Research Note RN/03/08

Department of Computer Science, University College London

On Learning Vector−Valued Functions

Charles A. Micchelli[1]

Department of Mathematics and Statistics
State University of New York
The University at Albany
1400 Washington Avenue, Albany, NY, 12222, USA
E-mail: *cam at math dot albany dot edu*

and

Massimiliano Pontil

Department of Computer Sciences
University College London
Gower Street, London WC1E, England, UK
E-mail: *m dot pontil at cs dot ucl dot ac dot uk*

February 18, 2003 (revised on July 14, 2003)

Abstract

In this paper, we provide a study of learning in a Hilbert space of *vector−valued* functions. We motivate the need for extending learning theory of scalar−valued functions by practical considerations and establish some basic results for learning vector−valued functions which should prove useful in applications. Specifically, we allow an output space $\mathcal{Y}$ to be a Hilbert space and we consider a reproducing kernel Hilbert space of functions whose values lie in $\mathcal{Y}$. In this setting, we derive the form of the minimal norm interpolant to a finite set of data and apply it to study some regularization functionals which are important in learning theory. We consider specific examples of such functionals corresponding to multiple−output regularization networks and support vector machines, both for regression and classification. Finally, we provide classes of operator−valued kernels of the dot product and translation invariant type.

[1]Partially supported by NSF Grant No. ITR-0312113.

1 Introduction

The problem of computing a function from empirical data is addressed in several areas of mathematics and engineering. Depending on the context, this problem goes under the name of function estimation (statistics), function learning (machine learning theory), function approximation and interpolation (approximation theory), among others. The type of functions typically studied are real−valued functions (in learning theory, the related binary classification problem is often treated as a special case). There is a large literature on the subject. We recommend [11, 14, 15, 27, 28] and references therein.

In this work we address the problem of computing functions whose range is in a Hilbert space, discuss ideas within the perspective of learning theory and elaborate on their connections to interpolation and optimal estimation. Despite its importance, learning vector−values functions has been only marginally studied within the learning theory community and this paper is a first attempt to set down a framework to study this problem.

We focus on Hilbert spaces of vector−valued functions which admit a reproducing kernel [3]. In the scalar case these spaces have received a considerable attention over the past few years in machine learning theory due to the successful application of kernel-based learning methods to complex data, ranging from images, text data, speech data, biological data, among others, see for example [14, 24, 27] and references therein. In Section 2 we outline the theory of reproducing kernel Hilbert spaces (RKHS) of vector−valued functions. These RKHS admit a kernel with values which are bounded linear operators on the output space. They have been studied by Burbea and Masani [10], but only in the context of complex analysis and used recently for the solution of partial differential equations by Amodei [2]. Section 3 treats the problem of minimal norm interpolation (MNI) in the context of RKHS. MNI plays a central role in many approaches to function estimation and so we shall highlight its relation to learning. In particular, in Section 4 we use MNI to resolve the form of the minimizer of regularization functionals in the context of vector−valued functions. In Section 5 we discuss the form of operator−valued kernels which are either of the dot product or translation invariant form as they are often used in learning. Finally, in Section 6 we describe examples where we feel there is practical need for vector−valued learning as well as report on numerical experiments which highlight the advantages of the proposed methods.

2 Reproducing kernel Hilbert spaces of vector−valued functions

Let $\mathcal{Y}$ be a real Hilbert space with inner product $(\cdot,\cdot)$, $\mathcal{X}$ a set, and $\mathcal{H}$ a linear space of functions on $\mathcal{X}$ with values in $\mathcal{Y}$. We assume that $\mathcal{H}$ is also a Hilbert space with inner product $\langle\cdot,\cdot\rangle$.

Definition 2.1 *We say that $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS) when for any $y \in \mathcal{Y}$ and $x \in \mathcal{X}$ the linear functional which maps $f \in \mathcal{H}$ to $(y, f(x))$ is continuous.*

In this case, according to the Riesz Lemma (see, e.g., [1]), there is, for every $x \in \mathcal{X}$ and $y \in \mathcal{Y}$ a function $K(x|y) \in \mathcal{H}$ such that, for all $f \in \mathcal{H}$,

$$(y, f(x)) = \langle K(x|y), f\rangle.$$

Since $K(x|y)$ is linear in y, we write $K(x|y) = K_x y$ where $K_x : \mathcal{Y} \to \mathcal{H}$ is a linear operator. The above equation can be now rewritten as

$$(y, f(x)) = \langle K_x y, f \rangle. \tag{2.1}$$

For every $x, t \in \mathcal{X}$ we also introduce the linear operator $K(x,t) : \mathcal{Y} \to \mathcal{Y}$ defined, for every $y \in \mathcal{Y}$, by

$$K(x,t)y := (K_t y)(x). \tag{2.2}$$

We say that $\mathcal{H}$ is *normal* provided there does not exist $(x, y) \in \mathcal{X} \times (\mathcal{Y} \backslash \{0\})$ such that the linear functional $(y, f(x)) = 0$ for all $f \in \mathcal{H}$.

In the proposition below we state the main properties of the function K. To this end, we let $\mathcal{L}(\mathcal{Y})$ be the set of all bounded linear operators from $\mathcal{Y}$ into itself and, for every $A \in \mathcal{L}(\mathcal{Y})$, we denote by A^* its adjoint. We also use $\mathcal{L}_+(\mathcal{Y})$ to denote the cone of nonnegative bounded linear operators, i.e. $A \in \mathcal{L}_+(\mathcal{Y})$ provided that, for every $y \in \mathcal{Y}$, $(y, Ay) \geq 0$. When this inequality is strict for all $y \neq 0$ we say A is positive definite. Finally, we denote by $\mathbb{N}_m$ the set of positive integers up to and including m.

Proposition 2.1 *If $K(x,t)$ is defined, for every $x, t \in X$, by equation (2.2) and K_x is given by equation (2.1) the kernel K satisfies, for every $x, t \in \mathcal{X}$, the following properties:*

(a) *For every $y, z \in \mathcal{Y}$, we have that*

$$(y, K(x,t)z) = \langle K_t z, K_x y \rangle. \tag{2.3}$$

(b) *$K(x,t) \in \mathcal{L}(\mathcal{Y})$, $K(x,t) = K(t,x)^*$, and $K(x,x) \in \mathcal{L}_+(\mathcal{Y})$. Moreover, $K(x,x)$ is positive definite for all $x \in \mathcal{X}$ if and only if $\mathcal{H}$ is normal.*

(c) *For any $m \in \mathbb{N}$, $\{x_j : j \in \mathbb{N}_m\} \subseteq \mathcal{X}$, $\{y_j : j \in \mathbb{N}_m\} \subseteq \mathcal{Y}$ we have that*

$$\sum_{j,\ell \in \mathbb{N}_m} (y_j, K(x_j, x_\ell) y_\ell) \geq 0. \tag{2.4}$$

(d) $\|K_x\| = \|K(x,x)\|^{\frac{1}{2}}$.

(e) $\|K(x,t)\| \leq \|K(x,x)\|^{\frac{1}{2}} \|K(t,t)\|^{\frac{1}{2}}$.

(f) *For every $f \in \mathcal{H}$ and $x \in \mathcal{X}$ we have that*

$$\|f(x)\| \leq \|f\| \|K(x,x)\|^{\frac{1}{2}}.$$

Proof. We prove (a) by merely choosing $f = K_t z$ in equation (2.1) to obtain that

$$\langle K_x y, K_t z \rangle = (y, (K_t z)(x)) = (y, K(x,t)z).$$

Consequently, from this equation, we conclude that $K(x,t)$ admits an algebraic adjoint $K(t,x)$ defined everywhere on $\mathcal{Y}$ and so the uniform boundness principle [1] implies that $K(x,t) \in \mathcal{L}(\mathcal{Y})$ and $K(x,t) = K(t,x)^*$, see also [1, p. 48]. Moreover, choosing $t = x$ in

equation (2.3) proves that $K(x,x) \in \mathcal{L}_+(\mathcal{Y})$. As for the positive definiteness of $K(x,x)$ merely use equations (2.1) and (2.3). These remarks prove (b) . As for (c), we again use equation (2.3) to get

$$\sum_{j,\ell \in \mathbb{N}_m} (y_j, K(x_j, x_\ell) y_\ell) = \sum_{j,\ell \in \mathbb{N}_m} \langle K_{x_j} y_j, K_{x_\ell} y_\ell \rangle = \| \sum_{j \in \mathbb{N}_m} K_{x_j} y_j \|^2 \geq 0.$$

For the proof of (d) we choose $y \in \mathcal{Y}$ and observe that

$$\|K_x y\|^2 = (y, K(x,x)y) \leq \|y\| \|K(x,x)y\| \leq \|y\|^2 \|K(x,x)\|$$

which implies that $\|K_x\| \leq \|K(x,x)\|^{\frac{1}{2}}$. Similarly, we have that

$$\begin{aligned} \|K(x,x)y\|^2 &= (K(x,x)y, K(x,x)y) \\ &= \langle K_x K(x,x)y, K_x y \rangle \\ &\leq \|K_x y\| \|K_x K(x,x)y\| \leq \|y\| \|K_x\|^2 \|K(x,x)y\| \end{aligned}$$

thereby implying that $\|K_x\|^2 \geq \|K(x,x)\|$, which proves (d). For the claim (e) we compute

$$\begin{aligned} \|K(x,t)y\|^2 &= (K(x,t)y, K(x,t)y) \\ &= \langle K_x K(x,t)y, K_t y \rangle \\ &\leq \|K_x K(x,t)y\| \|K_t y\| \leq \|y\| \|K_t\| \|K_x\| \|K(x,t)y\| \end{aligned}$$

which gives

$$\|K(x,t)y\| \leq \|K_x\| \|K_t\| \|y\|$$

and establishes (e). For our final assertion we observe, for all $y \in \mathcal{Y}$, that

$$(y, f(x)) = \langle K_x y, f \rangle \leq \|f\| \|K_x y\| \leq \|f\| \|y\| \|K_x\|$$

which implies the desired result, namely $\|f(x)\| \leq \|f\| \|K_x\|$. □

For simplicity of terminology we say that $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ is a *kernel* if it satisfies properties (a)−(c). So far we have seen that if $\mathcal{H}$ is a RKHS of vector−valued functions, there exists a kernel. In the spirit of Moore-Aronszajn's theorem for RKHS of scalar functions [3], it can be shown that a kernel determines a RKHS of vector−valued functions. We state the theorem below, however, as the proof parallels the scalar case we do not elaborate on the details.

Theorem 2.1 *If $K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{Y})$ is a kernel then there exists a unique (up to an isometry) RKHS which admits K as the reproducing kernel.*

Let us observe that in the case $\mathcal{Y} = \mathbb{R}^n$ the kernel K is a $n \times n$ matrix of scalar−valued functions. The elements of this matrix can be identified by appealing to equation (2.3). Indeed by choosing $y = e_k$, and $z = e_\ell$, $k, \ell \in \mathbb{N}_n$, these being the standard coordinate bases for $\mathbb{R}^n$, yields the formula

$$(K(x,t))_{k\ell} = \langle K_x e_k, K_t e_\ell \rangle. \tag{2.5}$$

In particular, when $n = 1$ equation (2.1) becomes $f(x) = \langle K_x, f \rangle$, which is the standard reproducing kernel property, while equation (2.3) reads $K(x,t) = \langle K_x, K_t \rangle$.

The case $\mathcal{Y} = \mathbb{R}^n$ serves also to illustrate some features of the operator−valued kernels which are not present in the scalar case. In particular, let $\mathcal{H}_\ell, \ell \in \mathbb{N}_n$ be RKHS of scalar−valued functions on $\mathcal{X}$ with kernels $K_\ell, \ell \in \mathbb{N}_n$ and define the kernel whose values are in $\mathcal{L}(\mathbb{R}^n)$ by the formula

$$D = diag(K_1, \ldots, K_n).$$

Clearly if $\{f_\ell : \ell \in \mathbb{N}_n\}, \{g_\ell : \ell \in \mathbb{N}_n\} \subseteq \mathcal{H}$ we have that

$$\langle f, g \rangle = \sum_{\ell \in \mathbb{N}_n} \langle f_\ell, g_\ell \rangle_\ell$$

where $\langle \cdot, \cdot \rangle_\ell$ is the inner product in the RKHS of scalar functions with kernel K_ℓ. Diagonal kernels can be effectively used to generate a wide variety of operator−valued kernels which have the flexibility needed for learning. We have in mind the following construction. For every set $\{A_j : j \in \mathbb{N}_m\}$ of $r \times n$ matrices and $\{D_j : j \in \mathbb{N}_m\}$ of diagonal kernels, the operator−valued function

$$K(x,t) = \sum_{j \in \mathbb{N}_m} A_j^* D_j(x,t) A_j, \ x, t \in \mathcal{X} \tag{2.6}$$

is a kernel. We conjecture that *all* operator−valued kernels are limits of kernels of this type. Generally, the kernel in equation (2.6) cannot be diagonalized, that is it cannot be rewritten in the form A^*DA, unless all the matrices $A_j, j \in \mathbb{N}_m$ can all be transformed into a diagonal matrix by the same matrix. For r much smaller than n, (2.6) results in *low rank kernels* which should be an effective tool for learning. In particular, in many practical situations the components of f may be *linearly related*, that is, for very $x \in \mathcal{X}$ $f(x)$ lies on a linear subspace $\mathcal{M} \subseteq \mathcal{Y}$. In this case it is desirable to use a kernel which has the property that $f(x) \in \mathcal{M}, \ x \in \mathcal{X}$ for all $f \in \mathcal{H}$. An elegant solution to this problem is to use a *low rank kernels* modeled by the output examples themselves, namely

$$K(x,t) = \sum_{j \in \mathbb{N}_m} \lambda_j y_j K_j(x,t) y_j^*$$

where λ_j are nonnegative constants and $K_j, j \in \mathbb{N}_m$ are some prescribed scalar−valued kernels.

Property (c) of Proposition 2.1 has an interesting interpretation concerning RKHS of *scalar−valued functions*. Every $f \in \mathcal{H}$ determines a function F on $\mathcal{X} \times \mathcal{Y}$ defined by

$$F(x,y) := (y, f(x)), \ x \in \mathcal{X}, y \in \mathcal{Y}. \tag{2.7}$$

We let $\mathcal{H}^1$ be the linear space of all such functions. Thus, $\mathcal{H}^1$ consists of functions which are *linear* in their second variable. We make $\mathcal{H}^1$ into a Hilbert space by choosing $\|F\| = \|f\|$.

It then follows that $\mathcal{H}^1$ is a RKHS with reproducing *scalar−valued* kernel defined, for all $(x, y), (t, z) \in \mathcal{X} \times \mathcal{Y}$, by the formula

$$K_1((x, y), (t, z)) = (y, K(x, t)z).$$

This idea is known in the statistical context, see e.g. [13, p. 138], and can be extended in the following manner. We define, for any $p \in \mathbb{N}_n$, the family of functions

$$K_p((x, y), (t, z)) := (y, K(x, t)z)^p, \ (x, y), (t, z) \in \mathcal{X} \times \mathcal{Y}$$

The lemma of Schur, see e.g. [3, p. 358], implies that K_p is a scalar kernel on $\mathcal{X} \times \mathcal{Y}$. The functions in the associated RKHS consist of scalar−valued functions which are homogeneous polynomials of degree p in their second argument. These spaces may be of practical value for learning polynomial functions of projections of vector−valued functions.

A kernel K can be realized by a mapping $\Phi : \mathcal{X} \to \mathcal{L}(\mathcal{W}, \mathcal{Y})$ where $\mathcal{W}$ is a Hilbert space, by the formula

$$K(x, t) = \Phi(x)\Phi^*(t), \ x, t \in \mathcal{X}$$

and functions f in $\mathcal{H}$ can be represented as $f = \Phi w$ for some $w \in \mathcal{W}$, so that $\|f\|_{\mathcal{H}} = \|w\|_{\mathcal{W}}$. Moreover, when $\mathcal{H}$ is separable we can choose $\mathcal{W}$ to be the separable Hilbert space of square summable sequences, [1]. In the scalar case, $\mathcal{Y} = \mathbb{R}$, Φ is referred to in learning theory as the *feature map* and it is central in developing kernel−based learning algorithms, see e.g. [14, 24].

3 Minimal norm interpolation

In this section we turn our attention to the problem of minimal norm interpolation of vector−valued functions within the RKHS framework developed above. This problem consists in finding, among *all* functions in $\mathcal{H}$ which interpolate a given set of points, a function with minimum norm. We will see later in the paper that the minimal norm interpolation problem plays a central role in characterizing regularization approaches to learning.

Definition 3.1 *For distinct points $\{x_j : j \in \mathbb{N}_m\}$ we say that the linear functionals defined for $f \in \mathcal{H}$ as $L_{x_j} f := f(x_j)$, $j \in \mathbb{N}_m$ are* linearly independent *if and only if there does not exist $\{c_j : j \in \mathbb{N}_m\} \subseteq \mathcal{Y}$ (not all zero) such that for all $f \in \mathcal{H}$*

$$\sum_{j \in \mathbb{N}_m} (c_j, f(x_j)) = 0. \tag{3.1}$$

Lemma 3.1 *The functionals $\{L_{x_j} : j \in \mathbb{N}_m\}$ are linearly independent if and only if for any $\{y_j : j \in \mathbb{N}_m\} \subseteq \mathcal{Y}$ there are unique $\{c_j : j \in \mathbb{N}_m\} \subseteq \mathcal{Y}$ such that*

$$\sum_{\ell \in \mathbb{N}_m} K(x_j, x_\ell)c_\ell = y_j, \ j \in \mathbb{N}_m. \tag{3.2}$$

PROOF. We denote by $\mathcal{Y}^m$ the $m-$th Cartesian product of $\mathcal{Y}$. We make $\mathcal{Y}^m$ into a Hilbert space by defining for every $c = (c_j : j \in \mathbb{N}_m) \in \mathcal{Y}^m$ and $d = (d_j : j \in \mathbb{N}_m) \in \mathcal{Y}^m$ their inner product $\langle c, d\rangle := \sum_{j\in\mathbb{N}_m}(c_j, d_j)$. Let us consider the bounded linear operator $A : \mathcal{H} \to \mathcal{Y}^m$, defined for $f \in \mathcal{H}$ by

$$Af = (f(x_j) : j \in \mathbb{N}_m).$$

Therefore, by construction, c satisfies equation (3.1) when $c \in Ker(A^*)$ and hence the set of linear functionals $\{L_{x_j} : j \in \mathbb{N}_m\}$ are linearly independent if and only if $Ker(A^*) = \{0\}$. Since $Ker(A^*) = Ran(A)^\perp$, see [1], this is equivalent to the condition that $Ran(A) = \mathcal{Y}^m$. From the reproducing kernel property we can identify A^*, by computing for $c \in \mathcal{Y}^m$ and $f \in \mathcal{H}$

$$\langle c, Af\rangle = \sum_{j\in\mathbb{N}_m}(c_j, f(x_j)) = \sum_{j\in\mathbb{N}_m}\langle K_{x_j}c_j, f\rangle.$$

Thus, we have established that $A^*c = \sum_{j\in\mathbb{N}_m} K_{x_j}c_j$. We now consider the symmetric bounded linear operators $B := AA^* : \mathcal{Y}^m \to \mathcal{Y}^m$ which we identify for $c = (c_j : j \in \mathbb{N}_m) \in \mathcal{Y}^m$ as

$$Bc = (\sum_{j\in\mathbb{N}_m} K(x_\ell, x_j)c_j : \ell \in \mathbb{N}_m).$$

Consequently, (3.2) can be equivalently written as $Bc = y$ and so this equation means that $y \in Ran(B) = Ker(A^*)^\perp$. Since it can be verified that $Ker(A^*) = Ker(B)$ we have shown that $Ker(A^*) = \{0\}$ if and only if the linear functionals $\{L_{x_j} : j \in \mathbb{N}_m\}$ are linearly independent and the result follows.

□

Theorem 3.1 *If the linear functionals $L_{x_j}f = f(x_j), f \in \mathcal{H},\ j \in \mathbb{N}_m$ are linearly independent then the unique solution to the variational problem*

$$\min\left\{\|f\|^2 : f(x_j) = y_j,\ j \in \mathbb{N}_m\right\} \tag{3.3}$$

is given by

$$\hat{f} = \sum_{j\in\mathbb{N}_m} K_{x_j}c_j$$

where $\{c_j : j \in \mathbb{N}_m\} \subseteq \mathcal{Y}$ is the unique solution of the linear system of equations

$$\sum_{\ell\in\mathbb{N}_m} K(x_j, x_\ell)c_\ell = y_j,\ j \in \mathbb{N}_m. \tag{3.4}$$

PROOF. Let f be any element of $\mathcal{H}$ such that $f(x_j) = y_j,\ j \in \mathbb{N}_m$. This function always exists since we have shown that the operator A maps onto $\mathcal{Y}^m$. We set $g := f - \hat{f}$ and observe that

$$\|f\|^2 = \|g + \hat{f}\|^2 = \|g\|^2 + 2\langle\hat{f}, g\rangle + \|\hat{f}\|^2.$$

However, since $g(x_j) = 0,\ j \in \mathbb{N}_m$ we obtain that

$$\langle\hat{f}, g\rangle = \sum_{j\in\mathbb{N}_m}\langle K_{x_j}c_j, g\rangle = \sum_{j\in\mathbb{N}_m}(c_j, g(x_j)) = 0.$$

It follows that

$$\|f\|^2 = \|g\|^2 + \|\hat{f}\|^2 \geq \|\hat{f}\|^2$$

and we conclude that $\hat{f}$ is the unique solution to (3.3).

□

When the set of linear functionals $\{L_{x_j} : j \in \mathbb{N}_m\}$ are dependent, generally data $y \in \mathcal{Y}^m$ may not admit an interpolant. Thus, the variational problem in Theorem 3.1 requires that $y \in Ran(A)$. Since

$$Ran(A) = Ker(A^*)^\perp = Ker(B)^\perp = Ran(B)$$

we see that if $y \in Ran(A)$, that is, when the data admit an interpolant in $\mathcal{H}$, then the system of equations (3.2) has a (generally not unique) solution and so the function in equation (3.4) is still solution of the extremal problem. Hence, we have proved the following result.

Theorem 3.2 *If $y \in Ran(A)$ the minimum of problem (3.3) is unique and admits the form $\hat{f} = \sum_{j\in\mathbb{N}_m} K_{x_j} c_j$, where the coefficients $\{c_j : j \in \mathbb{N}_m\}$ solve the linear system of equations*

$$\sum_{\ell\in\mathbb{N}_m} K(x_j, x_\ell)c_\ell = y_j,\ j \in \mathbb{N}_m.$$

An alternative approach to proving the above result is to "trim" the set of linear functionals $\{L_{x_j} : j \in \mathbb{N}_m\}$ to a maximally linearly independent set and then apply Theorem 3.1 to this subset of linear functionals. This approach, of course, requires that $y \in Ran(A)$ as well.

4 Regularization

We begin this section the approximation scheme that arises from the minimization of the functional

$$E(f) := \sum_{j\in\mathbb{N}_m} \|y_j - f(x_j)\|^2 + \mu\|f\|^2 \tag{4.1}$$

where μ is a fixed positive constant and $\{(x_j, y_j) : j \in \mathbb{N}_m\} \subseteq \mathcal{X} \times \mathcal{Y}$. Our initial remarks shall prepare for the general case (4.5) treated later. Problem (4.1) is a special form of regularization functionals introduced by Tikhonov, Ivanov and others to solve ill-posed problems, [26]. Its application to learning is discussed, for example, in [15]. A justification, from the theory of optimal estimation, that regularization is the *optimal algorithm* to learn a function from noisy data is given in [21].

Theorem 4.1 *If $\hat{f}$ minimizes E in $\mathcal{H}$, it is unique and has the form*

$$\hat{f} = \sum_{j\in\mathbb{N}_m} K_{x_j} c_j \tag{4.2}$$

where the coefficients $\{c_j : j \in \mathbb{N}_m\} \subseteq \mathcal{Y}$ are the unique solution of the linear equations

$$\sum_{\ell\in\mathbb{N}_m} (K(x_j, x_\ell) + \mu\delta_{j\ell})c_\ell = y_j,\ j \in \mathbb{N}_m. \tag{4.3}$$

PROOF. The proof is similar to that of Theorem 3.1. We set $g = f - \hat{f}$ and note that

$$E(f) = E(\hat{f}) + \sum_{j\in\mathbb{N}_m} \|g(x_j)\|^2 - 2\sum_{j\in\mathbb{N}_m} (y_j - \hat{f}(x_j), g(x_j)) + 2\mu\langle\hat{f}, g\rangle + \mu\|g\|^2.$$

Using equations (2.1), (4.2), and (4.3) gives the equations

$$\begin{aligned} \langle\hat{f}, g\rangle &= \sum_{j\in\mathbb{N}_m} (c_j, g(x_j)) \\ \sum_{j\in\mathbb{N}_m} (y_j - \hat{f}(x_j), g(x_j)) &= \mu \sum_{j\in\mathbb{N}_m} (c_j, g(x_j)) \end{aligned}$$

and so it follows that

$$E(f) = E(\hat{f}) + \sum_{j\in\mathbb{N}_m} \|g(x_j)\|^2 + \mu\|g\|^2$$

from which we conclude that $\hat{f}$ is the unique minimizer of E. □

The representation theorem above embodies the fact that regularization is also a MNI problem in the space $\mathcal{H}\times\mathcal{Y}^m$. In fact, we can make this space into a Hilbert space by setting, for every $f \in \mathcal{H}$ and $\xi = (\xi_j : j \in \mathbb{N}_m) \in \mathcal{Y}^m$

$$\|(f, \xi)\|^2 := \sum_{j\in\mathbb{N}_m} \|\xi_j\|^2 + \mu\|f\|^2$$

and note that the regularization procedure (4.1) is equivalent to the MNI problem defined in Theorem 3.2 with linear functionals defined at $(f, \xi) \in \mathcal{H} \times \mathcal{Y}^m$ by the equation $L_{x_j}(f, \xi) := f(x_j) + \xi_j, j \in \mathbb{N}_m$ corresponding to data $y = (y_j : j \in \mathbb{N}_m)$.

Let us consider again the case that $\mathcal{Y} = \mathbb{R}^n$. The linear system of equations (4.3) reads

$$(G + \mu I)c = y \tag{4.4}$$

and we view G as a $m \times m$ block matrix, where each block is a $n \times n$ matrix (so G is a $mn \times mn$ scalar matrix), and $c = (c_j : j \in \mathbb{N}_m)$, $y = (y_j : j \in \mathbb{N}_m)$ are vectors in $\mathbb{R}^{mn}$. Specifically, the $j-$th, $k-$th block of G is $G_{jk} = K(x_j, x_k)$, $j, k \in \mathbb{N}_m$. Proposition 1 assures that G is symmetric and nonnegative definite. Moreover the diagonal elements of G are positive semi-definite. There is a wide variety of circumstances where the linear systems of equations (4.4) can be effectively solved. Specifically, as before we choose the kernel

$$K := A^T D A$$

where $D := diag(K_1, \ldots, K_n)$ and each $K_\ell, \ell \in \mathbb{N}_n$ is a prescribed scalar$-$valued kernel, and A is a nonsingular $n \times n$ matrix. In this case the linear system of equations (4.4) becomes

$$\tilde{A}^T \tilde{D} \tilde{A} c = y$$

where $\tilde{A}$ is the $m \times m$ block diagonal matrix whose $n \times n$ block elements are formed by the matrix A and $\tilde{D}$ is the $m\times m$ matrix whose $j-$th, $k-$th block is $\tilde{D}_{jk} := D(x_j, x_k), j, k \in \mathbb{N}_m$. When A is *upper triangular* this system of equations can be *efficiently* solved by first solving

the system $\tilde{A}\tilde{D}z = y$ and then solving the system $\tilde{A}^T c = z$. Both of these steps can be implemented by solving only $n \times n$ systems coupled with vector substitution.

We can also reformulate the solution of the regularization problem (4.1) in terms of the *feature map*. Indeed, if $\Phi : \mathcal{X} \to \mathcal{L}(\mathcal{W}, \mathcal{Y})$ is any such map, where $\mathcal{W}$ is some Hilbert space, the solution which minimizes (4.4) has the form $f(x) = \Phi(x)w$ where $w \in \mathcal{Y}$ is given by the formula

$$w = \mu \left(\sum_{j \in \mathbb{N}_m} \Phi^*(x_j)\Phi(x_j) + \mu I \right)^{-1} \sum_{j \in \mathbb{N}_m} \Phi^*(x_j) y_j.$$

Let $V : \mathcal{Y}^m \times \mathbb{R}_+ \to \mathbb{R}$ be a prescribed function and consider the problem of minimizing the functional

$$E(f) := V\left((f(x_j) : j \in \mathbb{N}_m), \|f\|^2\right) \tag{4.5}$$

over all functions $f \in \mathcal{H}$. A special case is covered by the functional of the form

$$E(f) := \sum_{j \in \mathbb{N}_m} Q(y_j, f(x_j)) + h(\|f\|^2)$$

where $h : \mathbb{R}_+ \to \mathbb{R}_+$ is a strictly increasing function and $Q : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ is some prescribed *loss function*. In particular functional (4.1) corresponds to the choice $h(t) := \mu t^2, t \in \mathbb{R}_+$, and $Q(y, f(x)) = \|y - f(x)\|^2$. However even for this choice of h other loss functions are important in applications, see [27].

Within this general setting we provide a representation theorem for any function which minimizes the functional in equation (4.5). This result is well-known in the scalar case, see [24] and references therein. The proof below uses the representation for minimal norm interpolation presented above. This method of proof has the advantage that it can be extended to normed linear spaces which is the subject of current investigation.

Theorem 4.2 *If for every $y \in \mathcal{Y}^m$ the function $h : \mathbb{R}_+ \to \mathbb{R}_+$ defined for $t \in \mathbb{R}_+$ by $h(t) := V(y, t)$ is strictly increasing and $f_0 \in \mathcal{H}$ minimizes the functional (4.5), then $f_0 = \sum_{j \in \mathbb{N}_m} K_{x_j} c_j$ for some $\{c_j : j \in \mathbb{N}_m\} \subseteq \mathcal{Y}$. In addition, if V is strictly convex, the minimizer is unique.*

PROOF. Let f be any function such that $f(x_j) = f_0(x_j), j \in \mathbb{N}_m$ and define $y_0 := (f_0(x_j) : j \in \mathbb{N}_m)$. By the definition of f_0 we have that

$$V\left(y_0, \|f_0\|^2\right) \leq V\left(y_0, \|f\|^2\right)$$

and so

$$\|f_0\| = \min\{\|f\| : f(x_j) = f_0(x_j), j \in \mathbb{N}_m,\ f \in \mathcal{H}\}. \tag{4.6}$$

Therefore by Theorem 3.2 the result follows. When V is strictly convex the uniqueness of a global minimum of E is immediate. □

There are examples of functions V above for which the functional E given by equation (4.5) may have more than one local minimum. The question arises to what extent a *local minimum* of E has the form described in Theorem 4.2. First, let us explain what we mean by a local minimum of E. A function $f_0 \in \mathcal{H}$ is a local minimum for E provided that

there is a positive number $\epsilon > 0$ such that whenever $f \in \mathcal{H}$ satisfies $\|f_0 - f\| \leq \epsilon$ then $E(f_0) \leq E(f)$. Our first observation shows that the conclusion of Theorem 4.2 remains valid for local minima of E.

Theorem 4.3 *If V satisfies the hypotheses of Theorem 4.2 and $f_0 \in \mathcal{H}$ is a local minimum of E then $f_0 = \sum_{j \in \mathbb{N}_m} K_{x_j} c_j$ for some $\{c_j : j \in \mathbb{N}_m\} \subseteq \mathcal{Y}$.*

PROOF. If g is any function in $\mathcal{H}$ such that $g(x_j) = 0, j \in \mathbb{N}_m$ and t a real number such that $|t|\|g\| \leq \epsilon$ then

$$V\left(y_0, \|f_0\|^2\right) \leq V\left(y_0, \|f_0 + tg\|^2\right).$$

Consequently, we have that $\|f_0\|^2 \leq \|f_0 + tg\|^2$ from which it follows that $(f_0, g) = 0$. Thus, f_0 satisfies equation (4.6) and the result follows.

□

So far we obtained a representation theorem for both global or local minima of E when V is strictly increasing in its last argument. Our second observation does not require this hypothesis. Instead we shall see that if V is merely differentiable at a local minima and its partial derivative relative to its last coordinate is nonzero the conclusion of Theorems 4.2 and 4.3 remain valid. To state this observation we explain precisely what we require of V. There exist $c = (c_j : j \in \mathbb{N}_m) \in \mathcal{Y}^m$ and a nonzero constant $a \in R$ such that for any $g \in \mathcal{H}$ the derivative of the univariate function h defined for $t \in \mathbb{R}$ by the equation

$$h(t) := V((f_0(x_j) + tg(x_j) : j \in \mathbb{N}_m), \|f_0 + tg\|^2)$$

is given by

$$h'(0) = \sum_{j \in \mathbb{N}_m} (c_j, g(x_j)) + a\langle f_0, g\rangle.$$

For example, the regularization functional in equation (4.1) has this property.

Theorem 4.4 *If V is differentiable at a local minimum $f_0 \in \mathcal{H}$ of E as defined above then there exists $c = (c_j : j \in \mathbb{N}_m) \in \mathcal{Y}^m$ such that $f_0 = \sum_{j \in \mathbb{N}_m} K_{x_j} c_j$.*

PROOF. By the definition of f_0 we have that $h(0) \leq h(t)$ for all $t \in \mathbb{R}$ such that $|t|\|g\| \leq \epsilon$. Hence $h'(0) = 0$ and so we conclude that

$$\langle f_0, g\rangle = -\frac{1}{a} \sum_{j \in \mathbb{N}_m} (c_j, g(x_j)) = -\frac{1}{a} \langle \sum_{j \in \mathbb{N}_m} K_{x_j} c_j, g\rangle$$

and since g was arbitrary the result follows. □

We now discuss specific examples of loss functions which lead to quadratic programming (QP) problems. These problems are a generalization of the support vector machine (SVM) regression algorithm for scalar functions, see [27].

Example 4.1: We choose $\mathcal{Y} = \mathbb{R}^n$, $y = (y_j : j \in \mathbb{N}_m), f = (f_\ell : \ell \in \mathbb{N}_n) : \mathcal{X} \to \mathcal{Y}$ and $Q(y, f(x)) := \sum_{\ell \in \mathbb{N}_n} \max(0, |y_\ell - f_\ell(x)| - \epsilon)$, where ϵ is a positive parameter and consider the problem

$$\min \left\{ \sum_{j \in \mathbb{N}_m} Q(y_j, f(x_j)) + \mu \|f\|^2 : \ f \in \mathcal{H} \right\}. \tag{4.7}$$

Using Theorem 3.1 we transform the above problem into the *quadratic programming problem*

$$\min\left\{\sum_{\ell\in\mathbb{N}_m}(\xi_j+\xi_j^*)\cdot 1+\mu\sum_{j,\ell\in\mathbb{N}_m}(c_j,K(x_j,x_\ell)c_\ell)\right\}$$

subject, for all $j\in\mathbb{N}_m$, to the constraints on the vectors $c_j\in\mathbb{R}^n,\xi_j,\xi_j^*\in\mathbb{R}^m, j\in\mathbb{N}_m$ that

$$\begin{aligned} y_j-\textstyle\sum_{\ell\in\mathbb{N}_m}K(x_j,x_\ell)c_\ell &\leq \epsilon+\xi_j\\ \textstyle\sum_{\ell\in\mathbb{N}_m}K(x_j,x_\ell)c_\ell-y_j &\leq \epsilon+\xi_j^*\\ \xi_j,\ \xi_j^* &\geq 0\end{aligned}$$

where the inequalities are meant to hold component-wise, the symbol 1 is used for the vector in $\mathbb{R}^n$ whose all of components are equal to one, and "$\cdot$" stands for the standard inner product in $\mathbb{R}^m$. Using results from quadratic programming, see [20, pp. 123–5], this primal problem can be transformed to its *dual problem*, namely,

$$\max\left\{-\frac{1}{2}\sum_{j,\ell\in\mathbb{N}_m}(\alpha_j-\alpha_j^*,K(x_j,x_\ell)(\alpha_\ell-\alpha_\ell^*))+\sum_{j\in\mathbb{N}_m}(\alpha_j-\alpha_j^*)\cdot y_j-\epsilon(\alpha_j+\alpha_j^*)\cdot 1\right\}$$

subject, for all $j\in\mathbb{N}_m$, to the constraints on the vectors $\alpha_j,\alpha_j^*\in\mathbb{R}^n, j\in\mathbb{N}_m$ that

$$0\leq\alpha_j,\alpha_j^*\leq(2\mu)^{-1}.$$

If $\{(\hat\alpha_j,-\hat\alpha_j^*):j\in\mathbb{N}_n\}$ is a solution to the dual problem, then $\{\hat c_j:=\hat\alpha_j-\hat\alpha_j^*:j\in\mathbb{N}_m\}$ is a solution to the primal problem. The optimal parameters $\{(\xi_j,\xi_j^*):j\in\mathbb{N}_m\}$ can be obtained, for example, by setting $c=\hat c$ in the primal problem above and minimizing the resulting function only with respect to ξ,ξ^*.

Example 4.2: The setting is as in Example 4.1 but now we choose the loss function

$$Q(y,f(x)):=\max\{\max(0,|y_\ell-f_\ell(x)|-\epsilon):\ell\in\mathbb{N}_m\}.$$

In this case, problem (4.7) is equivalent to the following quadratic programming problem

$$\min\left\{\sum_{j\in\mathbb{N}_m}(\xi_j+\xi_j^*)+\sum_{j,\ell\in\mathbb{N}_m}(c_j,K(x_j,x_\ell)c_\ell)\right\}$$

subject, for all $j\in\mathbb{N}_m$, to the constraints on the vectors $c_j\in\mathbb{R}^n,\xi_j,\xi_j^*\in\mathbb{R}, j\in\mathbb{N}_m$ that

$$\begin{aligned} y_j-\textstyle\sum_{\ell\in\mathbb{N}_m}K(x_j,x_\ell)c_\ell &\leq \epsilon+\xi_j\\ \textstyle\sum_{\ell\in\mathbb{N}_m}K(x_j,x_\ell)c_\ell-y_j &\leq \epsilon+\xi_j^*\\ \xi_j,\ \xi_j^* &\geq 0.\end{aligned}$$

The corresponding dual problem is

$$\max\left\{-\frac{1}{2}\sum_{j,\ell\in\mathbb{N}_m}(\alpha_j-\alpha_j^*,K(x_j,x_\ell)(\alpha_\ell-\alpha_\ell^*))+\sum_{j\in\mathbb{N}_m}(\alpha_j-\alpha_j^*)\cdot y_j-\epsilon(\alpha_j+\alpha_j^*)\cdot 1\right\}$$

subject, for all $j\in\mathbb{N}_m$, to the constraints on the vectors $\alpha_j,\alpha_j^*\in\mathbb{R}^n, j\in\mathbb{N}_m$ that

$$(\alpha_j,1),(\alpha_j^*,1)\leq(2\mu)^{-1},\quad \alpha_j,\alpha_j^*\geq 0$$

where the symbol 1 is used for the vector in $\mathbb{R}^n$ whose all of components are equal to one, and $\cdot$ stands for the standard inner product in $\mathbb{R}^m$.

We note that when $\mathcal{Y}=\mathbb{R}$, the loss functions used in Examples 4.1 and 4.2 are the same as the loss function used in the scalar SVM regression problem, i.e. the loss $Q(y,f(x)):=\max(0,|y-f(x)|-\epsilon)$, see [27]. In particular the dual problems above coincide. Likewise, we can also consider squared versions of the loss functions in the above examples and show that the dual problems are still quadratic programming problems. Those problems reduce to the SVM-regression problem with the loss $\max(0,|y-f(x)|-\epsilon)^2$.

Examples 4.1 and 4.2 share some similarities with the multiclass classification problem considered in the context of SVM. This problem consists in learning a function from an input space $\mathcal{X}$ to the index set $\mathbb{N}_n$. To every index $\ell\in\mathbb{N}_n, n>2$ we associate a $j-$th class or category denoted by C_j. A common approach to solve such problem is by learning a vector$-$valued function $f=(f_\ell:\ell\in\mathbb{N}_n):\mathcal{X}\to\mathbb{R}^n$ and classify x in the class C_q such that $q=\operatorname{argmax}_{\ell\in\mathbb{N}_n}f_\ell(x)$. Given a training set $\{(x_j,k_j):j\in\mathbb{N}_m\}\subseteq\mathcal{X}\times\mathbb{N}_n$ with $m>n$ if, for every $j\in\mathbb{N}_m$, x_j belongs to class C_{k_j}, the output $y_j\in\mathbb{R}^n$ is a binary vector with all component equal -1 except for the k_j-th component which equals 1. Thus, the function f_k should separate as best as possible examples in the class C_k (the positive examples) from examples in the remaining classes[2] (the negative examples). In particular, if we choose $Q(y_j,f(x_j)):=\sum_{\ell\in\mathbb{N}_n}\max(0,1-f_{k_j}(x_j)+f_\ell(x_j))$, the minimization problem (4.7) leads to the multiclass formulation studied in [27, 30] while the choice

$$Q(y_j,f(x_j)):=\max\{\max(0,1-f_{k_j}(x_j)+f_\ell(x_j)):\ell\in\mathbb{N}_n\}$$

leads to the multiclass formulation of [12] (see also [6] for related work in the context of linear programming). For further information on the quadratic programming problem formulation and discussion of algorithms for the solution of these optimization problems see the above references.

5 Kernels

In this section we consider the problem of characterizing a wide variety of kernels of a form which have been found to be useful in learning theory, see [27]. We begin with a discussion of *dot product kernels*. To recall this idea we let our space $\mathcal{X}$ to be $\mathbb{R}^n$ and on $\mathcal{X}$ we put the usual Euclidean inner product $x\cdot y=\sum_{j\in\mathbb{N}_n}x_jy_j$, $x=(x_j:j\in\mathbb{N}_n)$, $y=(y_j:j\in\mathbb{N}_n)$. A

[2]The case $n=2$ (binary classification) is not included here because is merely reduces to learning a scalar$-$valued function.

dot product kernel is any function of $x \cdot y$ and a typical example is to choose $K_p(x,y) = (x \cdot y)^p$, where p is a positive integer. Our first result provides a substantial generalization. To this end we let $\mathbb{N}^m$ be the set of vectors in $\mathbb{R}^m$ whose components are nonnegative integers and if $\alpha = (\alpha_j : j \in \mathbb{N}_m) \in \mathbb{N}^m, z \in \mathbb{R}^m$ we define $z^\alpha := \prod_{j \in \mathbb{N}_m} z_j^{\alpha_j}$, $\alpha! := \prod_{j \in \mathbb{N}_m} \alpha_j!$, and $|\alpha| := \sum_{j \in \mathbb{N}_m} \alpha_j$.

Definition 5.1 *We say that a function $h : \mathbb{C}^m \to \mathcal{L}(\mathcal{Y})$ is entire whenever there is a sequence of bounded operators $\{A_\alpha : \alpha \in \mathbb{N}^m\} \subseteq \mathcal{L}(\mathcal{Y})$, such that, for every $z \in \mathbb{C}^m$ and any $c \in \mathcal{Y}$ the function*

$$\sum_{\alpha \in \mathbb{N}^m} (c, A_\alpha c) z^\alpha$$

is an entire function on $\mathbb{C}^m$ and for every $c \in \mathcal{Y}$ and $z \in \mathbb{C}^m$ it equals $(c, h(z)c)$. In this case we write

$$h(z) = \sum_{\alpha \in \mathbb{N}^m} A_\alpha z^\alpha, \ z \in \mathbb{C}^m. \tag{5.1}$$

Let $B_1, \ldots, B_m$ be any $n \times n$ complex matrices and h an entire function with values in $\mathcal{L}(\mathcal{Y})$. We let $h((B_1), \ldots, (B_m))$ be the $n \times n$ matrix whose $j, \ell-$element is the operator $h((B_1)_{j\ell}, \ldots, (B_m)_{j\ell}) \in \mathcal{L}(\mathcal{Y})$

Proposition 5.1 *If $\{K_j : j \in \mathbb{N}_m\}$ is a set of kernels on $\mathcal{X} \times \mathcal{X}$ with values in $\mathbb{R}$, and $h : \mathbb{R}^m \to \mathcal{L}(\mathcal{Y})$ an entire function of the type (5.1) where $\{A_\alpha : \alpha \in \mathbb{N}^m\} \subseteq \mathcal{L}_+(\mathcal{Y})$ then*

$$K = h(K_1, \ldots, K_m)$$

is a kernel on $\mathcal{X} \times \mathcal{X}$ with values in $\mathcal{L}(\mathcal{Y})$.

Proof. For any $\{c_j : j \in \mathbb{N}_m\} \subseteq \mathcal{Y}$, and $\{x_j : j \in \mathbb{N}_m\} \subseteq \mathcal{X}$ we must show that

$$\sum_{j, \ell \in \mathbb{N}_m} (c_j, h(K(x_j, x_\ell), \ldots, K_m(x_j, x_\ell)) c_\ell) \geq 0.$$

To this end, we write the sum in the form

$$\sum_{\alpha \in \mathbb{N}^m} \sum_{j \in \mathbb{N}_m} K_1^{\alpha_1}(x_j, x_\ell), \ldots, K_m^{\alpha_m}(x_j, x_\ell)(c_j, A_\alpha c_\ell).$$

Since $A_\alpha \in \mathcal{L}_+(\mathcal{Y})$, it follows that the matrix $((c_j, A_\alpha c_\ell)), j, \ell \in \mathbb{N}_m$ is positive semidefinite. Therefore, by the lemma of Schur, see e.g. [3, p. 358], we see that the matrix whose $j-$th, $\ell-$th element appears in the sum above is positive semidefinite for each $\alpha \in \mathbb{N}^m$. From this fact the result follows.

□

In our next observation we show that the function h with the property described in Proposition 5.1 must have the form (5.1) with $\{A_\alpha : \alpha \in \mathbb{N}^m\} \subseteq \mathcal{L}_+(\mathcal{Y})$ provided it and the kernels $K_1, \ldots, K_m$ satisfy some additional conditions. This issue is resolved in [17] in the scalar case. To apply those results we find the following definition useful.

Definition 5.2 *We say that the set of kernels $K_j : \mathcal{X}\times\mathcal{X} \to \mathbb{C}$, $j \in \mathbb{N}_m$, is* full *if for every n and any $n\times n$ positive semidefinite matrices $B_\ell, \ell \in \mathbb{N}_m$, there exist points $\{x_j : j \in \mathbb{N}_n\} \subseteq \mathcal{X}$ such that, for every $\ell \in \mathbb{N}_m$, $B_\ell = (K_\ell(x_j, x_k))_{j,k\in\mathbb{N}_m}$.*

For example when $m = 1$ and $\mathcal{X}$ is a *infinite* dimensional Hilbert space with inner product $(\cdot,\cdot)$, the kernel $(\cdot,\cdot)$ is full.

Proposition 5.2 *If $\{K_j : j \in \mathbb{N}_m\}$ is a set of full kernels on $\mathcal{X} \times \mathcal{X}$, $h : \mathbb{C}^m \to \mathcal{L}(\mathcal{Y})$, there exists some $r > 0$ such that the constant*

$$\gamma = \sup\{\|h(z)\| : \|z\| \le r\}$$

is finite, and $h(K_1, \dots, K_m)$ is a kernel with values in $\mathcal{L}(\mathcal{Y})$, then h is entire of the form

$$h(z) = \sum_{\alpha\in\mathbb{N}^m} A_\alpha z^\alpha,\ z \in \mathbb{C}^m$$

for some $\{A_\alpha : \alpha \in \mathbb{N}^m\} \subseteq \mathcal{L}_+(\mathcal{Y})$.

Proof. For any $c \in \mathcal{Y}$ and any positive semidefinite $n \times n$ matrices $\{B_j : j \in \mathbb{N} : m\}$, the $n \times n$ matrix

$$(c, h(B_1, \dots, B_m)c)$$

is positive semidefinite. Therefore, the function $g : \mathbb{C}^m \to \mathbb{C}$ defined by equation

$$g(z) := (c, h(z)c),\ z \in \mathbb{C}^m$$

has the property that $g(B_1, \dots, B_m)$ is an $n \times n$ positive semidefinite matrix. We now can apply the result in [17] and conclude that g is an entire function on $\mathbb{C}^m$ with nonnegative coefficients. Therefore, we have for all $z \in \mathbb{C}^m$ that

$$g(z) = \sum_{\alpha\in\mathbb{N}^m} g_\alpha z^\alpha$$

where, for all $\alpha \in \mathbb{N}^m$, $g_\alpha \ge 0$. By the polarization identity in $\mathcal{Y}$ we conclude that for any $c, d \in \mathcal{Y}$ the function f defined, for $z \in \mathbb{C}^m$ as

$$f(z) := (c, h(z)d)$$

is also entire on $\mathbb{C}^m$, that is, for all $z \in \mathbb{C}^m$ we have that

$$f(z) = \sum_{\alpha\in\mathbb{N}^m} f_\alpha z^\alpha$$

for some coefficients $f_\alpha, \alpha \in \mathbb{N}^m$. We will use these coefficients to construct the operators $A_\alpha, \alpha \in \mathbb{N}^m$. To this end, we express them by the Cauchy integral formula

$$f_\alpha = \alpha! \oint_T \frac{(c, h(z)d)}{\xi^{\alpha+1}} d\xi$$

where $T := \{z = (z_j : j \in \mathbb{N}_m) : |z_j| = r, j \in \mathbb{N}_m\}$, the $m-$torus. Hence, we obtain the estimate

$$|f_\alpha| \leq \alpha! \frac{M\|c\|\|d\|(2\pi)^m}{r^{|\alpha|}}.$$

From this fact and the observation that each f_α is bilinear in c and d we conclude that there is an $A_\alpha \in \mathcal{L}(\mathcal{Y})$ such that $f_\alpha = (c, A_\alpha d)$, see [1]. But $g_\alpha = (c, A_\alpha c)$, and so we obtain that $A_\alpha \in \mathcal{L}_+(\mathcal{Y})$. This complete the proof.

□

Let us now comment on translation invariant kernels on $\mathbb{R}^n$ with values in $\mathcal{L}(\mathcal{Y})$. This case is covered in great generality in [7, 16] where the notion of operator−valued Borel measures is described and used to generalize the theorem of Bochner, see e.g. [16, p. 12]. The interested reader can find complete details in these references. For the purpose of applications we point out the following sufficient condition on a function $h : \mathbb{R}^n \to \mathcal{L}(\mathcal{Y})$ to give rise to a translation invariant kernel. Specifically, for each $x \in \mathbb{R}^n$, we let $W(x) \in \mathcal{L}_+(\mathcal{Y})$, and define for $x \in \mathbb{R}^n$ the function

$$h(x) = \int_{\mathbb{R}^n} e^{ix\cdot t} W(t) dt.$$

Consequently, for every $m \in \mathbb{N}, \{x_j : j \in \mathbb{N}_m\} \subseteq \mathbb{R}^n$, and $\{c_j : j \in \mathbb{N}_m\} \subseteq \mathcal{Y}$ we have that

$$\sum_{j,\ell \in \mathbb{N}_m} (c_j, h(x_j - x_\ell)c_\ell) = \int_{\mathbb{R}^n} (c_j e^{ix_j\cdot t}, W(t)(c_\ell e^{ix_\ell \cdot t}))$$

which implies that $K(x, y) := h(x - t)$ is a kernel on $\mathbb{R}^n \times \mathbb{R}^n$. In [7, 16] all translation invariant kernels are characterized in form

$$h(x) = \int_{\mathbb{R}^n} e^{ix\cdot t} d\mu(t)$$

where $d\mu$ is a operator−valued Borel measure relative to $\mathcal{L}_+(\mathbb{R}^n)$. In particular this says, for example, that a function

$$h(x - t) = \sum_{j \in \mathbb{N}_m} e^{i(x-t)\cdot t_j} B_j$$

where $\{B_j : j \in \mathbb{N}_m\} \subseteq \mathcal{L}_+(\mathbb{R}^n)$ gives a translation invariant kernel too.

We also refer to [7, 16] to assert the claim that the function $g : \mathbb{R}_+ \to \mathbb{R}_+$ such that $K(x, t) := g(\|x - t\|^2)$ is a *radial* kernel on $\mathcal{X} \times \mathcal{X}$ into $\mathcal{L}(\mathcal{Y})$ for *any* Hilbert space $\mathcal{X}$ has the form

$$g(s) = \int_0^\infty e^{-\sigma s} d\mu(\sigma)$$

where again $d\mu$ is a operator−valued measure with values in $\mathcal{L}_+(\mathcal{Y})$. When the measure is discrete and $\mathcal{Y} = \mathbb{R}^n$ we conclude that

$$K(x, t) = \sum_{j \in \mathbb{N}_m} e^{-\sigma_j \|x-t\|^2} B_j$$

is a kernel for any $\{B_j : j \in \mathbb{N}_m\} \subseteq \mathcal{L}_+(\mathbb{R}^n)$ and $\{\sigma_j : j \in \mathbb{N}_m\} \subset \mathbb{R}_+$. This is the form of the kernel we use in our numerical simulations.

6 Practical considerations

In this final section we describe several issues of a practical nature. They include a description of some numerical simulations we have performed, a list of several problems for which we feel learning vector−valued functions is valuable and a brief review of previous literature on the subject.

6.1 Numerical simulations

In order to investigate the practical advantages offered by learning within the proposed function spaces we have carried out a series of three preliminary experiments where we tried to learn a target function f_0 by minimizing the regularization functional in equation (4.1) under different conditions. In all experiments set f_0 to be of the form

$$f_0(x) = K(x, x_1)c_1 + K(x, x_2)c_2, \quad c_1, c_2 \in \mathbb{R}^n, x \in [-1, 1]. \tag{6.1}$$

I the first experiment, the kernel K was given by a mixture of two gaussian functions, namely

$$K(x, t) = Se^{-6\|x-t\|^2} + Te^{-30\|x-t\|^2}, \quad x, t \in [-1, 1] \tag{6.2}$$

where $S, T \in \mathcal{L}_+(\mathbb{R}^n)$. Here we compared the solution obtained by minimizing equation (4.1) either in the RKHS of kernel K above (we call this *method 1*) or in the RKHS of the diagonal kernel

$$I\left(\frac{tr(S)}{n}e^{-6\|x-t\|^2} + \frac{tr(T)}{n}e^{-30\|x-t\|^2}\right), \quad x, t \in [-1, 1] \tag{6.3}$$

where tr denotes the trace of a squared matrix (we call this *method 2*). We performed different

Table 1: Experiment 1: average difference between the test error of method 1 and the test error of method 2 with its standard deviation (bottom row in each cell) for different values of the number of outputs n and the noise level a. The average error of method 1 (not reported here) ranged between 2.80e−3 and 1.423.

$a \setminus n$	2	3	5	10	20
0.1	2.01e−4	9.28e−4	.0101	.0178	.0331
	(1.34e−4)	(5.07e−4)	(.0203)	(.0101)	(.0238)
0.5	1.83−3	.0276	.0197	.0279	.5508
	(1.17e−3)	(.0123)	(.126)	(.0131)	(.3551)
1	.0167	.0238	.0620	.3397	.5908
	(.0104)	(.0097)	(.0434)	.3608	(0.3801)

series of simulations, each identified by the number of the outputs $n \in \{2, 3, 5, 10, 20\}$ and an output noise parameter $a \in \{0.1, 0.5, 1\}$. Specifically, for each pair (n, a), we computed the matrices $S = A^*A$ and $T = B^*B$ where the elements of matrices A and B were uniformly sampled in the interval $[0, 1]$. The centers $x_1, x_2 \in [-1, 1]$ and the vector parameters $c_1, c_2 \in$

$[-1,1]^n$ defining the target function f_0 were also uniformly distributed. The training set $\{(x_j, y_j) : j \in \mathbb{N}_m\} \subset [-1,1] \times \mathbb{R}^n$ was generated by sampling the function f_0 with noise, namely x_j was uniformly distributed in $[-1,1]$ and $y_j = f_0(x_j) + \epsilon_j$ with ϵ_j also uniformly sampled in the interval $[-a,a]$. In all simulations we used a set of $m = 20$ points for training and a separate validation set of 100 data to select the optimal value of the regularization parameter. Then we computed on a separate test set of 200 samples the mean squared error between the target function f_0 and the function learned from the training set. This simulation was repeated 100 times, that is for each pair (n, a), 100 test error measures were produced for method 1 and method 2. As a final comparison measure of method 1 and method 2 we computed the average difference between the test error of method 2 and the test error of method 1 and its standard deviation, so, for example, a positive value of this average means that method 1 performs better than method 2. These results are shown in table 1 were each cell contains the average difference error and, below it, its standard deviation. Not surprisingly, the results indicate that the use of the non-diagonal kernel *always* on the average improves performance, especially when the number of outputs increases.

Table 2: Experiment 2: average difference between the test error of method 3 and the test error of method 4 with its standard deviation for different values of the number of outputs n and the noise level a. The average error of method 3 (not reported here) ranged between 3.01e−3 and 6.95e−2.

$a \setminus n$	2	3	5	10	20
0.1	8.01e−4	7.72e−4	-6.85e−4	6.72e−4	7.42e−4
	(6.76e−4)	(8.05e−4)	(5.00e−4)	5.76e−4	3.99e−3
0.5	1.57e−3	.0133	7.30−3	-7.30e−3	8.64e−3
	(1.89e−3)	(.0152)	(4.51e−3)	(4.51e−3)	(4.63e−3)
1	.0113	.0149	9.17e−3	.0242	.0233
	(.0086)	(.0245)	(5.06e−3)	(.0103)	(.0210)

In the second experiment, the target function in equation (6.1) was modelled by the diagonal kernel

$$K(x,t) = I\left(ae^{-6\|x-t\|^2} + be^{-30\|x-t\|^2}\right), \quad x,t \in [-1,1]$$

where $a, b \in [0,1]$, $a + b = 1$. In this case we compared the optimal solution produced by minimizing equation (4.1) either in the RKHS of the above kernel (we call this *method 3*) or in the RKHS of the kernel

$$\frac{an}{tr(S)} Se^{-6\|x-t\|^2} + \frac{bn}{tr(T)} Te^{-30\|x-t\|^2}, \quad x,t \in [-1,1]$$

(we call this *method 4*). The results of this experiment are shown in table 2 where it is evident that the non diagonal kernel *very often* does worse than the diagonal kernel. Comparing these results with those in table 1 we conclude that the non diagonal kernel does not offer any advantage when $n = 2, 3$. On the other hand, this type of kernel helps dramatically when the number for outputs is larger or equal to 5 (it does little damage in experiment 2 and great

benefit in experiment 1). There is no current explanation for this phenomenon. Thus, it appears to us that such kernel and its generalization as we discussed above may be important for applications where the number of outputs is large and complex relations among them are likely to occur.

The last experiment addressed hyper-parameters tuning. Specifically, we again choose a target function as in equation (6.1) and K as in equation (6.2) but, now, $S = S^0$ where $S^0_{\ell q} = 1$ if $\ell = q$, -1 if $\ell = q+1$ (with periodic conditions, i.e. $n+1 = 1$), and zero otherwise, and $T = T^0$ is a sparse out-of-diagonal p.d. binary matrix. Thus, S^0 models correlations between adjacent components of f (it acts as a difference operator) whereas T_0 accounts for possible "far away" component correlations. To learn this target function using three different parameterized kernel models. *Model* 1 consists of kernels as in equation (6.2) with $S = S^0 + \lambda I$ and $T = \rho T_0$, $\lambda, \rho \geq 0$. *Model* 2 is as model 1 but with $\rho = 0$. Finally, *model* 3 consists of diagonal kernels as in equation (6.3) with $S = S_0$ and $T = \nu T_0$, $\nu > 0$. The data generation setting was like in the above experiments except that the matrices S_0 and T_0 were kept fixed in each simulation (this explain the small variance in the plots below) and the output noise parameter was fixed to 0.5. Hyper-parameters of each model as well as the regularization parameter in equation (4.1) were found by minimizing the squared error computed on a separate validation set containing 100 independently generated points. Figure 1 depicts the mean square test error with its standard deviation as a function of the number of training points (5,10,20,30) for $n = 10, 20, 40$ output components. The result clearly indicated the advantage offered by model 1.

Figure 1: *Experiment 3 ($n = 10, 20, 40$) Mean square test error as a function of the training set size for model* 1 *(solid line), model* 2 *(dashed line) and model* 3 *(doted line). See text for a description on these models.*

Although the above experiments are preliminary and consider a simple data-setting, they enlighten the value offered by matrix-valued kernels.

6.2 Application scenarios

We outline some practical problems arising in the context of learning vector−valued functions where the above theory can be of value. Although we do not treat the issues that would arise in a detailed study of these problems, we hope that our discussion will motive substantial studies of them.

A first class of problems deals with finite dimensional vector spaces. For example, an interesting problem arises in reinforcement learning or control when we need to learn a map

between a set of sensors placed in a robot or autonomous vehicle and a set of actions taken by the robot in response to the surrounding environment, see e.g. [18]. Another instance of this class of problems deals with the space of $n \times m$ matrices over a field whose choice depends on the specific application. Such spaces can be equipped with the standard Frobenius inner product. One of many problems which come to mind is to compute a map which transforms an $\ell \times k$ image x into a $n \times m$ image y. Here both images are normalized to the unit interval, so that $\mathcal{X} = [0,1]^{\ell \times k}$ and $\mathcal{Y} = [0,1]^{n \times m}$. There are many specific instances of this problem. In *image reconstruction* the input x is an incomplete (occluded) image obtained by setting to 0 the pixel values of a fixed subset in an underling image y which forms our target output. A generalization of this problem is *image de-noising* where x is an $n \times m$ image corrupted by a fixed but unknown noise process and the output y is the underling "clean" image. Yet a different application is *image morphing* which consists in computing the pointwise correspondence between a pair of images depicting two similar objects, e.g. the faces of two people. The first image, x_0, is meant to be fixed (a reference image) whereas the second one, x, is sampled from a set of possible images in $\mathcal{X} = [0,1]^{n \times m}$. The output $y \in \mathbb{R}^{2(n \times m)}$ is a vector field in the input image plane which associates each pixel of the input image to the vector of coordinates of the corresponding pixel in the reference image, see e.g. [8] for more information on these issues.

A second class of problems deals with spaces of strings, such as text, speech, or biological sequences. In this case $\mathcal{Y}$ is a space of finite sequences whose elements are in a (usually finite) set $\mathcal{A}$. A famous problem asks to compute a *text-to-speech map* which transforms a word from a certain fixed language into its sound as encoded by a string of phonemes and stresses. This problem was studied by Sejnowski and Rosenberg in their NETtalk system [25]. Their approach consists in learning several Boolean functions which are subsequently concatenated to obtained the desired text-to-speech map. Another approach is to learn directly a vector−valued function. In this case $\mathcal{Y}$ could be described by means of an appropriate RKHS, see below. Another important problem is *protein structure prediction* which consists in predicting the 3D structure of a protein immerse in a aqueous solution, see, e.g., [4]. Here $x = (x^j \in \mathcal{A} : j \in \mathbb{N}_\ell)$ is the primary sequence of a protein, i.e. a sequence of twenty possible symbols representing the amino acids present in nature. The length ℓ of the sequence varies typically between a few tens and a few thousand elements. The output $y = (y_j \in \mathbb{R}^3 : j \in \mathbb{N}_\ell)$ is a vector with the same length as x and y_j is the position of the $j-$th amino acid of the input sequence. Towards the solution of this difficult problem an intermediate step consists in predicting a neighborhood relation among the amino acids of a protein. In particular, recent work has focused on predicting a squared binary matrix C, called the contact map matrix, where $C(j,k) = 1$ if the amino acids j and k are at a distance smaller or of the order of $10A$, and zero otherwise, see [4] and references therein.

Note that in the last class of problems the output space is *not* a Hilbert space. An approach to solve this difficulty is to embed $\mathcal{Y}$ into a RKHS. This requires *choosing* a scalar−valued kernel $G : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ which provides a RKHS space $\mathcal{H}_G$. We associate every element $y \in \mathcal{Y}$ with the function $G(y, \cdot) \in \mathcal{H}_G$ and denote this map by $G : \mathcal{Y} \to \mathcal{H}_G$. If we wish to learn a function $f : \mathcal{X} \to \mathcal{Y}$, we instead learn the composition mapping $\tilde{f} := G \circ f$. We then can obtain $f(x)$ for every $x \in \mathcal{X}$ as

$$f(x) = \text{argmin}_{y \in \mathcal{Y}} \|G(y, \cdot) - \tilde{f}(x)\|_G.$$

If G is chosen to be bijective the minimum is unique. This idea is also described in [29] where some applications of it are presented.

As third and a final class of problems which illustrate the potential of vector−valued learning we point to the possibility of learning a curve, or even a manifold. In the first case, the usual paradigm for the computer generation of a curve in $\mathbb{R}^n$ is to start with scalar−valued basis functions $\{M_j : j \in \mathbb{N}_m\}$, and a set of *control points*, $\{c_j : j \in \mathbb{N}_m\} \subseteq \mathbb{R}^n$, and consider the vector−valued function

$$f = \sum_{j \in \mathbb{N}_m} c_j M_j$$

see e.g. [22]. We are given data $\{y_j : j \in \mathbb{N}_m\} \subseteq \mathbb{R}^n$ and we want to learn the control points. However the input data, which belong to a unit interval, say $\mathcal{X} = [0, 1]$, is unknown. We then fix $\{x_j : j \in \mathbb{N}_m\} \subset [0, 1]$ and we learn a parameterization $\tau : [0, 1] \to [0, 1]$ such that

$$f(\tau(x_i)) = y_i.$$

So the problem is to learn both τ and the control points. Similarly, if $\{M_j : j \in \mathbb{N}_m\}$ are functions in $\mathbb{R}^k$, $k < n$ we face the problem of learning a manifold $\mathcal{M}$ obtained by embedding $\mathbb{R}^k$ in $\mathbb{R}^n$. It seems that to learn the manifold $\mathcal{M}$ a kernel of the form

$$K(x, t) = \sum_{j \in \mathbb{N}_m} M_j(x) M_j(t) A_j$$

where $\{A_j : j \in \mathbb{N}_m\} \subset \mathcal{L}_+(\mathbb{R}^n)$ would be appropriate, because the functions generated by such kernel lie in the manifold generated by the basis functions, namely $K(x, t)c \in \mathcal{M}$ for every $x, t \in \mathbb{R}^k, c \in \mathbb{R}^n$.

6.3 Previous works on vector−valued learning

The problem of learning vector−valued functions has been addressed in the statistical literature under the name of *multiple response estimation* or *multiple output regression*, see [19] and references therein. Here we briefly discuss two methods that we find interesting.

A well-studied technique is the Wold *partial least squares* approach. This method is similar to principal component analysis but with the important difference that the principal directions are computed simultaneously in the input and output spaces, both being finite dimensional Euclidean spaces, see e.g. [19]. Once the principal directions have been computed, the data are projected along the first n directions and a least square fit is computed in the projected space where the optimal value of the parameter n can be estimated by means of cross validation. We note that like principle component analysis, partial least squares are *linear models*, that is they can model only vector−valued functions which depend linearly on the input variables. Recent work by Trejo and Rosipal [23] and Bennett and Embrechts [5] has reconsidered PLS in the context of *scalar* RKHS.

Another statistically motivated method is Breiman and Friedman *curd & whey* procedure [9]. This method consists of two main steps where, first, the coordinates of a vector−valued function are separately estimated by means of a least squares fit or by ridge regression and, second, they are combined in a way which exploit possible correlations among the responses

(output coordinates). The authors show experiments where their method is capable of reducing prediction error when the outputs are correlated and not increasing the error when the outputs are uncorrelated. The curd & whey method is also primarily restricted to model linear relation of possibly non linear functions. However it should be possible to "kernelized" this method following the same lines as in [5].

Acknowledgements: We are grateful to Dennis Creamer, Head of the Computational Science Department at National University of Singapore for providing both of us with the opportunity to complete this work in a scientifically stimulating and friendly environment. Kristin Bennett of the Department of Mathematical Sciences at RPI and Wai Shing Tang of the Mathematics Department at NUS provided us with several helpful references and remarks. Bernard Buxton of the Department of Computer Science at UCL read a preliminary version of the manuscript and made useful suggestions. Finally, we are grateful to Phil Long of the Genome Institute of NUS for discussions which led to Theorem 4.3.

References

[1] N.I. Akhiezer and I.M. Glazman. *Theory of linear operators in Hilbert spaces*, volume I. Dover reprint, 1993.

[2] L. Amodei. Reproducing kernels of vector–valued function spaces. Proc. of Chamonix, A. Le Meehaute et al. Eds., pp. 1–9, 1997.

[3] N. Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.*, 686:337–404, 1950.

[4] P. Baldi, G. Pollastri, P. Frasconi, and A. Vullo. New machine learning methods for the prediction of protein topologies. Artificial Intelligence and Heuristic Methods for Bioinformatic, P. Frasconi and R. Shamir Eds., IOS Press, 2002.

[5] K.P. Bennett and M.J. Embrechts. An optimization perspective on kernel partial least squares regression, *Learning Theory and Practice - NATO ASI Series*, J. Suykens et al. Eds., IOS Press, 2003.

[6] K.P. Bennett and O.L. Mangasarian. Multicategory discrimination via linear programming. *Optimization Methods and Software*, 3:722–734, 1993.

[7] S.K. Berberian. *Notes on spectral theory.* Van Nostrand Company, New York, 1966.

[8] D. Beymer and T. Poggio. Image representations for visual learning. *Science*, 272(5270):1905–1909, June 1996.

[9] L. Breiman and J. Friedman. Predicting multivariate responses in multiple linear regression (with discussion). *J. Roy. Statist. Soc. B.*, 59:3–37, 1997.

[10] J. Burbea and P. Masani. *Banach and Hilbert spaces of vector-valued functions.* Pitman Rsearch Notes in Mathematics Series 90, 1984.

[11] V. Cherkassky and F. Mulier. *Learning from Data: Concepts, Theory, and Methods*. Wiley, New York, 1998.

[12] K. Cramer and Y. Singer. On the algorithmic implementation of multi-class kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292, 2001.

[13] N.A.C. Cressie. *Statistics for Spatial Data*. Wiley, New York, 1993.

[14] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines*. Cambridge University Press, 2000.

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

[16] P.A. Fillmore. *Notes on operator theory*. Van Nostrand Company, New York, 1970.

[17] C.H. FitzGerald, C. A. Micchelli, and A. M. Pinkus. Functions that preserves families of positive definite functions. *Linear Algebra and its Appl.*, 221:83–102, 1995.

[18] U. Franke, D. Gavrila, S. Goerzig, F. Lindner, F. Paetzold, and C. Woehler. Autonomous driving goes downtown. *IEEE Intelligent Systems*, pages 32–40, November/December 1998.

[19] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning: Data Mining, Inference, and Prediciton*. Springer Series in Statistics, 2002.

[20] O.L. Mangasarian. *Nonlinear Programming*. Classics in Applied Mathematics. SIAM, 1994.

[21] A.A. Melkman and C.A. Micchelli. Optimal estimation of linear operators in hilbert spaces from inaccurate data. *SIAM J. of Numerical Analysis*, 16(1):87–105, 1979.

[22] C.A. Micchelli. *Mathematical Aspects of Geometric Modeling*. CBMS-NSF Regional Conference Series in Applied MAthematics. SIAM, Philadelphia, PA, 1995.

[23] R. Rosipal and L. J. Trejo. Kernel partial least squares regression in reproducing kernel hilbert spaces. *J. of Machine Learning Research*, 2:97–123, 2001.

[24] B. Schölkopf and A.J. Smola. *Learning with Kernels*. The MIT Press, Cambridge, MA, USA, 2002.

[25] T.J. Sejnowski and C.R. Rosenberg. Parallel networks which learn to pronounce english text. *Complex Systems*, 1:145–163, 1987.

[26] A. N. Tikhonov and V. Y. Arsenin. *Solutions of Ill-posed Problems*. W. H. Winston, Washington, D.C., 1977.

[27] V. N. Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.

[28] G. Wahba. *Splines Models for Observational Data*. Series in Applied Mathematics, Vol. 59, SIAM, Philadelphia, 1990.

[29] J. Weston, O. Chapelle, A. Elisseeff, B. Schölkopf, and Vapnik. Kernel dependency estimation. *Advances in Neural Information Processing Systems*, S. Becker et al. Eds., 15, MIT Press, Cambridge, MA, USA, 2003.

[30] J. Weston and C. Watkins. Multi-class support vector machines. Technical Report CSD-TR-98-04, Royal Holloway, University of London, Department of Computer Science, 1998.