A Generalized Kernel Approach to Structured Output Learning

Hachem Kadri, Mohammad Ghavamzadeh, Philippe Preux

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A Generalized Kernel Approach to Structured Output Learning

Hachem Kadri — Mohammad Ghavamzadeh — Philippe Preux

N° 7956

February 2012

Thème COG

ISSN 0249-6399 ISRN INRIA/RR--7956--FR+ENG

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
LILLE - NORD EUROPE

A Generalized Kernel Approach to Structured Output Learning

Hachem Kadri*, Mohammad Ghavamzadeh†, Philippe Preux‡

Thème COG — Systèmes cognitifs
Équipes-Projets SequeL

Rapport de recherche n° 7956 — February 2012 — 19 pages

Abstract: We study the problem of structured output learning from a regression perspective. We first provide a general formulation of the kernel dependency estimation (KDE) problem using operator-valued kernels. We show that some of the existing formulations of this problem are special cases of our framework. We then propose a covariance-based operator-valued kernel that allows us to take into account the structure of the kernel feature space. This kernel operates on the output space and encodes the interactions between the outputs without any reference to the input space. To address this issue, we introduce a variant of our KDE method based on the conditional covariance operator that in addition to the correlation between the outputs takes into account the effects of the input variables. Finally, we evaluate the performance of our KDE approach using both covariance and conditional covariance kernels on two structured output problems, and compare it to the state-of-the-art kernel-based structured output regression methods.

Key-words: structured outputs, operator-valued kernel, function-valued RKHS, covariance operator

* Sequel Team, INRIA Lille. E-mail: hachem.kadri@inria.fr
† Sequel Team, INRIA Lille. E-mail: mohammad.ghavamzadeh@inria.fr
‡ Sequel/INRIA-Lille, LIFL/CNRS. E-mail: philippe.preux@inria.fr

Centre de recherche INRIA Lille – Nord Europe
Parc Scientifique de la Haute Borne
40, avenue Halley, 59650 Villeneuve d'Ascq
Téléphone : +33 3 59 57 78 00 — Télécopie : +33 3 59 57 78 50

Une approche à noyau généralisé pour la prédiction de sorties structurées

Résumé : Nous étudions le problème de la prédiction de sorties structurées par des méthodes de régression. Nous commençons par donner une formulation générale de la méthode KDE (Kernel Dependency Estimation) utilisant les noyaux à valeurs opérateurs. Nous proposons ensuite un noyau à valeurs opérateurs basé sur l'opérateur de covariance qui a l'avantage de nous permettre de prendre en compte la structure de l'espace caractéristique du noyau de sortie. Nous introduisons aussi une variante de notre méthode KDE basé sur l'opérateur de covariance conditionnelle qui, en plus de la corrélation entre les sorties, va prendre en compte les effets des variables d'entrée. Enfin, nous évaluons la performance de notre approche utilisant les noyaux à valuers opérateurs basés sur l'opérateur de covariance et de covariance conditionnelle sur deux problèmes de prédiction de sorties structurées.

Mots-clés : prédiction de sorties structurées, noyaux à valeurs opérateurs, espace de Hilbert à noyau reproduisant de fonctions à valeurs fonctions, opérateur de covariance

1 Introduction

In many situations one is faced with the task of learning a mapping between objects of different nature, which can be generally characterized by complex data structures. This is often the case in a number of practical applications such as statistical machine translation (Wang and Shawe-Taylor, 2010), and speech recognition or synthesis (Cortes et al., 2005). Designing algorithms that are sensitive enough to detect structural dependencies among these complex data is becoming increasingly important. While classical learning algorithms can be easily extended to include complex inputs, more refined and sophisticated algorithms are needed to handle complex outputs. In this case, several mathematical and methodological difficulties arise and these difficulties increase with the complexity of the output space. Depending on the nature of this space, complex output data can be divided into three classes: (1) Euclidean, i.e vectors or real-valued functions; (2) mildly non-Euclidean, i.e points on a manifold and shapes; and (3) strongly non-Euclidean, i.e structured data like trees or graphs. To date, most powerful methodologies have been developed for output data that resides in a Euclidean space. Multi-task learning (vector outputs) and functional data analysis (functional outputs) have received a great deal of scientific attention in the machine learning and statistics communities (Caruana, 1997; Ando and Zhang, 2005; Ramsay and Silverman, 2005). However a relatively little attention has been paid to the structured output case. Recently, great efforts have been made to expand the applicability and robustness of general learning engines to structured output contexts, and international workshops have been dedicated to this research topic. For example, we can cite the ICML workshop on "Learning in Structured Output Spaces" (Brefeld et al., 2006), the NIPS workshop "Structured Input - Structured Output" (Borgwardt et al., 2008), and the SAMSI[1] research program on "Analysis of Object Data" (Müller et al., 2011).

One difficulty encountered when working with structured data is that usual Euclidean methodologies cannot be applied in this case. Reproducing kernels provide an elegant way to overcome this problem. Defining a suitable kernel on the structured data allows to: (1) encapsulate the structural information in a kernel function, (2) transform the problem to a Euclidean space. For structured outputs, two different but closely related kernel-based approaches can be found in the literature: kernel dependency estimation (KDE) and joint kernel maps (Bakir et al., 2007). KDE was first proposed by Weston et al. Weston et al. (2003), and then reformulated by Cortes et al. Cortes et al. (2005). The idea is to define a kernel on the output space $\mathcal{Y}$ to project the structured outputs in a real-valued reproducing kernel Hilbert space (RKHS) $\mathcal{F}_\mathcal{Y}$, and then one can perform a scalar-valued kernel ridge regression (KRR) between the input space $\mathcal{X}$ and the feature space $\mathcal{F}_\mathcal{Y}$. Having the regression coefficients, the structured output prediction is obtained by computing the pre-image from $\mathcal{F}_\mathcal{Y}$. Note that in the original formulation of KDE (Weston et al., 2003), an additional step is added at the beginning (before regression) to reduce the dimension of $\mathcal{F}_\mathcal{Y}$ and decorrelates the outputs via kernel principal components analysis (kPCA). The second kernel-based approach for structured output prediction is based on joint kernels which are nonlinear similarity measures between pairs of input-output (Tsochantaridis et al., 2005; Weston et al., 2007). While in KDE sepa-

[1] SAMSI stands for Statistical and Applied Mathematical Sciences Institute.

rate kernels for inputs and outputs are used to project input and output data in two feature spaces (which can be different), the joint kernel in the second approach permits to map these data into a single feature space and then allows to take into account prior knowledge on input-output and output correlations. However, this improvement require an exhaustive computation of pre-images during training; a problem encountered only in the testing stage with KDE.

In this paper we will focus our attention on KDE-type methods for structured output learning. KDE is a regression based approach and we agree with Cortes et al. (2005) that minimizing a similarity measure capturing the closeness of the outputs in the output feature space (like in regression) is a natural way of learning the relevant structure. Moreover, avoiding the computation of pre-images during training is an important advantage over joint-kernel based methods. The main drawback of previous formulations of the KDE approach is that prior known input-output correlations cannot be easily incorporated into the method. We overcome this problem by generalizing these formulations and allowing KDE to capture output and input-output dependencies using operator-valued (multi-task) kernels. Our idea is to improve the high-dimensional regression step in KDE by doing an operator-valued KRR rather than a scalar-valued one. By this way, operator-valued kernels allow to map input data and structured output features into a joint feature space in which existing correlations can be exploited.

Section 2 describes our KDE general formulation and shows how the formulation in Cortes et al. (2005) can be recovered from it by choosing a particular operator-valued kernel. Note that operator-valued kernels have been recently used for structured outputs in order to deal with the problem of link prediction (Brouard et al., 2011). However, the authors considered only the case of identity-based operator-valued kernels which is equivalent to the KDE formulation of Cortes et al. (2005), and did not discuss the associated pre-image problem since their link prediction application did not require this step. In Section 3, we make a step further by defining operator-valued kernels using covariance operators on a RKHS which allow to take into account the structure of the output kernel feature space, and providing a way for making structured prediction from them. Section 4 discusses some related works on joint-kernel based structured prediction and on covariance operators in RKHS. In Section 5, we evaluate the performance of our approach and compare it to previous KDE formulation on a number of structured output problems. Finally, Section 6 concludes the paper.

2 Operator-valued Kernel Formulation of Kernel Dependency Estimation

Given $(x_i, y_i)_{i=1}^n \in \mathcal{X} \times \mathcal{Y}$ where $\mathcal{X}$ is the input space and $\mathcal{Y}$ the structured output space, we consider the problem of learning the mapping f from $\mathcal{X}$ to $\mathcal{Y}$. The idea of KDE is to embed output data by using a mapping Φ_l between the structured output space $\mathcal{Y}$ and a Euclidean feature space $\mathcal{F}_\mathcal{Y}$ defined by a scalar-valued a kernel l. Instead of learning f in order to predict an output y for an input x, we first learn the mapping g from $\mathcal{X}$ to $\mathcal{F}_\mathcal{Y}$, and then compute the pre-image of $g(x)$ by the inverse mapping of Φ_l, i.e., $y = f(x) = \Phi_l^{-1}(g(x))$ (see Figure 1). Note that in the original KDE algorithm (Weston et al., 2003),

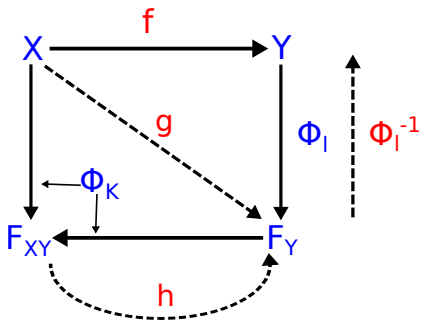

Figure 1: Kernel Dependency Estimation. Our generalized formulation consists in learning the mapping g using an operator-valued kernel ridge regression rather than a scalar-valued one as in the formulations of Weston et al. (2003); Cortes et al. (2005). Using an operator-valued kernel mapping, we construct a joint feature space from information of input and output spaces in which input-output and outputs correlation can be taken into account.

the feature space $\mathcal{F}_\mathcal{Y}$ is reduced using kernel PCA. However, all previous KDE formulations use ridge regression with a scalar-valued kernel k on $\mathcal{X}$ to learn the mapping g. This approach has the drawback of not taking into account dependencies between data in the feature space $\mathcal{F}_\mathcal{Y}$. Indeed, variables in $\mathcal{F}_\mathcal{Y}$ can be highly correlated since they are the projection of y_i using the mapping Φ_l, and considering this correlation is essential to retain and exploit the structure of the outputs. To overcome this problem, we propose to use an operator-valued (multi-task) kernel-based regression approach to learn g in a vector (or function)-valued RKHS built from a kernel which outputs an operator that encodes the relationship between the output's components. As we show in Figure 1, the feature space introduced by an operator-valued kernel is a joint feature space which contains information of the input space $\mathcal{X}$ and the output feature space $\mathcal{F}_\mathcal{Y}$. So, by using operator-valued kernels we allow KDE to exploit input-output and output correlations as in the joint kernel structured prediction approach (Tsochantaridis et al., 2005; Weston et al., 2007).

We now describe our KDE formulation that takes into account the general case where the feature spaces associated to input and output kernels can be infinite dimensional, contrary to Cortes et al. (2005) in which only the case of finite feature spaces is considered. The key idea of our formulation is to use operator-valued kernels to incorporate prior knowledge about the structure of the output feature space while taking into account the input data as well. We start by recalling some properties of these kernels and their associated RKHSs. For more details in the context of learning theory, see Micchelli and Pontil (2005). Table 1 summarizes the notations used in the paper. Let $\mathcal{L}(\mathcal{F}_\mathcal{Y})$ the set of bounded operators from $\mathcal{F}_\mathcal{Y}$ to $\mathcal{F}_\mathcal{Y}$.

Definition 1 *(Non-negative $\mathcal{L}(\mathcal{F}_\mathcal{Y})$-valued kernel)*
A non-negative $\mathcal{L}(\mathcal{F}_\mathcal{Y})$-valued kernel K is an operator-valued function on $\mathcal{X}\times\mathcal{X}$, i.e., $K : \mathcal{X}\times\mathcal{X} \to \mathcal{L}(\mathcal{F}_\mathcal{Y})$, such that for all $x_i, x_j \in \mathcal{X}$, $\varphi_i, \varphi_j \in \mathcal{F}_\mathcal{Y}$, and $m \in \mathbb{N}^*_+$

i. $K(x_i, x_j) = K(x_j, x_i)^*$ *($*$ denotes the adjoint),*

ii. $\sum_{i,j=1}^{m} \langle K(x_i, x_j)\varphi_i, \varphi_j\rangle_{\mathcal{F}_\mathcal{Y}} \geq 0.$

input space	$\mathcal{X}$
structured output space	$\mathcal{Y}$
input data	$x_i \in \mathcal{X}$
structured output data	$y_i \in \mathcal{Y}$
scalar-valued kernel	$k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$
scalar-valued kernel	$l : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$
output feature space	$\mathcal{F}_\mathcal{Y}$
output feature map	$\Phi_l : \mathcal{Y} \to \mathcal{F}_\mathcal{Y}$
set of operators from $\mathcal{F}_\mathcal{Y}$ to $\mathcal{F}_\mathcal{Y}$	$\mathcal{L}(\mathcal{F}_\mathcal{Y})$
operator-valued kernel	$K : \mathcal{X} \times \mathcal{X} \to \mathcal{L}(\mathcal{F}_\mathcal{Y})$
joint feature space	$\mathcal{F}_{\mathcal{X}\mathcal{Y}}$
joint feature map	$\Phi_K : \mathcal{X} \times \mathcal{F}_\mathcal{Y} \to \mathcal{F}_{\mathcal{X}\mathcal{Y}}$
a mapping from $\mathcal{X}$ to $\mathcal{Y}$	f
a mapping from $\mathcal{X}$ to $\mathcal{F}_\mathcal{Y}$	g

Table 1: This table summarizes the notations used in the paper.

Note that both properties above guarantee that the operator-valued kernel matrix $\mathbf{K} = \big[K(x_i, x_j) \in \mathcal{L}(\mathcal{F}_\mathcal{Y})\big]_{i,j=1}^n$ is positive. Given a non-negative $\mathcal{L}(\mathcal{F}_\mathcal{Y})$-valued kernel K on $\mathcal{X} \times \mathcal{X}$, there exists a unique RKHS of $\mathcal{F}_\mathcal{Y}$-valued functions defined over $\mathcal{X} \times \mathcal{X}$ whose reproducing kernel is K.

Definition 2 *($\mathcal{F}_\mathcal{Y}$-valued RKHS)*
A RKHS $\mathcal{F}_{\mathcal{X}\mathcal{Y}}$ of $\mathcal{F}_\mathcal{Y}$-valued functions $g : \mathcal{X} \to \mathcal{F}_\mathcal{Y}$ is a Hilbert space such that there is a non-negative $\mathcal{L}(\mathcal{F}_\mathcal{Y})$-valued kernel K with the following properties:

i. $\forall x \in \mathcal{X},\ \forall \varphi \in \mathcal{F}_\mathcal{Y}, \qquad K(x, \cdot)\varphi \in \mathcal{F}_{\mathcal{X}\mathcal{Y}},$

ii. $\forall g \in \mathcal{F}_{\mathcal{X}\mathcal{Y}},\ \forall x \in \mathcal{X},\ \forall \varphi \in \mathcal{F}_\mathcal{Y},$
$$\langle g, K(x, \cdot)\varphi \rangle_{\mathcal{F}_{\mathcal{X}\mathcal{Y}}} = \langle g(x), \varphi \rangle_{\mathcal{F}_\mathcal{Y}}.$$

Every RKHS $\mathcal{F}_{\mathcal{X}\mathcal{Y}}$ of $\mathcal{F}_\mathcal{Y}$-valued functions is associated with a unique non-negative $\mathcal{L}(\mathcal{F}_\mathcal{Y})$-valued kernel K, called the reproducing kernel.

Operator-valued KDE is performed in two steps:

Step 1 (kernel ridge) Regression: We first use an operator-valued kernel-based regression technique and learn the function g in the $\mathcal{F}_\mathcal{Y}$-valued RKHS $\mathcal{F}_{\mathcal{X}\mathcal{Y}}$ from the training data of the form $\big(x_i, \Phi_l(y_i)\big)_{i=1}^n \in \mathcal{X} \times \mathcal{F}_\mathcal{Y}$, where Φ_l is the mapping from the structured output space $\mathcal{Y}$ to the scalar-valued RKHS $\mathcal{F}_\mathcal{Y}$. Similar to other KDE formulations, we consider kernel ridge regression and solve the following optimization problem:

$$\underset{g \in \mathcal{F}_{\mathcal{X}\mathcal{Y}}}{\arg\min} \sum_{i=1}^n \|g(x_i) - \Phi_l(y_i)\|_{\mathcal{F}_\mathcal{Y}}^2 + \lambda \|g\|^2, \tag{1}$$

where $\lambda > 0$ is a regularization parameter. Using the representer theorem, the solution of (1) has the following form[2]:

$$g(\cdot) = \sum_{i=1}^n K(\cdot, x_i)\psi_i \tag{2}$$

[2]As in the scalar-valued case, operator-valued kernels provide an elegant way of dealing with nonlinear problems (mapping g) by reducing them to linear ones (mapping h) in some feature space ($\mathcal{F}_{\mathcal{X}\mathcal{Y}}$) (see Figure 1).

where $\psi_i \in \mathcal{F}_\mathcal{Y}$. Substituting (2) in (1), we obtain an analytic solution for the optimization problem as

$$\mathbf{\Psi} = (\mathbf{K} + \lambda I)^{-1}\mathbf{\Phi_l}, \tag{3}$$

where $\mathbf{\Phi_l}$ is the column vector of $\big[\Phi_l(y_i) \in \mathcal{F}_\mathcal{Y}\big]_{i=1}^n$. Eq. (3) is a generalization of the scalar-valued kernel ridge regression solution to vector or functional outputs (Caponnetto and De Vito, 2007), in which the kernel matrix is a block operator matrix. Note that features $\Phi_l(y_i)$ in this equation can be explicit or implicit. We show in the following by substituting (3) in the next step (step 2) that, even with implicit features, we are still able to formulate the structured output prediction problem in terms of explicit quantities that are readily computable via input and output kernels.

Step 2 (pre-image) Prediction: In order to compute the structured prediction $f(x)$ for an input x, we solve the following pre-image problem:

$$\begin{aligned}
f(x) &= \arg\min_{y\in\mathcal{Y}} \|g(x) - \Phi_l(y)\|^2_{\mathcal{F}_\mathcal{Y}} \\
&= \arg\min_{y\in\mathcal{Y}} \|\sum_{i=1}^n K(x_i, x)\psi_i - \Phi_l(y)\|^2_{\mathcal{F}_\mathcal{Y}} \\
&= \arg\min_{y\in\mathcal{Y}} \|\mathbf{K}_x\mathbf{\Psi} - \Phi_l(y)\|^2_{\mathcal{F}_\mathcal{Y}} \\
&= \arg\min_{y\in\mathcal{Y}} \|\mathbf{K}_x(\mathbf{K} + \lambda I)^{-1}\mathbf{\Phi_l} - \Phi_l(y)\|^2_{\mathcal{F}_\mathcal{Y}} \\
&= \arg\min_{y\in\mathcal{Y}}\ l(y, y) - 2\langle \mathbf{K}_x(\mathbf{K} + \lambda I)^{-1}\mathbf{\Phi_l}, \Phi_l(y)\rangle_{\mathcal{F}_\mathcal{Y}},
\end{aligned}$$

where $\mathbf{K}_x$ is the row vector of operators corresponding to the input x. At this step, the operator-valued kernel formulation has an inherent difficulty in expressing the pre-image problem without kernel mappings. Indeed, Φ_l can be unknown and only implicitly defined through the kernel l. In this case, the usual kernel trick is not sufficient to solve the pre-image problem. Thus, we introduce the following variant (generalization) of the kernel trick: $\langle \mathcal{T}\Phi_l(y_1), \Phi_l(y_2)\rangle_{\mathcal{F}_\mathcal{Y}} = [\mathcal{T}l(y_1, \cdot)](y_2)$, where $\mathcal{T}$ is an operator in $\mathcal{L}(\mathcal{F}_\mathcal{Y})$. Note that the usual kernel trick $\langle \Phi_l(y_1), \Phi_l(y_2)\rangle_{\mathcal{F}_\mathcal{Y}} = l(y_1, y_2)$ is recovered when $\mathcal{T}$ is the identity operator. It is easy to check that our proposed trick holds if we consider the feature space associated to the kernel l, i.e., $\Phi_l(y) = l(y, .)$. A proof for the more general case in which the features Φ_l can be any implicit mapping of a Mercer kernel is given for self-adjoint operator $\mathcal{T}$ in Appendix 7.1 in the supplementary material. Using this trick, the pre-image problem can be written as

$$f(x) = \arg\min_{y\in\mathcal{Y}}\ l(y, y) - 2\big[\mathbf{K}_x(\mathbf{K} + \lambda I)^{-1}\mathbf{L}_\bullet\big](y), \tag{4}$$

where $\mathbf{L}_\bullet$ is the column vector whose ith element is $l(y_i, \cdot)$. We can now express $f(x)$ only using kernel functions. Note that the regression and prediction step solutions of the KDE formulation in Cortes et al. (2005) can be recovered from Equations (3) and (4) using an operator-valued kernel K of the form $K(x_i, x_j) = k(x_i, x_j)\mathcal{I}$, in which k is a scalar-valued kernel on $\mathcal{X}$ and $\mathcal{I}$ is the identity operator in $\mathcal{F}_\mathcal{Y}$.

Once we have a general formulation of KDE using operator-valued kernels, we turn our attention to the problem of building operator-valued kernels suitable to take into account the structure of the kernel feature space $\mathcal{F}_\mathcal{Y}$ and then output and input-output correlations. This is described in the next section.

3 Covariance-based Operator-valued Kernels

In this section, we study the problem of designing operator-valued kernels suitable for structured outputs in the KDE formulation. This is quite important in order to take full advantage of the operator-valued KDE formulation. In fact, the main purpose of using the operator-valued kernel formulation is to take into account the dependencies between the variables $\Phi_l(y_i)$, i.e., the projection of y_i in the feature space $\mathcal{F}_\mathcal{Y}$, with the objective of capturing the structure of the output data encapsulated in $\Phi_l(y_i)$. Operator-valued kernels have been studied more in the context of multi-task learning, where the output is assumed to be in $\mathbb{R}^d$ with d the number of tasks (Evgeniou et al., 2005; Caponnetto et al., 2008). Some work has also been focused on extending these kernels to the domain of functional data analysis to deal with the problem of regression with functional responses, where outputs are considered to be in the L^2-space (Lian, 2007; Kadri et al., 2010). However, the operator- valued Kernels used in these contexts for discrete (vector) or continuous (functional) outputs cannot be used in our formulation. This is because in our case the feature space $\mathcal{F}_\mathcal{Y}$ can be known only implicitly by the output kernel l, and depending on l, $\mathcal{F}_\mathcal{Y}$ can be finite or infinite dimensional. Therefore, we focus our attention to operators that act on scalar-valued RKHSs. Covariance operators on RKHS have recently received a considerable amount of attention in the machine learning community. These operators that provide the simplest measure of dependency have been successfully applied to the problem of dimensionality reduction (Fukumizu et al., 2004), and played an important role in dealing with a number of statistical test problems (Gretton et al., 2005). We use the following covariance-based operator-valued kernel in our KDE formulation:

$$K(x_i, x_j) = k(x_i, x_j)C_{YY}, \tag{5}$$

where k is a scalar-valued kernel on $\mathcal{X}$ and $C_{YY} : \mathcal{F}_\mathcal{Y} \to \mathcal{F}_\mathcal{Y}$ is the covariance operator defined as

$$\langle \varphi_i, C_{YY}\varphi_j \rangle_{\mathcal{F}_\mathcal{Y}} = \mathbb{E}\big[\varphi_i(\mathcal{Y})\varphi_j(\mathcal{Y})\big].$$

The empirical covariance operator $\widehat{C}_{YY}^{(n)}$ is given by

$$\widehat{C}_{YY}^{(n)} = \frac{1}{n}\sum_{i=1}^{n} l(\cdot, y_i) \otimes l(\cdot, y_i), \tag{6}$$

where $\otimes$ is the tensor product $(\varphi_1 \otimes \varphi_2)h = \langle \varphi_2, h\rangle\varphi_1$. The operator-valued kernel (5) is nonnegative since

$$\sum_{i,j=1}^{m} \langle K(x_i, x_j)\varphi_i, \varphi_j\rangle_{\mathcal{F}_\mathcal{Y}} = \sum_{i,j=1}^{m} \langle k(x_i, x_j)\widehat{C}_{YY}^{(n)}\varphi_i, \varphi_j\rangle$$
$$= \sum_{p=1}^{n}\sum_{i,j=1}^{m} \frac{1}{n}\langle l(., y_p), \varphi_i\rangle k(x_i, x_j)\langle l(., y_p), \varphi_j\rangle \geq 0.$$

The last step is due to positive semi-definiteness of the scalar-valued kernel k.

The operator-valued kernel in Eq. (5) is a separable kernel since it operates on the output space and then encodes the interactions between the outputs without any reference to the input space. Although this property can be

restrictive in specifying input-output correlations, because of their simplicity, most of the operator-valued kernels proposed in the literature belong to this category (see Álvarez et al. (2011), for a review of separable and beyond separable operator-valued kernels). To address this issue, we propose a variant of the kernel in Eq. (5) based on the conditional covariance operator rather than the covariance operator and define it as follows:

$$K(x_i, x_j) = k(x_i, x_j) C_{YY|X}, \tag{7}$$

where $C_{YY|X} = C_{YY} - C_{YX} C_{XX}^{-1} C_{XY}$ is the conditional covariance operator on $\mathcal{F}_\mathcal{Y}$. This operator allows the operator-valued kernel to simultaneously encode the correlations between the outputs and to take into account (non-parametrically) the effects of the input data. In Proposition 1 (proof in Appendix 7.2 in the supplementary material), we show how the pre-image problem (4) can be formulated using the covariance-based operator-valued kernels in Eqs. (5) and (7).

Proposition 1 *The Gram matrix expression of the pre-image problem in Eq.* (4) *can be written for the case of the covariance operator-valued kernels, Eq.* (5)*, and conditional covariance operator-valued kernels, Eq.* (7)*, as follows:*

$$\arg\min_{y \in \mathcal{Y}} \; l(y,y) - 2\mathbf{L}_y^\top (\mathbf{k}_x^\top \otimes \mathbf{T})(\mathbf{k} \otimes \mathbf{T} + n\lambda I_{n^2})^{-1} \operatorname{vec}(I_n),$$

where $\mathbf{T} = \mathbf{L}$ *for the covariance operator and* $\mathbf{T} = \mathbf{L} - (\mathbf{k} + n\epsilon I_n)^{-1}\mathbf{k}\mathbf{L}$ *for the conditional covariance operator in which* ϵ *is a regularization parameter required for the operator inversion,* $\mathbf{k}$ *and* $\mathbf{L}$ *are Gram matrices associated to the scalar-valued kernels* k *and* l*,* $\mathbf{k}_x$ *and* $\mathbf{L}_y$ *are the column vectors* $\big(k(x,x_1),\ldots,k(x,x_n)\big)^\top$ *and* $\big(l(y,y_1),\ldots,l(y,y_n)\big)^\top$*,* vec *is the vector operator such that* $\operatorname{vec}(A)$ *is the vector of columns of the matrix* A*, and finally* $\otimes$ *is the Kronecker product.*

4 Related Work

In this section, we provide a brief overview of the related works and compare them with our proposed operator-valued kernel formulation. This comparison indicates that our formulation is an important step towards a general kernelization framework to deal with complex input/output problems.

4.1 KDE

The main goal of our operator-valued KDE is to generalize KDE by taking into account output and input-output correlations. Existing KDE formulations try to address these issues either by performing a kernel PCA to decorrelate the outputs (Weston et al., 2003), or by incorporating some form of prior knowledge in the regression step using regularization (Cortes et al., 2007). Our KDE formulation using (conditional) covariance-based operator-valued kernel is a more natural way to take into account these dependencies, since it does not require any particular form of the regression model. Moreover, compared to kernel PCA, it does not need a dimension reduction step which may imply a loss of information when the spectrum of the output kernel matrix does not decrease

rapidly, and assumes that the dimensionality of the low-dimensional subspace (the number of principal components) is known and fixed in advance. Also, it should be noted that the conditional covariance-based operator-valued kernel formulation, contrary to previous KDE formulations, succeeds to take into account the effect of the explanatory variables (input data).

4.2 Joint Kernels Meet Operator-valued Kernels

Another approach to take into account output and input-output correlations is to use joint kernels that are scalar-valued functions (similarity measure) of input-output pairs (Weston et al., 2007). The problem of learning the mapping f from $\mathcal{X}$ to $\mathcal{Y}$ is now reformulated as learning a function $\hat{f}$ from $\mathcal{X} \times \mathcal{Y}$ to $\mathbb{R}$ using a joint kernel J (Tsochantaridis et al., 2005). Our proposed operator-valued kernel formulation includes the joint kernel approach. In fact, like joint kernels, operator-valued kernels induce (implicitly) a similarity measure between input-output pairs. This can be seen from the feature space formulation of operator-valued kernels (Caponnetto et al., 2008; Kadri et al., 2011). A feature map associated with an operator-valued kernel K is a continuous function Φ such that

$$\langle K(x_1, x_2)\Phi_l(y_1), \Phi_l(y_2)\rangle = \langle \Phi_k\big(x_1, \Phi_l(y_1)\big), \Phi_k\big(x_2, \Phi_l(y_2)\big)\rangle.$$

So the joint kernel is an inner product between an output $\Phi_l(y_2)$ and the result of applying the operator-valued kernel $K(x_1, x_2)$ to another output $\Phi_l(y_1)$. We now show how some of the joint kernels proposed in the literature (Weston et al., 2007) can be recovered by a suitable choice of operator-valued kernel.

1) Tensor Product Joint Kernel $J\big((x_1, y_1), (x_2, y_2)\big) = k(x_1, x_2)l(y_1, y_2)$ can be recovered from the operator-valued kernel $K(x_1, x_2) = k(x_1, x_2)I$.

2) Diagonal Regularization Joint Kernel

$$J\big((x_1, y_1), (x_2, y_2)\big) = (1-\lambda)k(x_1, x_2)\langle y_1, y_2\rangle + \lambda \sum_{i=1}^{q} x_1^i x_2^i y_1^i y_2^i$$

can be recovered by selecting the output kernel $l(y_1, y_2) = \langle y_1, y_2\rangle$ and the operator-valued kernel $K(x_1, x_2) = \big[(1-\lambda)k(x_1, x_2)\big]I + \lambda \odot_{x_1 \odot x_2}$, where $\odot$ is the point-wise product operator.

4.3 Dimensionality Reduction in Supervised Learning with RKHS

Another interesting observation is the link between our formulation and the work by Fukumizu et al. (2004) on kernel dimensionality reduction in regression. While these works tackle two different problems, they both use the conditional covariance operator as their main tool. Fukumizu et al. (2004) state that their method "cannot be viewed as a kernelization of an underlying linear algorithm". However, we believe that using our proposed framework with the conditional operator-valued kernel would be possible to show that the method proposed by Fukumizu et al. (2004) is in fact the kernelization of a linear algorithm.

Algorithm	λ	σ_k	σ_l	**RBF Loss**
KDE - Cortes	0.01	0.1	10	1.0423 ± 0.0996
KDE - Weston[3]	0.001	0.07	7	1.0246 ± 0.0529
KDE - Covariance	0.1	1	12	0.7616 ± 0.0304
KDE - Conditional Covariance	0.1	1	12	0.6241 ± 0.0411

Table 2: Performance (mean and standard deviation of RBF loss) of the KDE algorithms of Weston et al. (2003) and Cortes et al. (2005) and our proposed KDE method with covariance and conditional covariance operator-valued kernels on an image reconstruction problem of handwritten digits.

5 Experimental Results

In this section, we evaluate our operator-valued kernel formulation on an image reconstruction problem and an optical character recognition task. While the output of the first problem is numeric, the second problem has non-numerical output. In the first problem, we compare our method using both covariance and conditional covariance operator-valued kernels with the KDE algorithms of Weston et al. (2003) and Cortes et al. (2005). In the second problem, we only evaluate the two implementations of our KDE method.

5.1 Image Reconstruction

Here we consider the image reconstruction problem used in Weston et al. (2003). This problem takes the top half (the first 8 pixel lines) of a USPS postal digit as input and estimates its bottom half. We apply our KDE method using both covariance and conditional covariance operator-valued kernels and compare it with the KDE algorithms of Weston et al. (2003) and Cortes et al. (2005). In all these methods, we use RBF kernels for both input and output with the parameters shown in Table 2. This table also contains the ridge parameter used by these algorithms. We tried a number of values for these parameters and those in the table yielded the best performance.

We perform 5-fold cross validation on 600 digits selected randomly from the first 1000 digits of the USPS handwritten 16 by 16 pixel digit database, training with a single fold (200 examples as in Weston et al. (2003)) and testing on the remainder. Given a test input, we solve the problem and then choose as output the pre-image from the training data that is closest to this solution. The loss function used to evaluate the prediction $\hat{y}$ for an output y is the RBF loss induced by the output kernel, i.e., $||\Phi_l(y) - \Phi_l(\hat{y})||^2 = 2 - 2\exp\big(-||y-\hat{y}||^2/(2\sigma_l^2)\big)$. Table 2 shows the mean and standard deviation of the RBF loss and the kernel and ridge parameters for the four KDE algorithms described above.

Our proposed operator-valued kernel approach produced promising results in this experiment. Improvements in prediction accuracy obtained over the previous KDE algorithms can be explained by the fact that using the covariance operator we can capture the output correlations, and using the conditional co-

[3]These results are obtained using the Spider toolbox available at www.kyb.mpg.de/bs/people/spider.

Algorithm	λ	**WRC(%)**
KDE - Covariance	0.01	83.8 ± 5
KDE - Conditional Covariance	0.01	86.1 ± 4

Table 3: Performance (mean and standard deviation of Well Recognized word Characters -WRC-) of our proposed KDE method with covariance and conditional covariance operator-valued kernels on an optical character recognition (OCR) task.

variance we can take also into account information from the inputs. In this context, kPCA-based KDE (the algorithm by Weston et al. (2003)) performs better than the KDE formulation of Cortes et al. (2005). Indeed, the KDE formulation of Cortes et al. (2005) is equivalent to using an identity-based operator-valued kernel in our formulation, and thus, incapable in considering dependencies in the output feature space (contrary to the other methods considered here).

5.2 Optical Character Recognition

In order to evaluate the effectiveness of our proposed method in problems with non-numerical outputs, we use an optical character recognition (OCR) problem. This problem is the one used in Taskar et al. (2004) and Cortes et al. (2007). The dataset is a subset of the handwritten words collected by Rob Kassel at the MIT Spoken Language Systems Group. It contains $6,877$ word instances with a total of $52,152$ characters. The image of each character has been normalized into a 16 by 8 binary-pixel representation. The OCR task consists in predicting a word from the sequence of pixel-based images of its handwritten segmented characters.

Table 3 reports the results of our experiments. The performance is measured as the percentage number of word characters correctly recognized (WCR). These results are obtained using 5-fold cross validation on 100 words (about 800 characters), training with a single fold (20 words) and testing on the remainder. In our experiment, we use a polynomial kernel of third degree on the image characters. We use the input and output feature maps proposed in Cortes et al. (2007). The feature map associated to an input sequence of images $x = x_1 \dots x_q$ is defined by $\Phi_k(x) = \left[k(c_1, x_{v(c_1)}), \dots, k(c_N, x_{v(c_N)})\right]^\top$ where c_m, $m = 1, \dots, N$, are all the image segments in the training set, $v(c_m)$ is the position of the character c_m in the word, and $k(c_m, x_{v(c_m)}) = 0$ if $v(c_m) > q$. For the output space, the feature map $\Phi_l(y)$ associated to an output string $y = y_1, \dots, y_q$ is a $26p$-dimensional vector defined by $\Phi_l(y) = \left[\phi_l(y_1),\ ldots, \phi_l(y_q), 0, \dots, 0\right]^\top$, where p is the maximum length of a sequence of images in the training set, and $\phi_l(y_j), 1 \leqslant j \leqslant q$, is a 26-dimensional vector whose components are all zero except from the entry of index y_j. With this output feature space, the pre-image is easily computed since each position can be obtained separately. Note that this occurs since with the OCR dataset a one-to-one mapping of images to characters is provided.

Experiments on OCR task support the results obtained in the image reconstruction problem. The covariance based operator-valued kernel operates on the output space and encodes the interactions between the outputs without any

reference to the input space, while using the conditional covariance we consider also the effects of the input variables which allow to improve performance.

6 Conclusions and Future Work

In this paper, we studied the problem of structured output learning using kernel dependency estimation (KDE) methods. We provided a general formulation of this problem using operator-valued kernels. Several existing KDE methods including those by Weston et al. (2003) and Cortes et al. (2005) can be recovered from this formulation by choosing a particular operator-valued kernel. We then defined a covariance-based operator-valued kernel for our proposed KDE formulation. This kernel takes into account the structure of the output kernel feature space and encodes the interactions between the outputs, but makes no reference to the input space. We addressed this issue by introducing a variant of our KDE method based on the conditional covariance operator that in addition to the correlation between the outputs takes into account the effects of the input variables. Finally, we evaluated the performance of our KDE method using both covariance and conditional covariance kernels on two structured output problems, and compared it to the state-of-the-art kernel-based structured output regression methods.

In our both experiments, we used relatively small training and test sets. Experiments on larger datasets are necessary in order to have a better evaluation of our method in general, and to better observe the potential advantage of the conditional covariance kernel w.r.t. the covariance kernel and other KDE methods. Another interesting direction for future research is to investigate operator-valued kernels in the context of classification-based structured output learning (Tsochantaridis et al., 2005) and to compare the resulting algorithms with structured SVMs.

7 Appendix

7.1 (Generalized) Kernel Trick

In this section, we prove the (generalized) kernel trick used in the paper, i.e., $\langle \mathcal{T}\Phi_l(y_1), \Phi_l(y_2)\rangle_{\mathcal{F}_\mathcal{Y}} = [\mathcal{T}l(y_1,\cdot)](y_2)$, where $\mathcal{T} \in \mathcal{L}(\mathcal{F}_\mathcal{Y})$ and $\mathcal{F}_\mathcal{Y}$ is a RKHS with kernel l.

Case 1: Φ_l is the feature map associated to the reproducing kernel l, i.e., $\Phi_l(y) = l(\cdot, y)$.

Here the proof is straightforward, we may write

$$\begin{aligned}\langle \mathcal{T}\Phi_l(y_1), \Phi_l(y_2)\rangle_{\mathcal{F}_\mathcal{Y}} &= \langle \mathcal{T}l(\cdot, y_1), l(\cdot, y_2)\rangle_{\mathcal{F}_\mathcal{Y}} \\ &= [\mathcal{T}l(y_1,\cdot)](y_2).\end{aligned}$$

The second equality follows from the reproducing property.

Case 2: Φ_l is an implicit feature map of a Mercer kernel, i.e., $\mathcal{T}$ is a self-adjoint operator in $\mathcal{L}(\mathcal{F}_\mathcal{Y})$.

We first recall the Mercer's theorem:

Theorem 2 (Mercer's theorem) *Suppose that l is a symmetric real-valued kernel on $\mathcal{Y}^2$ such that the integral operator $T_l : L_2(\mathcal{Y}) \to L_2(\mathcal{Y})$, defined as*

$$(T_l f)(y_1) := \int_{\mathcal{Y}} l(y_1, y_2) f(y_2) dy_2$$

is positive. Let $\gamma_j \in L_2(\mathcal{Y})$ be the normalized eigenfunctions of T_l associated with the eigenvalues $\lambda_j > 0$, sorted in non-increasing order. Then

$$l(y_1, y_2) = \sum_{j=1}^{N_f} \lambda_j \gamma_j(y_1) \gamma_j(y_2) \qquad (*)$$

holds for almost all (y_1, y_2) with $N_f \in \mathbb{N}$.

Since l is a Mercer kernel, the eigenfunctions $(\gamma_i)_{i=1}^{N_f}$ can be chosen to be orthogonal w.r.t. the dot product in $L_2(\mathcal{Y})$. Hence, it is straightforward to construct a dot product $\langle \cdot, \cdot \rangle$ such that $\langle \gamma_i, \gamma_j \rangle = \delta_{ij}/\lambda_j$ (δ_{ij} is the Kronecker delta) and the orthonormal basis $\big(e_j\big)_{j=1}^{N_f} = \big(\sqrt{\lambda_j}\gamma_j\big)_{j=1}^{N_f}$ (see Schölkopf et al. (1999) for more details). Therefore, the feature map associated to the Mercer kernel l is of the following form:

$$\Phi_l : y \longmapsto \big(\sqrt{\lambda_j}\gamma_j(y)\big)_{j=1}^{N_f} .$$

Now we compute $[\mathcal{T} l(y_1, \cdot)](y_2)$ using $(*)$:

$$\begin{aligned}
[\mathcal{T} l(y_1, \cdot)](y_2) &= \sum_{j=1}^{N_f} \lambda_j \gamma_j(y_1) \big[\mathcal{T}\gamma_j\big](y_2) \\
&= \sum_{i,j=1}^{N_f} \lambda_j \gamma_j(y_1) \langle \mathcal{T}\gamma_j, e_i \rangle e_i(y_2) \\
&= \sum_{i,j=1}^{N_f} \lambda_j \gamma_j(y_1) \lambda_i \langle \mathcal{T}\gamma_j, \gamma_i \rangle \gamma_i(y_2). \qquad \text{(i)}
\end{aligned}$$

Now let $\widehat{\mathcal{T}} = \big(\widehat{\mathcal{T}}_{ij}\big)_{i,j=1}^{N_f}$ be the matrix representation of the operator $\mathcal{T}$ in the basis $\big(e_j\big)_{j=1}^{N_f}$, thus, by definition we have $\widehat{\mathcal{T}}_{ij} = \langle \mathcal{T} e_i, e_j \rangle$. Using this and the

feature map expression of a Mercer kernel, we obtain

$$\begin{aligned}
\langle \mathcal{T}\Phi_l(y_1), \Phi_l(y_2)\rangle_{\mathcal{F}_\mathcal{Y}} &= \sum_{i=1}^{N_f} \big(\widehat{\mathcal{T}}\Phi_l(y_1)\big)_i \big(\Phi_l(y_2)\big)_i \\
&= \sum_{i=1}^{N_f} \left(\sum_{j=1}^{N_f} \widehat{\mathcal{T}}_{ij} \sqrt{\lambda_j}\gamma_j(y_1) \right) \sqrt{\lambda_i}\gamma_i(y_2) \\
&= \sum_{i,j=1}^{N_f} \langle \mathcal{T} e_i, e_j\rangle \sqrt{\lambda_j}\gamma_j(y_1)\sqrt{\lambda_i}\gamma_i(y_2) \\
&= \sum_{i,j=1}^{N_f} \langle \mathcal{T}\sqrt{\lambda_i}\gamma_i, \sqrt{\lambda_j}\gamma_j\rangle \sqrt{\lambda_j}\gamma_j(y_1)\sqrt{\lambda_i}\gamma_i(y_2) \\
&= \sum_{i,j=1}^{N_f} \lambda_j\gamma_j(y_1)\langle \mathcal{T}\gamma_i, \gamma_j\rangle \lambda_i\gamma_i(y_2) \\
&= \sum_{i,j=1}^{N_f} \lambda_j\gamma_j(y_1)\langle \mathcal{T}\gamma_j, \gamma_i\rangle \lambda_i\gamma_i(y_2). \qquad \text{(ii)}
\end{aligned}$$

Note that the last equality follows from the fact that $\mathcal{T}$ is a self-adjoint operator. The proof follows from (i) and (ii).

7.2 Proof of Proposition 1

In this section, we provide the proof of Proposition 1. We only show the proof for the covariance-based operator-valued kernels, since the proof for the other case (conditional covariance-based operator-valued kernels) is quite similar. Note that the pre-image problem is of the form

$$f(x) = \operatorname*{arg\,min}_{y\in\mathcal{Y}} \; l(y,y) - 2\big[\mathbf{K}_x(\mathbf{K}+\lambda I)^{-1}\mathbf{L}_\bullet\big](y),$$

and our goal is to compute its Gram matrix expression[3] in case $K(x_i,x_j) = k(x_i,x_j)\widehat{C}_{YY}^{(n)}$, where $\widehat{C}_{YY}^{(n)}$ is the empirical covariance operator defined as

$$\widehat{C}_{YY}^{(n)} = \frac{1}{n}\sum_{i=1}^{n} l(\cdot,y_i)\otimes l(\cdot,y_i).$$

Let $h = (h_i)_{i=1}^n$ be a vector of variables in the RKHS $\mathcal{F}_\mathcal{Y}$ such that $h = (\mathbf{K}+\lambda I)^{-1}\mathbf{L}_\bullet$. Since each h_i is in the RKHS $\mathcal{F}_\mathcal{Y}$, it can be decomposed as follows:

$$h_i = \alpha_{(i)}^\top \mathbf{L}_\bullet + h_\perp = \sum_{j=1}^{n} \alpha_{(i)}^j l(\cdot,y_j) + h_\perp,$$

where $\alpha_{(i)} \in \mathbb{R}^n$, $\mathbf{L}_\bullet = \big(l(\cdot,y_1),\ldots,l(\cdot,y_n)\big)^\top$, and $h_\perp$ is orthogonal to all $l(\cdot,y_i)$'s, $i=1,\ldots,n$. The idea here is similar to the one used by Fukumizu et al. (2011). Now we may write

$$(\mathbf{K}+\lambda I)h = \mathbf{L}_\bullet,$$

[3]Expressing the covariance operator $\widehat{C}_{YY}^{(n)}$ on the RKHS $\mathcal{F}_\mathcal{Y}$ using the kernel matrix $\mathbf{L}$

which gives us

$$\forall i \in 1, \ldots, n \qquad l(\cdot, y_i) = \sum_{j=1}^{n} \mathbf{K}_{ij} h_j + \lambda h_i.$$

Now using the empirical covariance operator, for each i, we may write

$$\begin{aligned}
l(\cdot, y_i) &= \sum_{j=1}^{n} k(x_i, x_j) \widehat{C}_{YY}^{(n)} h_j + \lambda h_i \\
&= \sum_{j=1}^{n} k(x_i, x_j) \Big(\frac{1}{n} \sum_{s=1}^{n} l(\cdot, y_s) \otimes l(\cdot, y_s) \Big) h_j + \lambda h_i \\
&= \sum_{j=1}^{n} \frac{1}{n} k(x_i, x_j) \Big(\sum_{s=1}^{n} l(\cdot, y_s) \otimes l(\cdot, y_s) \Big) \\
&\Big(\sum_{m=1}^{n} \alpha_{(j)}^{m} l(\cdot, y_m) + h_\perp \Big) + \lambda \sum_{m=1}^{n} \alpha_{(j)}^{m} l(\cdot, y_m) + \lambda h_\perp \\
&= \sum_{j=1}^{n} \frac{1}{n} k(x_i, x_j) \sum_{s=1}^{n} \sum_{m=1}^{n} \alpha_{(j)}^{m} l(y_s, y_m) l(\cdot, y_s) + 0 \\
&+ \lambda \mathbf{L}_{\bullet}^{\top} \alpha_{(i)} + \lambda h_\perp \\
&= \sum_{j=1}^{n} \frac{1}{n} k(x_i, x_j) \sum_{s=1}^{n} l(\cdot, y_s) \sum_{m=1}^{n} \alpha_{(j)}^{m} l(y_s, y_m) \\
&+ \lambda \mathbf{L}_{\bullet}^{\top} \alpha_{(i)} + \lambda h_\perp \\
&= \sum_{j=1}^{n} \frac{1}{n} k(x_i, x_j) \mathbf{L}_{\bullet}^{\top} \mathbf{L} \alpha_{(j)} + \lambda \mathbf{L}_{\bullet}^{\top} \alpha_{(i)} + \lambda h_\perp.
\end{aligned}$$

Now if take the inner-product of the above equation with all $l(y_s, \cdot),\ s = 1, \ldots, n$, we obtain that for each i

$$\begin{aligned}
l(y_i, y_s) &= \sum_{j=1}^{n} \frac{1}{n} k(x_i, x_j) \langle l(\cdot, y_s), \mathbf{L}_{\bullet}^{\top} \mathbf{L} \alpha_{(j)} \rangle \\
&+ \lambda \langle l(\cdot, y_s), \mathbf{L}_{\bullet}^{\top} \alpha_{(i)} \rangle + \lambda \langle l(\cdot, y_s), h_\perp \rangle \\
&= \sum_{j=1}^{n} \frac{1}{n} k(x_i, x_j) \mathbf{L}_{y_s}^{\top} \mathbf{L} \alpha_{(j)} + \lambda \mathbf{L}_{y_s}^{\top} \alpha_{(i)},
\end{aligned}$$

which gives us the vector form

$$\mathbf{L}_{y_i} = \sum_{j=1}^{n} \frac{1}{n} k(x_i, x_j) \mathbf{L} \mathbf{L} \alpha_{(j)} + \lambda \mathbf{L} \alpha_{(i)}. \qquad (*)$$

Defining the $n \times n$ matrix $\boldsymbol{\alpha} = (\alpha_{(1)}, \ldots, \alpha_{(n)})$, we may write $(*)$ in a matrix form as

$$\mathbf{L} = \frac{1}{n} \mathbf{L} \mathbf{L} \boldsymbol{\alpha} \mathbf{k} + \lambda \mathbf{L} \boldsymbol{\alpha},$$

which gives us

$$\frac{1}{n} \mathbf{L} \boldsymbol{\alpha} \mathbf{k} + \lambda \boldsymbol{\alpha} = I_n.$$

Now using $\text{vec}(ABC) = (C^\top \otimes A)\,\text{vec}(B)$, we have

$$\big(\frac{1}{n}\mathbf{k} \otimes \mathbf{L} + \lambda I_{n^2}\big)\,\text{vec}(\boldsymbol{\alpha}) = \text{vec}(I_n). \qquad (**)$$

Now we may write

$$\begin{aligned}
&\mathbf{K}_x(\mathbf{K} + \lambda I)^{-1}\mathbf{L}_\bullet \;=\; \mathbf{K}_x h = \sum_{i=1}^{n} K(x, x_i)h_i \\
&= \sum_{i=1}^{n} k(x, x_i)\widehat{C}_{YY}^{(n)} h_i = \sum_{i=1}^{n} \frac{1}{n} k(x, x_i)\mathbf{L}_\bullet^\top \mathbf{L}\alpha_{(i)} \\
&= \frac{1}{n}\mathbf{L}_\bullet^\top \mathbf{L}\boldsymbol{\alpha}\mathbf{k}_x = \frac{1}{n}\mathbf{L}_\bullet^\top\,\text{vec}(\mathbf{L}\boldsymbol{\alpha}\mathbf{k}_x) = \frac{1}{n}\mathbf{L}_\bullet^\top\big(\mathbf{k}_x^\top \otimes \mathbf{L}\big)\,\text{vec}(\boldsymbol{\alpha}) \\
&= \mathbf{L}_\bullet^\top\big(\mathbf{k}_x^\top \otimes \mathbf{L}\big)\big(\mathbf{k} \otimes \mathbf{L} + n\lambda I_{n^2}\big)^{-1}\,\text{vec}(I_n),
\end{aligned}$$

where the last equality comes from $(**)$. Thus, we may write

$$f(x) = \operatorname*{arg\,min}_{y \in \mathcal{Y}}\; l(y, y) - 2\mathbf{L}_y^\top(\mathbf{k}_x^\top \otimes \mathbf{L})(\mathbf{k} \otimes \mathbf{L} + n\lambda I_{n^2})^{-1}\,\text{vec}(I_n),$$

which concludes our proof.

7.3 Computational Complexity

We now provide an efficient procedure to reduce the computational complexity of the Gram matrix expression of the pre-image solution given by Proposition 1. This solution is obtained by computing the expression $C(x, y)$ defined as follows

$$C(x, y) = l(y, y) - 2\mathbf{L}_y^\top(\mathbf{k}_x^\top \otimes \mathbf{L})(\mathbf{k} \otimes \mathbf{L} + n\lambda I_{n^2})^{-1}\,\text{vec}(I_n), \qquad (8)$$

Computing $C(x, y)$ consists of large matrix operations and necessitates the computation of the inverse of a $n^2 \times n^2$ matrix since it involves the Kronecker product between two kernel matrices. To solve this problem, we propose to apply incomplete Cholesky decomposition to the kernel matrices $\mathbf{k} \in \mathbb{R}^{n \times n}$ and $\mathbf{L} \in \mathbb{R}^{n \times n}$ Bach and Jordan (2002). This consists of finding the matrices $\mathbf{U} \in \mathbb{R}^{n \times m_1}$ and $\mathbf{V} \in \mathbb{R}^{n \times m_2}$, with $m_1 \ll n$ and $m_2 \ll n$, such that:

$$\mathbf{k} = \mathbf{U}\mathbf{U}^\top,\; \mathbf{L} = \mathbf{V}\mathbf{V}^\top.$$

Using this decomposition in Equation 8, we obtain:

$$\begin{aligned}
&C(x, y) = l(y, y) - 2\mathbf{L}_y^\top(\mathbf{k}_x^\top \otimes \mathbf{L})[\mathbf{U}\mathbf{U}^\top \otimes \mathbf{V}\mathbf{V}^\top + n\lambda I_{n^2}]^{-1}\,\text{vec}(I_n) \\
&= l(y, y) - 2\mathbf{L}_y^\top(\mathbf{k}_x^\top \otimes \mathbf{L})[(\mathbf{U} \otimes \mathbf{V})(\mathbf{U}^\top \otimes \mathbf{V}^\top) + n\lambda I_{n^2}]^{-1}\,\text{vec}(I_n) \\
&\text{using } (\mathbf{A} \otimes \mathbf{B})(\mathbf{C} \otimes \mathbf{D}) = \mathbf{A}\mathbf{C} \otimes \mathbf{B}\mathbf{D} \\
&= l(y, y) - 2\mathbf{L}_y^\top(\mathbf{k}_x^\top \otimes \mathbf{L})\frac{1}{n\lambda}[I_{n^2} - (\mathbf{U} \otimes \mathbf{V})(n\lambda I_{m_1 m_2} + (\mathbf{U}^\top \otimes \mathbf{V}^\top)(\mathbf{U} \otimes \mathbf{V}))^{-1}(\mathbf{U}^\top \otimes \mathbf{V}^\top)]\,\text{vec}(I_n) \\
&\text{using the Woodbury formula } (\mathbf{A} + \mathbf{B}\mathbf{C})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}(\mathbf{I} + \mathbf{C}\mathbf{A}^{-1}\mathbf{B})^{-1}\mathbf{C}\mathbf{A}^{-1} \\
&= l(y, y) - \frac{2}{n\lambda}\mathbf{L}_y^\top(\mathbf{k}_x^\top \otimes \mathbf{L})[\text{vec}(I_n) - (\mathbf{U} \otimes \mathbf{V})(n\lambda I_{m_1 m_2} + \mathbf{U}^\top\mathbf{U} \otimes \mathbf{V}^\top\mathbf{V})^{-1}(\mathbf{U}^\top \otimes \mathbf{V}^\top)\,\text{vec}(I_n)] \\
&= l(y, y) - \frac{2}{n\lambda}\mathbf{L}_y^\top[(\mathbf{k}_x^\top \otimes \mathbf{L})\,\text{vec}(I_n) - (\mathbf{k}_x^\top \otimes \mathbf{L})(\mathbf{U} \otimes \mathbf{V})(n\lambda I_{m_1 m_2} + \mathbf{U}^\top\mathbf{U} \otimes \mathbf{V}^\top\mathbf{V})^{-1}(\mathbf{U}^\top \otimes \mathbf{V}^\top)\,\text{vec}(I_n)] \\
&= l(y, y) - \frac{2}{n\lambda}\mathbf{L}_y^\top[\text{vec}(\mathbf{L}\mathbf{k}_x) - (\mathbf{k}_x^\top\mathbf{U} \otimes \mathbf{L}\mathbf{V})(n\lambda I_{m_1 m_2} + \mathbf{U}^\top\mathbf{U} \otimes \mathbf{V}^\top\mathbf{V})^{-1}\,\text{vec}(\mathbf{V}^\top\mathbf{U})] \\
&\text{using } \text{vec}(\mathbf{A}\mathbf{B}\mathbf{C}) = (\mathbf{C}^\top \otimes \mathbf{A})\,\text{vec}(\mathbf{B})
\end{aligned}$$

Since $(\mathbf{U}^\top\mathbf{U} \otimes \mathbf{V}^\top\mathbf{V}) \in \mathbb{R}^{m_1 m_2 \times m_1 m_2}$, the cost of the matrix inversion is reduced from $O(n^6)$ to $O((m_1 m_2)^3)$. For the space complexity, we do not need to create a $n^2 \times n^2$ matrix but only a $n \times m_1 m_2$ (or $m_1 m_2 \times m_1 m_2$) matrix.

Acknowledgments

The authors would like to thank Arthur Gretton for his helpful suggestions.

References

M. Álvarez, L. Rosasco, and N. D. Lawrence. Kernels for vector-valued functions: A review. Technical report, University of Manchester, Massachusetts Institute of Technology, and University of Sheffield, 2011.

R. Ando and T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853, 2005.

F. Bach and M. Jordan. Kernel independent component analysis. *Journal of Machine Learning Research*, 3:1–48, 2002.

G. Bakir, T. Hofmann, B. Schölkopf, A. Smola, B. Taskar, and S. Vishwanathan, editors. *Predicting Structured Data*. MIT Press, 2007.

K. Borgwardt, K. Tsuda, S. Vishwanathan, and X. Yan. *NIPS Workshop on "Structured Input - Structured Outputs"*. 2008.

U. Brefeld, T. Joachims, B. Taskar, and E. Xing. *ICML Workshop on "Learning in Structured Output Spaces"*. 2006.

C. Brouard, F. d'Alché Buc, and M. Szafranski. Semi-supervised penalized output kernel regression for link prediction. In *Proceedings of the 28th International Conference on Machine Learning*, pages 593–600, 2011.

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

A. Caponnetto, C. Micchelli, M. Pontil, and Y. Ying. Universal multi-task kernels. *Journal of Machine Learning Research*, 68:1615–1646, 2008.

R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

C. Cortes, M. Mohri, and J. Weston. A general regression technique for learning transductions. In *Proceedings of the Twenty-second International Conference on Machine Learning*, pages 153–160, 2005.

C. Cortes, M. Mohri, and J. Weston. *A General Regression Framework for Learning String-to-String Mappings*. MIT Press, 2007.

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

K. Fukumizu, F. Bach, and M. Jordan. Dimensionality reduction for supervised learning with reproducing kernel hilbert spaces. *Journal of Machine Learning Research*, 5:73–99, 2004.

K. Fukumizu, L. Song, and A. Gretton. Kernel bayes' rule. In *Neural Information Processing Systems (NIPS)*, 2011.

A. Gretton, R. Herbrich, A. Smola, O. Bousquet, and B. Schölkopf. Kernel methods for measuring independence. *Journal of Machine Learning Research*, 6:1–47, 2005.

H. Kadri, E. Duflos, P. Preux, S. Canu, and M. Davy. Nonlinear functional regression: a functional RKHS approach. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, JMLR: W&CP 9*, pages 111–125, 2010.

H. Kadri, A. Rabaoui, P. Preux, E. Duflos, and A. Rakotomamonjy. Functional regularized least squares classification with operator-valued kernels. In *Proceedings of the 28th International Conference on Machine Learning*, pages 993–1000, 2011.

H. Lian. Nonlinear functional models for functional responses in reproducing kernel Hilbert spaces. *The Canadian Journal of Statistics*, 35:597–606, 2007.

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

H. Müller, J. Wang, I. Dryden, J. Ramsay, and S. Marron. *SAMSI Research Program on "Analysis of Object Data"*. 2011.

J. Ramsay and B. Silverman. *Functional Data Analysis, 2nd edition*. Springer Verlag, New York, 2005.

B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. J. Smola. Input space vs. feature space in kernel-based methods. *IEEE Transactions on Neural Networks*, 10(5):1000–1017, 1999.

B. Taskar, C. Guestrin, and D. Koller. Max-margin markov networks. In *Advances in Neural Information Processing Systems 16*. 2004.

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

Z. Wang and J. Shawe-Taylor. A kernel regression framework for SMT. *Machine Translation*, 24(2):87–102, 2010.

J. Weston, O. Chapelle, A. Elisseeff, B. Schölkopf, and V. Vapnik. Kernel dependency estimation. In *Proceedings of the Advances in Neural Information Processing Systems 15*, pages 873–880, 2003.

J. Weston, G. BakIr, O. Bousquet, B. Schölkopf, T. Mann, and W. Noble. *Joint Kernel Maps*. MIT Press, 2007.

Centre de recherche INRIA Lille – Nord Europe
Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Grenoble – Rhône-Alpes : 655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399