 1.21
Twon.	KFD		
	2.61 ± 0.15	2.96 ± 0.23	2.36 ± 0.13
Wavef.	KFD		
	9.86 ± 0.44	9.88 ± 0.44	9.63 ± 0.49

7.2 Experiments on face detection

This section reports the analysis we carried out on the problem of face detection, to the purpose of evaluating the effectiveness of the ν-method in comparison to SVMs. The structure of the experiments, including model selection and error estimation, follows the one reported above. The data we consider are image patches, we represent them in the simplest way unfolding the patch matrix in a one-dimensional vector of integer values – the gray levels. All the images of the two datasets are 19×19, thus the size of our data is 361.

The first dataset we use for training and testing is the well known CBCL dataset for frontal faces[4] composed of thousands of small images of positive and negative examples of size. The face images obtained from this benchmark are clean and nicely registered.

The second dataset we consider is made of low quality images acquired by a monitoring system installed in our department[5]. The data are very different from the previous set since they have been obtained from video frames (therefore they are more noisy and often blurred by motion), faces have not been registered, gray values have not been normalized. The RBF

[4] Available for download at `http://cbcl.mit.edu/software-datasets/FaceData2.html`.
[5] The dataset is available upon request.

#TRAIN + #TEST **CLASSIFIER**	**600+1400**	**700+1300**	**800+1200**
RBF-SVM	2.41 ± 1.39 $\sigma = 800$ $C = 1$	1.99 ± 0.82 $\sigma = 1000$ $C = 0.8$	1.60 ± 0.71 $\sigma = 1000$ $C = 0.8$
ν-method	$\mathbf{1.63 \pm 0.32}$ $\sigma = 341$ $t = 85$	$\mathbf{1.53 \pm 0.33}$ $\sigma = 341$ $t = 89$	$\mathbf{1.48 \pm 0.34}$ $\sigma = 300$ $t = 59$

Table 4: Average and standard deviation of the classification error of SVM and ν-method trained on training sets of increasing size. The data are the CBCL-MIT benchmark dataset of frontal faces (see text).

kernel may take into account slight data misalignment due to the intra-class variability, but in this case model selection is more crucial and the choice of an appropriate parameter for the kernel is advisable.

The experiments performed on these two sets follow the structure discussed in the previous section. Starting from the original set of data, in both cases we randomly extract 2000 data that we use for most of our experiments: for a fixed training set size we generate 50 resamplings of training and test data. Then we vary the training set size from 600 (300+300) to 800 (400+400) training examples. The results obtained are reported in Table 4 and Table 5. The tables show a comparison between the ν-method and SVM as the size of the training set grows. The results obtained are slightly different: while on the CBCL dataset the ν-method performance is clearly above the SVM classifier, in the second set of data the performance of the

#TRAIN + #TEST CLASSIFIER	**600+1400**	**700+1300**	**800+1200**
RBF-SVM	**3.99 ± 1.21** $\sigma = 570$ $C = 2$	**3.90 ± 0.92** $\sigma = 550$ $C = 1$	3.8 ± 0.58 $\sigma = 550$ $C = 1$
ν-method	4.36 ± 0.53 $\sigma = 250$ $t = 67$	4.19 ± 0.50 $\sigma = 180$ $t = 39$	**3.69 ± 0.54** $\sigma = 200$ $t = 57$

Table 5: Average and standard deviation of the classification error of SVM and ν-method trained on training sets of increasing size. The data are a have been acquired by a monitoring system developed in our laboratory (see text).

ν-method increases as the training set size grows.

At the end of this evaluation process we retrained the ν-method on the whole set of 2000 data and again tuned the parameters with KCV obtaining $\sigma = 200$ and $t = 58$. Then we used this classifier to test a batch of newly acquired data (the size of this new test set is of 6000 images) obtaining a classification error of 3.67%. These results confirm the generalization ability of the algorithm. For completeness we report that the SVM classifier trained and tuned on the whole dataset of above — $\sigma = 600$ and $C = 1$ — lead to an error rate of 3.92%.

8 Conclusion

In this paper we present and discuss several spectral algorithms for supervised learning. Starting from the standard regularized least squares we show that

a number of methods from the inverse problems theory lead to consistent learning algorithms. We provide a unifying theoretical analysis based on the concept of filter function showing that these algorithms, which differ from the computational viewpoint, are all consistent kernel methods. The iterative methods – like the ν-method and the iterative Landweber – and projections methods – like spectral cut-off or PCA – give rise to regularized learning algorithms in which the regularization parameter is the number of iterations or the number of dimensions in the projection, respectively.

We report an extensive experimental analysis on a number of datasets showing that all the proposed spectral algorithms are a good alternative, in terms of generalization performances and computational efficiency, to state of the art algorithms for classification, like SVM and adaboost. One of the main advantages of the methods we propose is their simplicity: each spectral algorithm is an easy-to-use linear method whose implementation is straightforward. Indeed our experience suggests that this helps dealing with overfitting in a transparent way and make the model selection step easier. In particular, the search for the best choice of the regularization parameter in iterative schemes is naturally embedded in the iteration procedure.

Acknowledgments

We would like to thank S. Pereverzev for useful discussions and suggestions and A. Destrero for providing the faces dataset. This work has been partially supported by the FIRB project LEAP RBIN04PARL and by the EU Integrated Project Health-e-Child IST-2004-027749.

References

Aronszajn, N. (1950). Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, *68*, 337–404.

Bartlett, P. L., Jordan, M. I., & McAuliffe, J. D. (2006). Convexity, classification, and risk bounds. *J. Amer. Statist. Assoc.*, *101*(473), 138–156.

Bauer, F., Pereverzev, S., , & Rosasco, L. (2006). On regularization algorithms in learning theory. *Journal of Complexity,*. (In Press, doi: 10.1016/j.jco.2006.07.001 , Online 19 October 2006,)

Bertero, M., & Boccacci, P. (1998). *Introduction to inverse problems in imaging*. Bristol: IOP Publishing.

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

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, M.I.T. (available at http://cbcl.mit.edu/projects/cbcl/publications/ps/MIT-CSAIL-TR-2006-062.pdf)

Caponnetto, A., & De Vito, E. (2006). Optimal rates for regularized least-squares algorithm. *Found. Comput. Math.* (In Press, DOI 10.1007/s10208-006-0196-8, Online August 2006)

Chapelle, O., Weston, J., & Scholkopf, B. (2003). Cluster kernels for semi-supervised learning. In *Neural information processing systems 15* (p. 585-592).

De Vito, E., Caponnetto, A., & Rosasco, L. (2005). Model selection for regularized least-squares algorithm in learning theory. *Found. Comput. Math.*, *5*(1), 59–85.

De Vito, E., Rosasco, L., & Caponnetto, A. (2006). Discretization error analysis for Tikhonov regularization. *Anal. Appl.*, *4*(1), 81–99.

De Vito, E., Rosasco, L., Caponnetto, A., De Giovannini, U., & Odone, F.

(2005, May). Learning from examples as an inverse problem. *Journal of Machine Learning Research*, *6*, 883–904.

De Vito, E., Rosasco, L., & Verri, A. (2005). *Spectral methods for regularization in learning theory* (Tech. Rep. No. Technical Rerport). DISI, Universitá degli Studi di Genova, Italy.

Engl, H. W., Hanke, M., & Neubauer, A. (1996). *Regularization of inverse problems* (Vol. 375). Dordrecht: Kluwer Academic Publishers Group.

Evgeniou, T., Pontil, M., & Poggio, T. (2000). Regularization networks and support vector machines. *Adv. Comp. Math.*, *13*, 1-50.

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

Golub, G. H., & Van Loan, C. F. (1996). *Matrix computations* (Third ed.). Baltimore, MD: Johns Hopkins University Press.

Hastie, S., T. andRosset, Tibshirani, R., & Zhu, J. (2004). The entire regularization path for the support vector machine. *JMLR*, *5*, 1391–1415.

Hastie, T., Tibshirani, R., & Friedman, J. (2001). *The elements of statistical learning*. New York: Springer.

Li, W., Lee, K.-H., & Leung, K.-S. (2007). Large-scale rlsc learning without

agony. In *Icml '07: Proceedings of the 24th international conference on machine learning* (pp. 529–536). New York, NY, USA: ACM Press.

Micchelli, C. A., Xu, Y., & H., Z. (2006). Universal kernels. *JMLR*, *7*, 2651–2667.

Ong, C., & Canu, S. (2004). *Regularization by early stopping* (Tech. Rep.). Computer Sciences Laboratory, RSISE, ANU. (available at http://asi.insa-rouen.fr/ scanu/)

Ong, C., X., M., Canu, S., & Smola, A. (2004). Learning with non-positive kernels. In *Proceedings of the 21 st international conference on machine learning, banff, canada, 2004.* (available at http://www.aicml.cs.ualberta.ca/_banff04/icml/pages/papers/392.pdf)

Poggio, T., & Girosi, F. (1992). A theory of networks for approximation and learning. In C. Lau (Ed.), *Foundation of neural networks* (p. 91-106). Piscataway, N.J.: IEEE Press.

Poggio, T., Rifkin, R., Mukherjee, S., & Niyogi, P. (2004). General conditions for predictivity in le arning theory. *Nature*, *428*, 419-422.

Pontil, M., & Verri, A. (1998). Properties of support vector machines. *Neural Computation*, *10*, 977–996.

Rakhlin, A., Mukherjee, S., & Poggio, T. (2005). Stability results in learning

theory. *Analysis and Applications*, *3*, 397-419.

Schölkopf, B., & Smola, A. (2002). *Learning with kernels*. Cambridge, MA: MIT Press.

Smale, S., & Zhou, D. (2005a). *Learning theory estimates via integral operators and their approximations* (Tech. Rep.). Toyota Technological Institute at Chicago, USA. (available at http://ttic.uchicago.edu/ smale/papers/sampIII5412.pdf)

Smale, S., & Zhou, D.-X. (2004). Shannon sampling and function reconstruction from point values. *Bull. Amer. Math. Soc. (N.S.)*, *41*(3), 279–305 (electronic).

Smale, S., & Zhou, D.-X. (2005b). Shannon sampling. II. Connections to learning theory. *Appl. Comput. Harmon. Anal.*, *19*(3), 285–302.

Smola, A., & Kondor, R. (2003). Kernels and regularization on graphs. In *Colt*.

Tikhonov, A., & Arsenin, V. (1977). *Solutions of ill posed problems*. Washington, D.C.: W. H. Winston.

Vapnik, V. (1982). *Estimation of dependences based on empirical data*. New York: Springer-Verlag. (Translated from the Russian by Samuel Kotz)

Vapnik, V. N. (1998). *Statistical learning theory*. New York: John Wiley &

Sons Inc.

Wahba, G. (1990). *Spline models for observational data* (Vol. 59). Philadelphia, PA: SIAM.

Wu, Q., Ying, Y., & Zhou, D.-X. (2006). Learning rates of least-square regularized regression. *Found. Comput. Math.*, *6*(2), 171–192.

Yao, Y., Rosasco, L., & Caponnetto, A. (2007). On early stopping in gradient descent learning,. *Constructive Approximation*. (In Press, available at http://math.berkeley.edu/ yao/publications/earlystop.pdf)

Zhang, T., & Ando, R. (2006). Analysis of spectral kernel design based semi-supervised learning. In Y. Weiss, B. Schölkopf, & J. Platt (Eds.), *Advances in neural information processing systems 18* (pp. 1601–1608). Cambridge, MA: MIT Press.

Zhu, X., Kandola, J., Ghahramani, Z., , & Lafferty, J. (2005). Nonparametric transforms of graph kernels for semi-supervised learning. In *Neural information processing systems 17*.