Input Output Kernel Regression

Celine Brouard, Florence d'Alché-Buc, Marie Szafranski

Input Output Kernel Regression: Supervised and Semi-Supervised Structured Output Prediction with Operator-Valued Kernels

Céline Brouard[1,2], Florence d'Alché-Buc[1,3] and Marie Szafranski[1,4]
[1] IBISC, Université d'Évry Val d'Essonne, 91037 Évry cedex, France
[2] Helsinki Institute for Information Technology,
Department of Computer Science, Aalto University, 02150 Espoo, Finland
[3] LTCI, CNRS, Télécom ParisTech, Université Paris-Saclay, 75013 Paris, France
[4] ENSIIE & LaMME, Université d'Évry Val d'Essonne, CNRS, INRA, 91037 Évry cedex, France

October 16, 2015

Abstract

In this paper, we introduce a novel approach, called Input Output Kernel Regression (IOKR), for learning mappings between structured inputs and structured outputs. The approach belongs to the family of Output Kernel Regression methods devoted to regression in feature space endowed with some output kernel. In order to take into account structure in input data and benefit from kernels in the input space as well, we use the Reproducing Kernel Hilbert Space theory for vector-valued functions. We first recall the ridge solution for supervised learning and then study the regularized hinge loss-based solution used in Maximum Margin Regression. Both models are also developed in the context of semi-supervised setting. We also derive an extension of Generalized Cross Validation for model selection in the case of the least-square model. Finally we show the versatility of the IOKR framework on two different problems: link prediction seen as a structured output problem and multi-task regression seen as a multiple and interdependent output problem. Eventually, we present a set of detailed numerical results that shows the relevance of the method on these two tasks.

1 Introduction

Many real world applications involve objects with an explicit or implicit discrete structure. Texts, images and videos in document processing and retrieval as well as genes and proteins in computational biology are all examples of implicit structured data that we may want to use as inputs or outputs in a prediction system. Besides these structured objects, structured output prediction can also concern multiple outputs linked by some relationship that is relevant to take into account. Surprisingly, although a lot of attention has been paid to learning from structured inputs for now two decades, this problem, often referred as *structured output learning*, has emerged relatively recently as a field of interest in statistical learning. In the literature, structured output prediction has been addressed from two main angles. A first

angle consists in *discriminative learning algorithms* that provide predictions by maximizing a scoring function over the output space. Conditional Random Fields (Lafferty et al., 2001) and their extension to kernels (Lafferty et al., 2004) were first proposed for discriminative modeling of graph-structured data and sequence labeling. Other discriminative learning algorithms based on maximum margin such as structured SVM (Tsochantaridis et al., 2004, 2005), Maximum Margin Markov Networks (M^3N) (Taskar et al., 2004) or Maximum Margin Regression (Szedmak et al., 2005) have then be developed and thoroughly studied. A common approach to those methods consists in defining a linear scoring function based on the image of an input-output pair by a joint feature map. Both methods, either based on Conditional Random Fields or maximum-margin techniques, are costly to train and generally assume that the output set $\mathcal{Y}$ is discrete. Keeping the idea of a joint feature map over inputs and outputs, a generative method called Joint Kernel Support Estimation has been recently proposed (Lampert and Blaschko, 2009). In this approach, a one-class SVM is used to learn the support of the joint-probability density $p(x, y)$. More recently, another angle to structured output prediction, that we called *Output Kernel Regression* (OKR), has emerged around the idea of using the kernel trick in the output space and making predictions in a feature space associated to the output kernel. As a first example, the seminal work of Kernel Dependency Estimation (KDE) was based on the definition of an input kernel as well as an output kernel. After a first version using kernel PCA to define a finite-dimensional output feature space (Weston et al., 2003), a more general KDE framework consisting in learning a linear function from the input feature space to the output feature space was proposed by Cortes et al. (2005). In this setting, predictions in the original output space are retrieved by solving a pre-image problem. Interestingly, the idea of Output Kernel Regression can be implemented without defining an input kernel as it is shown with Output Kernel Tree-based methods (Geurts et al., 2006, 2007a,b). In these approaches, a regression tree whose outputs are linear combinations of the training outputs in the output feature space is built using the kernel trick in the output space: the loss function which is locally minimized during the construction only involves inner products between training outputs. These methods are not limited to discrete output sets and they do not require expensive computations to make a prediction nor to train the model. Combined in ensembles such as random forests and boosting, they exhibit excellent performance. However these tree-based approaches suffer from two drawbacks: trees do not take into account structured input data except by using a flat description of them and the associated (greedy) building algorithm cannot be easily extended to semi-supervised learning.

In this work, we therefore propose to extend the methodology of Output Kernel Regression to another large family of nonparametric regression tools that allows to tackle structured data in the input space as well as in the output space. Moreover we will show that this new family of tools is useful in a semi-supervised context. Called Input Output Kernel Regression, this novel family for structured output prediction from structured inputs relies on Reproducing Kernel Hilbert Spaces (RKHS) for vector-valued functions with the following specificity: the output vector belongs to some output feature space associated to a chosen output kernel, as introduced in the following works (Brouard et al., 2011; Brouard, 2013). Let us recall that in the case of scalar-valued functions, the RKHS theory offers a flexible framework for penalized regression as witnessed by the abundant literature on the subject (Wahba, 1990; Pearce and Wand, 2006). A penalized regression problem is

seen as a minimization problem in a functional space built on an input scalar-valued kernel. Depending the nature of the prediction problem, appropriate penalties can be defined and representer theorem can be proven, facilitating the minimization problem to be further solved. In the RKHS theory, regularization constraint on the geometry of the probability distribution of labeled and unlabeled data can also be added to perform semi-supervised regression (Belkin et al., 2006). When functions are vector-valued, the adequate RKHS theory makes use of operator-valued kernels (Pedrick, 1957; Senkene and Tempel'man, 1973; Micchelli and Pontil, 2005). Operator-valued kernels have already been proposed to solve problems of multi-task regression (Evgeniou et al., 2005; Baldassarre et al., 2012), structured classification (Dinuzzo et al., 2011), vector autoregression (Lim et al., 2013) as well as functional regression (Kadri et al., 2010). The originality of this work is to consider that the output space is a feature space associated to a chosen output kernel. This new approach not only enhances setting of pattern recognition tasks by requiring to pay attention on both input and output sets but also opens new perspectives in machine learning. It encompasses in a unique framework kernel-based regression tools devoted to structured inputs as well as structured outputs.

1.1 Related Works

The paper of Micchelli and Pontil (2005) is devoted to the problem of learning functions with output values in a Hilbert space. They present the RKHS theory for vector-valued functions and study some regularization functionals in this context. Based on the work of Micchelli and Pontil (2005), Caponnetto et al. (2008) addressed the issue of universality of operator-valued kernels. Álvarez et al. (2012) reviewed the different methods that have been proposed to design or learn kernel in multi-output or multi-task learning. In this review, they also analyzed the connections existing between the bayesian and regularization frameworks.

In Brouard et al. (2011), the RKHS theory for vector-valued functions was used to address the output kernel regression problem in the semi-supervised setting. This approach was used to solve the link prediction problem. By working in the framework of RKHS theory for vector-valued functions, we extended the manifold regularization framework introduced by Belkin et al. (2006) to functions with values in a Hilbert space. We have also shown that the first step of KDE (Cortes et al., 2005) is a special case of IOKR using a particular operator-valued kernel.

Kadri et al. (2013) studied a formulation of KDE using operator-valued kernels. The first step of this approach is identical to the IOKR framework developed in Brouard et al. (2011) and Brouard (2013). The second step consists in extending the pre-image step of KDE using the RKHS theory for functions with values in a Hilbert space. They also proposed two operator-valued kernels based on covariance operators. They show that using these operator-valued kernels allow to express the pre-image problem using only input and output Gram matrices.

In parallel of Brouard et al. (2011), Minh and Sindhwani (2011) generalized the manifold regularization framework proposed by Belkin et al. (2006) for semi-supervised learning to functions with values in a Hilbert space.

1.2 Contributions

We introduce Input Output Kernel Regression (IOKR), a novel class of penalized regression problems based on the definition of an output scalar-valued kernel and an input operator-valued kernel. This article is an extended version of Brouard et al. (2011), that addresses more generally the problem of structured output prediction. In this work, we present several contributions regarding the RKHS theory for functions with values in a Hilbert space. We present the representer theorem for vector-valued functions in the semi-supervised setting. Based on this representer theorem, we study two particular models obtained using two different loss functions: the *IOKR-ridge* model introduced in Brouard et al. (2011) and a new model called *IOKR-margin*. This model extends the Maximum Margin Regression (MMR) framework introduced by Szedmak et al. (2005) to operator-valued kernels and to the semi-supervised setting. In this paper, we also put the reformulation of Kernel Dependency Estimation proposed by Cortes et al. (2005) into perspective in the Output Kernel Regression framework. We present the solutions corresponding to decomposable kernels. In the case of the least-squared loss function, we describe a new tool for model selection, which was first introduced in Brouard (2013). The selection of the hyperparameters is done by estimating the averaged error obtained with leave-one-out cross-validation as a closed-form solution. We show the versatility of the IOKR framework on two different problems: link prediction and multi-task regression. Finally, we present numerical results obtained with IOKR on these two tasks.

1.3 Organization of the Paper

This paper is organized as follows. In Section 2, we introduce the Output Kernel Regression approach, which can be used to solve structured output prediction problems. In Section 3 we describe the RKHS theory devoted to vector-valued function and present our contributions to this theory in the supervised and semi-supervised settings. We also present in this section models based on decomposable operator-valued kernels. We then show in Section 4 that, in the case of the least-squares loss function, the leave-one-out criterion can be estimated by a closed-form solution. The Section 5 is devoted to the framework of Input Output Kernel Regression (IOKR). In Section 6, the IOKR approach is applied on different link prediction problems and is illustrated on a multi-task regression problem.

The notations used in this paper are summarized in Table 1.

2 From Output Kernel Regression to Input Output Kernel Regression

We consider the general regression task consisting in learning a mapping between an input set $\mathcal{X}$ and an output set $\mathcal{Y}$. We assume that both $\mathcal{X}$ and $\mathcal{Y}$ are sample spaces and that $\mathcal{S}_n = \{(x_i, y_i), i = 1...n\}$ is an i.i.d. sample drawn from the joint probability law $\mathcal{P}$ defined on $\mathcal{X} \times \mathcal{Y}$. Outputs are supposed to be structured, for example objects such as sequences, graphs, nodes in a graph, or simply vectors of interdependent variables. It is realistic to assume that one can build a similarity $\kappa_y : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ between the elements of the output set $\mathcal{Y}$, such that κ_y takes into account the inherent structure of the elements of $\mathcal{Y}$

Meaning	Symbol
number of labeled examples	ℓ
number of unlabeled examples	n
input set	$\mathcal{X}$
set of labeled examples	$\mathcal{X}_\ell$
union of the labeled and unlabeled sets	$\mathcal{X}_{\ell+n}$
output set	$\mathcal{Y}$
input scalar kernel	$\kappa_x : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$
output scalar kernel	$\kappa_y : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$
input feature space	$\mathcal{F}_x$
output feature space	$\mathcal{F}_y$
input feature map	$\varphi_x : \mathcal{X} \to \mathcal{F}_x$
output feature map	$\varphi_y : \mathcal{Y} \to \mathcal{F}_y$
set of bounded operators from an Hilbert space $\mathcal{F}$ to itself	$\mathcal{B}(\mathcal{F})$
set of bounded operators from $\mathcal{F}$ to an Hilbert space $\mathcal{G}$	$\mathcal{B}(\mathcal{F}, \mathcal{G})$
operator-valued kernel	$\mathcal{K}_x : \mathcal{X} \times \mathcal{X} \to \mathcal{B}(\mathcal{F}_y)$
reproducing kernel Hilbert space of $\mathcal{K}_x$	$\mathcal{H}, \mathcal{H}_{\mathcal{K}_x}$
canonical feature map of $\mathcal{K}_x$	$\phi_x : \mathcal{X} \to \mathcal{B}(\mathcal{F}_y, \mathcal{H})$
gram matrix of $\mathcal{K}_x$ on $\mathcal{X}_\ell$ and $\mathcal{X}_{\ell+n}$	$\mathbf{K}_{x_\ell}, \mathbf{K}_{x_{\ell+n}}$
gram matrix of κ_x on $\mathcal{X}_\ell$ and $\mathcal{X}_{\ell+n}$	$K_{x_\ell}, K_{x_{\ell+n}}$
gram matrix of κ_y on $\mathcal{Y}_\ell$	K_{y_ℓ}
graph laplacian	L
matrix vectorization	vec
Kronecker product	$\otimes$
Hadamard product (element-wise product)	$\circ$

Table 1: Notations used in this paper

and has the properties of a positive definite kernel. Then, due to the Moore-Aronszajn theorem (Aronszajn, 1950), there exists a Hilbert space $\mathcal{F}_y$, called a *feature space*, and a corresponding function $\varphi_y : \mathcal{Y} \to \mathcal{F}_y$, called a *feature map* such that:

$$\forall (y, y') \in \mathcal{Y} \times \mathcal{Y}, \kappa_y(y, y') = \langle \varphi_y(y), \varphi_y(y') \rangle_{\mathcal{F}_y}.$$

The regression problem between $\mathcal{X}$ and $\mathcal{Y}$ can be decomposed into two tasks (see Figure 1):

- the first task is to learn a function h from the set $\mathcal{X}$ to the Hilbert space $\mathcal{F}_y$
- the second one is to define or learn a function f from $\mathcal{F}_y$ to $\mathcal{Y}$ to provide an output in the set $\mathcal{Y}$.

We call the first task, *Output Kernel Regression* (OKR), referring to previous works based on Output Kernel Trees (OK3) (Geurts et al., 2006, 2007a) and the second task, a *pre-image problem*. In this paper, we develop a general theoretical and practical framework for the OKR task, allowing to deal with structured inputs as well as structured outputs. To illustrate our approach, we have chosen two structured output learning tasks which do not

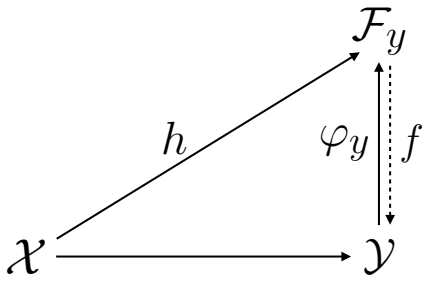

Figure 1: Schema of the Output Kernel Regression approach.

require to solve a pre-image problem. One is *multi-task regression* for which the dimension of the output feature space is finite, and the other one is *link prediction* for which prediction in the original set $\mathcal{Y}$ is not required. However, the approach we propose can be combined with pre-image solvers now available on the shelves. The interested reader may want to refer to Honeine and Richard (2011) or Kadri et al. (2013) to benefit from existing pre-image algorithms to solve structured output learning tasks.

In this work, we propose to build a family of models and learning algorithms devoted to *Output Kernel Regression* that present two additional properties compared to OK3-based methods: namely, models are able to take into account structure in input data and can be learned within the framework of penalized regression, enjoying various penalties including smoothness penalties for semi-supervised learning. To achieve this goal, we choose to use kernels both in the input and output spaces. As the models have values in a feature space and not in $\mathbb{R}$, we turn to the vector-valued reproducing kernel Hilbert spaces theory (Pedrick, 1957; Senkene and Tempel'man, 1973; Burbea and Masani, 1984) to provide a general framework for penalized regression of nonparametric vector-valued functions. In that theory, the values of kernels are operators on the output vectors which belong to some Hilbert space. Introduced in machine learning by the seminal work of Micchelli and Pontil (2005) to solve multi-task regression problems, operator-valued kernels (OVK) have then been studied under the angle of their universality (Caponnetto et al. (2008); Carmeli et al. (2010)) and developed in different contexts such as structured classification (Dinuzzo et al., 2011), functional regression (Kadri et al., 2010), link prediction (Brouard et al., 2011) or semi-supervised learning (Minh and Sindhwani, 2011; Brouard et al., 2011). With operator-valued kernels, models of the following form can be constructed:

$$\forall x \in \mathcal{X},\, h(x) = \sum_{i=1}^{n} \mathcal{K}_x(x, x_i)\mathbf{c}_i,\ \mathbf{c}_i \in \mathcal{F}_y, x_i \in \mathcal{X}, \tag{1}$$

extending nicely the usual kernel-based models devoted to real-valued functions.

In the case of IOKR, the output Hilbert space $\mathcal{F}_y$ is defined as a feature space related to a given output kernel. We therefore need to define a triplet $(\kappa_y, \mathcal{F}_y, \mathcal{K}_x)$ as a pre-requisite to solve the structured output learning task. By explicitly requiring to define an output kernel we emphasize the fact that an input operator-valued kernel cannot be defined without calling into question the output space, $\mathcal{F}_y$, and therefore, the output kernel κ_y. We will

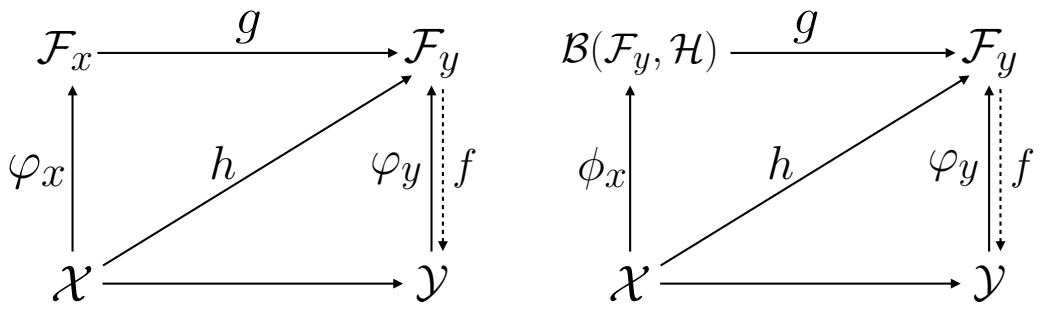

Figure 2: Diagrams describing Kernel Dependency Estimation (KDE) on the left and Input Output Kernel Regression (IOKR) on the right.

show in Section 6 that the same structured output prediction problem can be solved in different ways using different values for the triplet $(\kappa_y, \mathcal{F}_y, \mathcal{K}_x)$.

Interestingly, IOKR generalizes Kernel Dependency Estimation (KDE), a problem that was introduced in Weston et al. (2003) and was reformulated in a more general way by Cortes et al. (2005). If we call $\mathcal{F}_x$ a feature space associated to a scalar input kernel $\kappa_x : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and $\varphi_x : \mathcal{X} \to \mathcal{F}_x$ a corresponding feature map, KDE uses Kernel Ridge regression to learn a function h from $\mathcal{X}$ to $\mathcal{F}_y$ by building a function g from $\mathcal{F}_x$ to $\mathcal{F}_y$ and composing it with the feature map φ_x (see Figure 2). The function h is modeled as a linear function: $h(x) = W\varphi_x(x)$, where $W \in \mathcal{B}(\mathcal{F}_x, \mathcal{F}_y)$ is a linear operator from $\mathcal{F}_x$ to $\mathcal{F}_y$. The second phase consists in computing the pre-image of the obtained prediction.

In the case of IOKR, we build models of the general form introduced in Equation (1). Denoting ϕ_x the canonical feature map associated to the OVK $\mathcal{K}_x$, which is defined as: $\phi_x(x) = \mathcal{K}_x(\cdot, x)$, we can draw the chart depicted in Figure 2 on the right. The function ϕ_x maps inputs from $\mathcal{X}$ to $\mathcal{B}(\mathcal{F}_y, \mathcal{H})$. Indeed the value $\phi_x(x)y = \mathcal{K}_x(\cdot, x)y$ is a function of the RKHS $\mathcal{H}$ for all y in $\mathcal{F}_y$.

The model h is seen as the composition of a function g from $\mathcal{B}(\mathcal{F}_y, \mathcal{H})$ to the output feature space $\mathcal{F}_y$ and the input feature map ϕ_x. It writes as follows:

$$\forall x \in \mathcal{X},\ h(x) = \phi_x(x)^* \sum_{i=1}^{n} \phi_x(x_i)\mathbf{c}_i.$$

We can therefore see on Figure 2 how IOKR extends KDE. In Brouard et al. (2011), we have shown that we retrieve the model used in KDE when considering the following operator-valued kernel:

$$\mathcal{K}_x(x, x') = \kappa_x(x, x') * I,$$

where I is the identity operator from $\mathcal{F}_y$ to $\mathcal{F}_y$. Unlike KDE, that learns independently each component of the vectors $\varphi_y(y)$, IOKR takes into account the structure existing between these components.

The next section is devoted to the RKHS theory for vector-valued functions and to our contributions to this theory in the supervised and semi-supervised settings.

3 Operator-Valued Kernel Regression

In the following, we briefly recall the main elements of the RKHS theory devoted to vector-valued functions (Senkene and Tempel'man, 1973; Micchelli and Pontil, 2005) and then present our contributions to this theory.

Let $\mathcal{X}$ be a set and $\mathcal{F}_y$ a Hilbert space. In this section, no assumption is needed about the existence of an output kernel κ_y. We note $\tilde{\mathbf{y}}$ the vectors in $\mathcal{F}_y$. Given two Hilbert spaces $\mathcal{F}$ and $\mathcal{G}$, we note $\mathcal{B}(\mathcal{F},\mathcal{G})$ the set of bounded operators from $\mathcal{F}$ to $\mathcal{G}$ and $\mathcal{B}(\mathcal{F})$ the set of bounded operators from $\mathcal{F}$ to itself. Given an operator A, A^* denotes the adjoint of A.

Definition 1 *An operator-valued kernel on $\mathcal{X}\times\mathcal{X}$ is a function $\mathcal{K}_x : \mathcal{X}\times\mathcal{X} \to \mathcal{B}(\mathcal{F}_y)$ that verifies the two following conditions:*

- $\forall (x,x') \in \mathcal{X}\times\mathcal{X},\ \mathcal{K}_x(x,x') = \mathcal{K}_x(x',x)^*$,
- $\forall m \in \mathbb{N},\ \forall \mathcal{S}_m = \{(x_i,\tilde{\mathbf{y}}_i)\}_{i=1}^m \subseteq \mathcal{X}\times\mathcal{F}_y,\ \sum_{i,j=1}^m \langle \tilde{\mathbf{y}}_i, \mathcal{K}_x(x_i,x_j)\tilde{\mathbf{y}}_j\rangle_{\mathcal{F}_y} \geq 0$.

The following theorem shows that given any operator-valued kernel, it is possible to build a reproducing kernel Hilbert space associated to this kernel.

Theorem 2 (Senkene and Tempel'man (1973); Micchelli and Pontil (2005)) *Given an operator-valued kernel $\mathcal{K}_x : \mathcal{X}\times\mathcal{X} \to \mathcal{B}(\mathcal{F}_y)$, there is a unique Hilbert space $\mathcal{H}_{\mathcal{K}_x}$ of functions $h : \mathcal{X} \to \mathcal{F}_y$ which satisfies the following reproducing property:*

$$\forall h \in \mathcal{H}_{\mathcal{K}_x},\ \forall x \in \mathcal{X},\ h(x) = \mathcal{K}_x(x,\cdot)h,$$

where $\mathcal{K}_x(x,\cdot)$ is an operator in $\mathcal{B}(\mathcal{H}_{\mathcal{K}_x},\mathcal{F}_y)$.

As a consequence, $\forall x \in \mathcal{X}, \forall \tilde{\mathbf{y}} \in \mathcal{F}_y, \forall h \in \mathcal{H}_{\mathcal{K}_x},\ \langle \mathcal{K}_x(\cdot,x)\tilde{\mathbf{y}}, h\rangle_{\mathcal{H}_{\mathcal{K}_x}} = \langle \tilde{\mathbf{y}}, h(x)\rangle_{\mathcal{F}_y}$.

The Hilbert space $\mathcal{H}_{\mathcal{K}_x}$ is called the reproducing kernel Hilbert space associated to the kernel $\mathcal{K}_x$. This RKHS can be built by taking the closure of $\text{span}\{\mathcal{K}_x(\cdot,x)\boldsymbol{\alpha} \,|\, x \in \mathcal{X}, \boldsymbol{\alpha} \in \mathcal{F}_y\}$. The scalar product on $\mathcal{H}_{\mathcal{K}_x}$ between two functions $f = \sum_{i=1}^n \mathcal{K}_x(\cdot,x_i)\boldsymbol{\alpha}_i$ and $g = \sum_{j=1}^m \mathcal{K}_x(\cdot,t_j)\boldsymbol{\beta}_j$, $x_i, t_j \in \mathcal{X}$, $\boldsymbol{\alpha}_i, \boldsymbol{\beta}_j \in \mathcal{F}_y$, is defined as:

$$\langle f, g\rangle_{\mathcal{H}_{\mathcal{K}_x}} = \sum_{i=1}^n \sum_{j=1}^m \langle \boldsymbol{\alpha}_i, \mathcal{K}_x(x_i,t_j)\boldsymbol{\beta}_j\rangle_{\mathcal{F}_y}.$$

The corresponding norm $\|\cdot\|_{\mathcal{H}_{\mathcal{K}_x}}$ is defined by $\| f \|^2_{\mathcal{H}_{\mathcal{K}_x}} = \langle f, f\rangle_{\mathcal{H}_{\mathcal{K}_x}}$. For sake of simplicity we replace the notation $\mathcal{H}_{\mathcal{K}_x}$ by $\mathcal{H}$ in the rest of the paper.

As for scalar-valued functions, one of the most appealing feature of RKHS is to provide a theoretical framework for regularization with the representer theorems.

3.1 Regularization in Vector-Valued RKHS

Based on the RKHS theory for vector-valued functions, Micchelli and Pontil (2005) have proved a representer theorem for convex loss functions in the supervised case.

We note $S_\ell = \{(x_i,\tilde{\mathbf{y}}_i)\}_{i=1}^\ell \subseteq \mathcal{X}\times\mathcal{F}_y$ the set of labeled examples and $\mathcal{H}$ the RKHS with reproducing kernel $\mathcal{K}_x : \mathcal{X}\times\mathcal{X} \to \mathcal{B}(\mathcal{F}_y)$.

Theorem 3 (Micchelli and Pontil (2005)) *Let $\mathcal{L}$ be a convex loss function, and $\lambda_1 > 0$ a regularization parameter. The minimizer of the following optimization problem:*

$$\operatorname*{argmin}_{h \in \mathcal{H}} \mathcal{J}(h) = \sum_{i=1}^{\ell} \mathcal{L}(h(x_i), \tilde{\mathbf{y}}_i) + \lambda_1 \|h\|_{\mathcal{H}}^2 \ ,$$

admits an expansion:

$$\hat{h}(\cdot) = \sum_{j=1}^{\ell} \mathcal{K}_x(\cdot, x_j)\mathbf{c}_j \ ,$$

where the coefficients $\mathbf{c}_j, j = 1, \cdots, \ell$ are vectors in the Hilbert space $\mathcal{F}_y$.

In the following, we plug the expansion form of the minimizer into the optimization problem and consider the problem of finding the coefficients $\mathbf{c}_j$ for two different loss functions: the least-squares loss and the hinge loss.

3.1.1 Penalized Least Squares

Considering the least-squares loss function for regularization of vector-valued functions, the minimization problem becomes:

$$\operatorname*{argmin}_{h \in \mathcal{H}} \mathcal{J}(h) = \sum_{i=1}^{\ell} \|h(x_i) - \tilde{\mathbf{y}}_i\|_{\mathcal{F}_y}^2 + \lambda_1 \|h\|_{\mathcal{H}}^2 \ . \tag{2}$$

Theorem 4 (Micchelli and Pontil (2005)) *Let $\mathbf{c}_j \in \mathcal{F}_y, j = 1, \cdots, \ell$, be the coefficients of the expansion admitted by he minimizer $\hat{h}$ of the optimization problem in Equation (2). The vectors $\mathbf{c}_j \in \mathcal{F}_y$ satisfy the equations:*

$$\sum_{i=1}^{\ell} (\mathcal{K}_x(x_j, x_i) + \lambda_1 \delta_{ij})\mathbf{c}_i = \tilde{\mathbf{y}}_j \ ,$$

where δ is the Kronecker symbol: $\delta_{ii} = 1$ and $\forall j \neq i$, $\delta_{ij} = 0$.

3.1.2 Maximum Margin Regression

Szedmak et al. (2005) formulated a Support Vector Machine algorithm with vector output, called Maximum Margin Regression (MMR). The optimization problem of MMR in the supervised setting is the following:

$$\operatorname*{argmin}_{h} \mathcal{J}(h) = \sum_{i=1}^{\ell} \max(0, 1 - \langle \tilde{\mathbf{y}}_i, h(x_i) \rangle_{\mathcal{F}_y}) + \lambda_1 \|h\|_{\mathcal{H}}^2. \tag{3}$$

In Szedmak et al. (2005), the function h was modeled as: $h(x) = W\varphi_x(x) + b$, where φ_x is a feature map associated to a scalar-valued kernel. In this subsection, we extend this maximum margin based regression framework to the context of the vector-valued RKHS theory by searching h in the RKHS $\mathcal{H}$ associated to $\mathcal{K}_x$.

Similarly to SVM, the MMR problem (3) can be expressed according to a primal formulation that involves the optimization of $h \in \mathcal{H}$ and slack variables $\xi_i \in \mathbb{R}$, $i = 1, \ldots, \ell$, as well as its dual formulation which is expressed according to the Lagrangian parameters $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_\ell]^T \in \mathbb{R}^\ell$. The latter leads to solve a quadratic problem, for which efficient solvers exist. Both formulations are given below.

The primal form of the MMR optimization problem can be written as

$$\begin{aligned} \min_{h \in \mathcal{H}, \{\xi_i\} \in \mathbb{R}} \quad & \lambda_1 \|h\|_{\mathcal{H}}^2 + \sum_{i=1}^{\ell} \xi_i \\ \text{s.t.} \quad & \langle \tilde{\mathbf{y}}_i, h(x_i) \rangle_{\mathcal{F}_y} \geq 1 - \xi_i, i = 1, \ldots, \ell \\ & \xi_i \geq 0, i = 1, \ldots, \ell \end{aligned}$$

The Lagrangian of the above problem is given by:

$$\mathcal{L}_a(h, \boldsymbol{\xi}, \boldsymbol{\alpha}, \boldsymbol{\eta}) = \lambda_1 \|h\|_{\mathcal{H}}^2 + \sum_{i=1}^{\ell} \xi_i - \sum_{i=1}^{\ell} \alpha_i (\langle \mathcal{K}_x(\cdot, x_i) \tilde{\mathbf{y}}_i, h \rangle_{\mathcal{H}} - 1 + \xi_i) - \sum_{i=1}^{\ell} \eta_i \xi_i,$$

with α_i and η_i being Lagrange multipliers. By differentiating the Lagrangian with respect to ξ_i and h and setting the derivatives to zero, the dual form of the optimization problem can be expressed as:

$$\begin{aligned} \min_{\boldsymbol{\alpha} \in \mathbb{R}^\ell} \quad & \frac{1}{4\lambda_1} \sum_{i,j=1}^{\ell} \alpha_i \alpha_j \tilde{\mathbf{y}}_i^T \mathcal{K}_x(x_i, x_j) \tilde{\mathbf{y}}_j - \sum_{i=1}^{\ell} \alpha_i \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq 1, i = 1, \ldots, \ell \end{aligned}$$

and the solution $\hat{h}$ can be written as: $\hat{h}(\cdot) = \frac{1}{2\lambda_1} \sum_{j=1}^{\ell} \alpha_j \mathcal{K}_x(\cdot, x_j) \tilde{\mathbf{y}}_j$.

Note that, similarly to KDE, we retrieve the original MMR solution when using the following operator-valued kernel: $\mathcal{K}_x(x, x') = \kappa_x(x, x')\, I$.

3.2 Extension to Semi-Supervised Learning

In the case of real-valued functions, Belkin et al. (2006) have introduced a novel framework, called *manifold regularization*. This approach is based on the assumption that the data lie in a low-dimensional manifold. Belkin et al. (2006) have proved a representer theorem devoted to semi-supervised learning by adding a new regularization term which exploit the information of the geometric structure. This regularization term forces the target function h to be smooth with respect to the underlying manifold. In general, the geometry of this manifold is not known but it can be approximated by a graph. In this graph, nodes correspond to labeled and unlabeled data and edges reflect the local similarities between data in the input space. For example, this graph can be built using k-nearest neighbors. The representer theorem of Belkin et al. (2006) has been extended to the case of vector-valued functions in Brouard et al. (2011) and Minh and Sindhwani (2011). In the following, we present this theorem and derive the solutions for the least-squares loss function and maximum margin regression.

Let $\mathcal{L}$ be a convex loss function. Given a set of ℓ labeled examples $\{(x_i, \tilde{\mathbf{y}}_i)\}_{i=1}^{\ell} \subseteq \mathcal{X} \times \mathcal{F}_y$ and an additional set of n unlabeled examples $\{x_i\}_{i=\ell+1}^{\ell+n} \subseteq \mathcal{X}$, we consider the following optimization problem:

$$\operatorname*{argmin}_{h \in \mathcal{H}} \mathcal{J}(h) = \sum_{i=1}^{\ell} \mathcal{L}(h(x_i), \tilde{\mathbf{y}}_i) + \lambda_1 \|h\|_{\mathcal{H}}^2 + \lambda_2 \sum_{i,j=1}^{\ell+n} W_{ij} \|h(x_i) - h(x_j)\|_{\mathcal{F}_y}^2, \tag{4}$$

where λ_1, $\lambda_2 > 0$ are two regularization hyperparameters and W is the adjacency matrix of a graph built from labeled and unlabeled data. This matrix measures the similarity between objects in the input space. This optimization problem can be rewritten as:

$$\operatorname*{argmin}_{h \in \mathcal{H}} \mathcal{J}(h) = \sum_{i=1}^{\ell} \mathcal{L}(h(x_i), \tilde{\mathbf{y}}_i) + \lambda_1 \|h\|_{\mathcal{H}}^2 + 2\lambda_2 \sum_{i,j=1}^{\ell+n} L_{ij} \langle h(x_i), h(x_j) \rangle_{\mathcal{F}_y} ,$$

where L is the graph Laplacian given by $L = D - W$, and D is the diagonal matrix of general term $D_{ii} = \sum_{j=1}^{\ell+n} W_{ij}$. Instead of the graph Laplacian, other matrices, such as iterated Laplacians or diffusion kernels (Kondor and Lafferty, 2002), can also be used.

Theorem 5 (Brouard et al. (2011); Minh and Sindhwani (2011)) *The minimizer of the optimization problem in Equation* (4) *admits an expansion:*

$$\hat{h}(\cdot) = \sum_{j=1}^{\ell+n} \mathcal{K}_x(\cdot, x_j) \mathbf{c}_j ,$$

for some vectors $\mathbf{c}_j \in \mathcal{F}_y, j = 1, \cdots, \ell + n$.

This theorem extends the representer theorem proposed by Belkin et al. (2006) to vector-valued functions. Besides, it also extends Theorem 3 to the semi-supervised framework.

3.2.1 Semi-Supervised Penalized Least-Squares

Considering the least-squares cost, the optimization problem becomes:

$$\operatorname*{argmin}_{h \in \mathcal{H}} \mathcal{J}(h) = \sum_{i=1}^{\ell} \|h(x_i) - \tilde{\mathbf{y}}_i\|_{\mathcal{F}_y}^2 + \lambda_1 \|h\|_{\mathcal{H}}^2 + 2\lambda_2 \sum_{i,j=1}^{\ell+n} L_{ij} \langle h(x_i), h(x_j) \rangle_{\mathcal{F}_y}. \tag{5}$$

Theorem 6 (Brouard et al. (2011); Minh and Sindhwani (2011)) *The coefficients* $\mathbf{c}_j \in \mathcal{F}_y, j = 1, \cdots, \ell + n$ *of the expansion admitted by the minimizer* $\hat{h}$ *of the optimization problem* (5) *satisfy this equation:*

$$J_j \sum_{i=1}^{\ell+n} \mathcal{K}_x(x_j, x_i) \mathbf{c}_i + \lambda_1 \mathbf{c}_j + 2\lambda_2 \sum_{i=1}^{\ell+n} L_{ij} \sum_{m=1}^{\ell+n} \mathcal{K}_x(x_i, x_m) \mathbf{c}_m = J_j \, \tilde{\mathbf{y}}_j ,$$

where $J_j \in \mathcal{B}(\mathcal{F}_y)$ *is the identity operator if* $j \leq \ell$ *and the null operator if* $\ell < j \leq (\ell + n)$.

3.2.2 Semi-Supervised Maximum Margin Regression

The optimization problem in the semi-supervised case using the hinge loss is the following:

$$\operatorname*{argmin}_{h\in\mathcal{H}} \mathcal{J}(h) = \sum_{i=1}^{\ell} \max(0, 1 - \langle \tilde{\mathbf{y}}_i, h(x_i)\rangle_{\mathcal{F}_y}) + \lambda_1 \|h\|^2_{\mathcal{H}} + 2\lambda_2 \sum_{i,j=1}^{\ell+n} L_{ij}\langle h(x_i), h(x_j)\rangle_{\mathcal{F}_y}. \quad (6)$$

Theorem 7 *The solution of the optimization problem (6) is given by*

$$h(\cdot) = \frac{1}{2} B^{-1} \left(\sum_{i=1}^{\ell} \alpha_i \mathcal{K}_x(\cdot, x_i)\tilde{\mathbf{y}}_i \right),$$

where $B = \lambda_1 I + 2\lambda_2 \sum_{i,j=1}^{\ell+n} L_{ij}\mathcal{K}_x(\cdot, x_i)\mathcal{K}_x(x_j, \cdot)$ *is an operator from* $\mathcal{H}$ *to* $\mathcal{H}$*, and* $\boldsymbol{\alpha}$ *is the solution of*

$$\begin{array}{ll} \min\limits_{\boldsymbol{\alpha}\in\mathbb{R}^{\ell}} & \frac{1}{4}\sum\limits_{i,j=1}^{\ell} \alpha_i\alpha_j \langle \mathcal{K}_x(\cdot, x_i)\tilde{\mathbf{y}}_i, B^{-1}\mathcal{K}_x(\cdot, x_j)\tilde{\mathbf{y}}_j\rangle - \sum\limits_{i=1}^{\ell}\alpha_i \\ s.t. & 0 \le \alpha_i \le 1, i = 1, \ldots, \ell \end{array} \quad (7)$$

The proof of this theorem is detailed in the Appendix A.

3.3 Solutions when $\mathcal{F}_y = \mathbb{R}^d$

In this subsection we consider that the dimension of $\mathcal{F}_y$ is finite and equal to d. We first introduce the following notations:

- $\tilde{Y}_\ell = (\tilde{\mathbf{y}}_1, \ldots, \tilde{\mathbf{y}}_\ell)$ is a matrix of size $d \times \ell$,
- $C_\ell = (\mathbf{c}_1, \ldots, \mathbf{c}_\ell)$, $C_{\ell+n} = (\mathbf{c}_1, \ldots, \mathbf{c}_{\ell+n})$,
- $\mathbf{\Phi}_{x_\ell} = (\mathcal{K}_x(\cdot, x_1), \ldots, \mathcal{K}_x(\cdot, x_\ell))$, $\mathbf{\Phi}_{x_{\ell+n}} = (\mathcal{K}_x(\cdot, x_1), \ldots, \mathcal{K}_x(\cdot, x_{\ell+n}))$,
- $\mathbf{K}_{x_\ell}$ is a $\ell \times \ell$ block matrix, where each block is a $d \times d$ matrix. The (j,k)-*th* block of $\mathbf{K}_{x_\ell}$ is equal to $\mathcal{K}_x(x_j, x_k)$,
- $\mathbf{K}_{x_{\ell+n}}$ is a $(\ell+n)\times(\ell+n)$ block matrix such that the (j,k)-*th* block of $\mathbf{K}_{x_{\ell+n}}$ is equal to $\mathcal{K}_x(x_j, x_k)$,
- $I_{\ell d}$ and $I_{(\ell+n)d}$ are identity matrices of size $(\ell d)\times(\ell d)$ and $(\ell+n)d \times (\ell+n)d$,
- $J = (I_\ell, 0)$ is a $\ell \times (\ell+n)$ matrix that contains an identity matrix of size $\ell \times \ell$ on the left hand side and a zero matrix of size $\ell \times n$ on the right hand side,
- $\otimes$ denotes the Kronecker product and vec(A) denotes the vectorization of a matrix A, formed by stacking the columns of A into a single column vector.

In the supervised setting, the solutions for the least-squares loss and MMR can be rewritten as:

$$h_{ridge}(\cdot) = \mathbf{\Phi}_{x_\ell}(\lambda_1 I_{\ell d} + \mathbf{K}_{x_\ell})^{-1} \operatorname{vec}(\tilde{Y}_\ell),$$
$$h_{mmr}(\cdot) = \frac{1}{2\lambda_1}\mathbf{\Phi}_{x_\ell} \operatorname{vec}(\tilde{Y}_\ell \operatorname{diag}(\boldsymbol{\alpha})).$$

In the semi-supervised setting, these solutions become:

$$\begin{aligned} h_{ridge}(\cdot) &= \mathbf{\Phi}_{x_{\ell+n}} \left(\lambda_1 I_{(\ell+n)d} + ((J^T J + 2\lambda_2 L) \otimes I_d)\mathbf{K}_{x_{\ell+n}}\right)^{-1} \operatorname{vec}(\tilde{Y}_\ell J), \\ h_{mmr}(\cdot) &= \mathbf{\Phi}_{x_{\ell+n}} \left(2\lambda_1 I_{(\ell+n)d} + 4\lambda_2 (L \otimes I_d)\mathbf{K}_{x_{\ell+n}}\right)^{-1} \operatorname{vec}(\tilde{Y}_\ell \operatorname{diag}(\boldsymbol{\alpha}) J). \end{aligned} \tag{8}$$

For MMR, the vector $\boldsymbol{\alpha}$ is obtained by solving the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{\alpha}\in\mathbb{R}^\ell} \quad & \frac{1}{4} \operatorname{vec}\left(\tilde{Y}_\ell \operatorname{diag}(\boldsymbol{\alpha}) J\right)^T \\ & \left(\lambda_1 I_{(\ell+n)d} + 2\lambda_2 \mathbf{K}_{x_{\ell+n}}(L \otimes I_d)\right)^{-1} \mathbf{K}_{x_{\ell+n}} \operatorname{vec}\left(\tilde{Y}_\ell \operatorname{diag}\left(\boldsymbol{\alpha}\right) J\right) - \boldsymbol{\alpha}^T \mathbf{1} \\ \text{s.t.} \quad & 0 \leq \alpha_i \leq 1, i = 1, \ldots, \ell. \end{aligned} \tag{9}$$

3.4 Models for General Decomposable Kernel

In the remainder of this section we propose to derive models based on on a simple but powerful family of operator-values kernels (OVK) based on scalar-valued kernels, called *decomposable kernels* or *separable kernels* (Álvarez et al., 2012; Baldassarre et al., 2012). They correspond to the simplest generalization of scalar kernels to operator-valued kernel. Decomposable kernels were first defined to deal with multi-task regression (Evgeniou et al., 2005; Micchelli and Pontil, 2005) and later, with structured multi-class classification (Dinuzzo et al., 2011). Other kernels (Caponnetto et al., 2008; Álvarez et al., 2012) have also been proposed: for instance, Lim et al. (2013) introduced a Hadamard kernel based on the Hadamard product of decomposable kernels and transformable kernels to deal with nonlinear vector autoregressive models. Caponnetto et al. (2008) proved that they are universal, meaning that an operator-valued regressor built on them is a universal approximator in $\mathcal{F}_y$.

Proposition 8 *The class of decomposable operator-valued kernels is composed of kernels of the form:*

$$\begin{aligned} \mathcal{K}_x : \quad & \mathcal{X} \times \mathcal{X} \to \mathcal{B}(\mathcal{F}_y) \\ & (x, x') \mapsto \kappa_x(x, x')\, A \end{aligned}$$

where $\kappa_x : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a scalar-valued input kernel and $A \in \mathcal{B}(\mathcal{F}_y)$ is a positive semi-definite operator.

In the multi-task learning framework, $\mathcal{F}_y = \mathbb{R}^d$ is a finite dimensional output space and the matrix A encodes the existing relations among the d different tasks. This matrix can be estimated from labeled data or being learned simultaneously with the matrix C (Dinuzzo et al., 2011).

3.4.1 Penalized Least-Squares Regression

In this section, we will use the following notations: $\mathcal{F}_x$ and the function $\varphi_x : \mathcal{X} \to \mathcal{F}_x$ correspond respectively to the feature space and the feature map associated to the input scalar kernel κ_x. We note $\Phi_{x_\ell} = (\varphi_x(x_1), \ldots, \varphi_x(x_\ell))$ the matrix of dimension $dim(\mathcal{F}_x) \times \ell$, and $\Phi_{x_{\ell+n}} = (\varphi_x(x_1), \ldots, \varphi_x(x_{\ell+n}))$. Let $K_{x_\ell} = \Phi_{x_\ell}^T \Phi_{x_\ell}$ and $K_{x_{\ell+n}} = \Phi_{x_{\ell+n}}^T \Phi_{x_{\ell+n}}$ be respectively the Gram matrices of κ_x over the sets $\mathcal{X}_\ell$ and $\mathcal{X}_{\ell+n}$. I_ℓ denotes the identity matrix of size ℓ. We assume that $\mathcal{F}_y = \mathbb{R}^d$.

The minimizer h of the optimization problem for the penalized least-squares cost in the supervised setting (2) using a decomposable OVK can be expressed as:

$$\begin{aligned} \forall x \in \mathcal{X}, h(x) &= A \sum_{i=1}^{\ell} \kappa_x(x, x_i)\mathbf{c}_i = AC_\ell \Phi_{x_\ell}^T \varphi_x(x) = (\varphi_x(x)^T \Phi_{x_\ell} \otimes A)\,\mathrm{vec}(C_\ell) \\ &= (\varphi_x(x)^T \Phi_{x_\ell} \otimes A)\,(\lambda_1 I_{\ell d} + K_{x_\ell} \otimes A)^{-1}\,\mathrm{vec}(\tilde{Y}_\ell). \end{aligned} \tag{10}$$

Therefore, the computation of the solution h requires to compute the inverse of a matrix of size $\ell d \times \ell d$. A being a real symmetric matrix, we can write an eigen-decomposition of A:

$$A = E\Gamma E^T = \sum_{i=1}^{d} \gamma_i \mathbf{e}_i \mathbf{e}_i^T,$$

where $E = (\mathbf{e}_1, \ldots, \mathbf{e}_d)$ is a $d \times d$ matrix and Γ is a diagonal matrix containing the eigenvalues of A: $\Gamma = \mathrm{diag}\,(\gamma_1, \ldots, \gamma_d)$. Using the eigen-decomposition of A, we can prove that the solution $\hat{h}(x)$ can be obtained by solving d independent problems:

Proposition 9 *The minimizer of the optimization problem for the supervised penalized least squares cost* (2) *in the case of a decomposable operator-valued kernel can be expressed as:*

$$\forall x \in \mathcal{X}, h_{ridge}(x) = \sum_{j=1}^{d} \gamma_j \mathbf{e}_j \mathbf{e}_j^T \tilde{Y}_\ell (\lambda_1 I_\ell + \gamma_j K_{x_\ell})^{-1} \Phi_{x_\ell}^T \varphi_x(x), \tag{11}$$

and in the semi-supervised setting (5)*, it writes as*

$$\forall x \in \mathcal{X}, h_{ridge}(x) = \sum_{j=1}^{d} \gamma_j \mathbf{e}_j \mathbf{e}_j^T \tilde{Y}_\ell J \left(\lambda_1 I_{\ell+n} + \gamma_j K_{x_{\ell+n}} (J^T J + 2\lambda_2 L)\right)^{-1} \Phi_{x_{\ell+n}}^T \varphi_x(x).$$

We observe that, in the supervised setting, the complexity to solve Equation (10) is equal to $O((\ell d)^3)$, while the complexity for solving Equation (11) is $O(d^3 + \ell^3)$.

3.4.2 Maximum Margin regression

Proposition 10 *Given $\mathcal{K}_x(x, x') = \kappa_x(x, x')\,A$, the dual formulation of the MMR optimization problem* (7) *in the supervised setting becomes:*

$$\begin{aligned} \min_{\boldsymbol{\alpha} \in \mathbb{R}^\ell} \quad & \frac{1}{4\lambda_1} \boldsymbol{\alpha}^T (\tilde{Y}_\ell^T A \tilde{Y}_\ell \circ K_{x_\ell}) \boldsymbol{\alpha} - \boldsymbol{\alpha}^T \mathbf{1} \\ s.t. \quad & 0 \leq \alpha_i \leq 1, i = 1, \ldots, \ell \end{aligned}$$

and the solution is given by: $h_{mmr}(\cdot) = \frac{1}{2\lambda_1} A\tilde{Y}_\ell \, diag\,(\boldsymbol{\alpha})\Phi_{x_\ell}^T$.

In the semi-supervised MMR minimization problem (6)*, it writes as:*

$$\begin{aligned} \min_{\boldsymbol{\alpha}\in\mathbb{R}^\ell} \quad & \frac{1}{2}\boldsymbol{\alpha}^T\Big(\sum_{i=1}^{d}\gamma_i\tilde{Y}_\ell^T\mathbf{e}_i\mathbf{e}_i^T\tilde{Y}_\ell \circ J(2\lambda_1 I_{\ell+n} + 4\lambda_2\gamma_i K_{x_{\ell+n}}L)^{-1}K_{x_{\ell+n}}J^T\Big)\boldsymbol{\alpha} - \boldsymbol{\alpha}^T\mathbf{1} \\ s.t. \quad & 0 \le \alpha_i \le 1, i = 1,\dots,\ell \end{aligned}$$

The corresponding solution is:

$$h_{mmr}(\cdot) = \frac{1}{2}\sum_{j=1}^{d}\gamma_j\mathbf{e}_j\mathbf{e}_j^T\tilde{Y}_\ell \,\mathrm{diag}\,(\boldsymbol{\alpha})J\left(\lambda_1 I_{\ell+n} + 2\gamma_j\lambda_2 K_{x_{\ell+n}}L\right)^{-1}\Phi_{x_{\ell+n}}^T.$$

Proofs of propositions (9) and (10) are given in Appendix A.

4 Model Selection

Real-valued kernel-based models enjoy a closed-form solution for the estimate of the leave-one-out criterion in the case of kernel ridge regression (Golub et al., 1979; Rifkin and Lippert, 2007). In order to select the hyperparameters of OVK-based models with a least-squares loss presented below, we develop a closed-form solution for the leave-one-out estimate of the sum of square errors. This solution extends Allen's predicted residual sum of squares (PRESS) statistics (Allen, 1974) to vector-valued functions. This result was first presented in french in the phd thesis of Brouard (2013) in the case of decomposable kernels. In the following, we will use the notations used by Rifkin and Lippert (2007). We assume in this section that the dimension of $\mathcal{F}_y$ is finite.

Let $\mathcal{S} = \{(x_1, \tilde{\mathbf{y}}_1), \dots, (x_\ell, \tilde{\mathbf{y}}_\ell)\}$ be the training set composed of ℓ labeled points. We define $\mathcal{S}^i$, $1 \le i \le \ell$, as the labeled data set with the i^{th} point removed:

$$\mathcal{S}^i = \{(x_1, \tilde{\mathbf{y}}_1), \dots, (x_{i-1}, \tilde{\mathbf{y}}_{i-1}), (x_{i+1}, \tilde{\mathbf{y}}_{i+1}), \dots, (x_\ell, \tilde{\mathbf{y}}_\ell)\}.$$

In this section, $h_{\mathcal{S}}$ denotes the function obtained when the regression problem is trained on the entire training set $\mathcal{S}$ and we note $h_{\mathcal{S}^i}(x_i)$ the i^{th} leave-one-out value, that is the value at the point x_i of the function obtained when the training set is $\mathcal{S}^i$. The PRESS criterion corresponds to the sum of the ℓ leave-one-out square errors:

$$PRESS = \sum_{i=1}^{\ell}\|\tilde{\mathbf{y}}_i - h_{\mathcal{S}^i}(x_i)\|_{\tilde{\mathcal{Y}}}^2.$$

As for scalar-valued functions, we show that it is possible to compute this criterion without evaluating explicitly $h_{\mathcal{S}^i}(x_i)$ for $i = 1,\dots,\ell$ and for each value of the grid of parameters.

Assuming we know $h_{\mathcal{S}^i}$, we define the matrix $\tilde{Y}_\ell^i = (\tilde{\mathbf{y}}_1^i, \dots, \tilde{\mathbf{y}}_\ell^i)$, where the vector $\tilde{\mathbf{y}}_j^i$ is given by:

$$\tilde{\mathbf{y}}_j^i = \begin{cases} \tilde{\mathbf{y}}_j & \text{if } j \ne i \\ h_{\mathcal{S}^i}(x_i) & \text{if } j = i \end{cases}$$

In the following, we show that when using $\tilde{Y}_\ell^i$ instead of $\tilde{Y}_\ell$, the optimal solution corresponds to $h_{\mathcal{S}^i}$:

$$\begin{aligned}
&\sum_{j=1}^{\ell} \|\tilde{\mathbf{y}}_j^i - h_{\mathcal{S}}(x_j)\|_{\tilde{\mathcal{Y}}}^2 + \lambda_1 \|h_{\mathcal{S}}\|_{\mathcal{H}}^2 + \lambda_2 \sum_{j,k=1}^{\ell+n} W_{jk} \|h_{\mathcal{S}}(x_j) - h_{\mathcal{S}}(x_k)\|_{\tilde{\mathcal{Y}}}^2 \\
&\qquad \geq \sum_{j \neq i} \|\tilde{\mathbf{y}}_j^i - h_{\mathcal{S}}(x_j)\|_{\tilde{\mathcal{Y}}}^2 + \lambda_1 \|h_{\mathcal{S}}\|_{\mathcal{H}}^2 + \lambda_2 \sum_{j,k=1}^{\ell+n} W_{jk} \|h_{\mathcal{S}}(x_j) - h_{\mathcal{S}}(x_k)\|_{\tilde{\mathcal{Y}}}^2 \\
&\qquad \geq \sum_{j \neq i} \|\tilde{\mathbf{y}}_j^i - h_{\mathcal{S}^i}(x_j)\|_{\tilde{\mathcal{Y}}}^2 + \lambda_1 \|h_{\mathcal{S}^i}\|_{\mathcal{H}}^2 + \lambda_2 \sum_{j,k=1}^{\ell+n} W_{jk} \|h_{\mathcal{S}^i}(x_j) - h_{\mathcal{S}^i}(x_k)\|_{\tilde{\mathcal{Y}}}^2 \\
&\qquad \geq \sum_{j=1}^{\ell} \|\tilde{\mathbf{y}}_j^i - h_{\mathcal{S}^i}(x_j)\|_{\tilde{\mathcal{Y}}}^2 + \lambda_1 \|h_{\mathcal{S}^i}\|_{\mathcal{H}}^2 + \lambda_2 \sum_{j,k=1}^{\ell+n} W_{jk} \|h_{\mathcal{S}^i}(x_j) - h_{\mathcal{S}^i}(x_k)\|_{\tilde{\mathcal{Y}}}^2.
\end{aligned}$$

The second inequality comes from the fact that $h_{\mathcal{S}^i}$ is defined as the minimizer of the optimization problem when the i^{th} point is removed from the training set. As $h_{\mathcal{S}^i}$ is the optimal solution when $\tilde{Y}_\ell$ is replaced with $\tilde{Y}_\ell^i$, it can be written as:

$$\forall i = 1, \ldots, \ell, \; h_{\mathcal{S}^i}(x_i) = \phi_x(x_i)^T \mathbf{\Phi}_{x_{\ell+n}} B \operatorname{vec}(\tilde{Y}_\ell^i) = (KB)_{i,\cdot} \operatorname{vec}(\tilde{Y}_\ell^i),$$

where $K = \mathbf{K}_{x_{\ell \times (\ell+n)}}$ is the input gram matrix between the sets $\mathcal{X}_\ell$ and $\mathcal{X}_{\ell+n}$ and $B = (\lambda_1 I_{(\ell+n)d} + ((J^T J + 2\lambda_2 L) \otimes I_d)\mathbf{K}_{x_{\ell+n}})^{-1}(J^T \otimes I_d)$. $(KB)_{i,\cdot}$ corresponds to the i^{th} row of the matrix KB and $(KB)_{i,j}$ is the value of the matrix corresponding to the row i and the column j.

We can then derive an expression of $h_{\mathcal{S}^i}$ by computing the difference between $h_{\mathcal{S}^i}(x_i)$ and $h_{\mathcal{S}}(x_i)$:

$$\begin{aligned}
h_{\mathcal{S}^i}(x_i) - h_{\mathcal{S}}(x_i) &= (KB)_{i,\cdot} \operatorname{vec}(\tilde{Y}_\ell^i - \tilde{Y}_\ell) \\
&= \sum_{k=1}^{\ell} (KB)_{i,k} (\tilde{\mathbf{y}}_k^i - \tilde{\mathbf{y}}_k) \\
&= (KB)_{i,i} (h_{\mathcal{S}^i}(x_i) - \tilde{\mathbf{y}}_i),
\end{aligned}$$

which leads to

$$\begin{aligned}
&(I_d - (KB)_{i,i})\, h_{\mathcal{S}^i}(x_i) = h_{\mathcal{S}}(x_i) - (KB)_{i,i} \tilde{\mathbf{y}}_i \\
&\Rightarrow (I_d - (KB)_{i,i})\, h_{\mathcal{S}^i}(x_i) = (KB)_{i,\cdot} \operatorname{vec}(\tilde{Y}_\ell) - (KB)_{i,i} \tilde{\mathbf{y}}_i \\
&\Rightarrow h_{\mathcal{S}^i}(x_i) = (I_d - (KB)_{i,i})^{-1} \left((KB)_{i,\cdot} \operatorname{vec}(\tilde{Y}_\ell) - (KB)_{i,i} \tilde{\mathbf{y}}_i \right).
\end{aligned}$$

Let $L_{oo} = (h_{\mathcal{S}^1}(x_1), \ldots, h_{\mathcal{S}^\ell}(x_\ell))$ be the matrix containing the leave-one-out vector values over the training set. The equation above can be rewritten as:

$$\operatorname{vec}(L_{oo}) = (I_{\ell d} - \operatorname{diag}_b(KB))^{-1} (KB - \operatorname{diag}_b(KB)) \operatorname{vec}(\tilde{Y}_\ell),$$

where diag_b corresponds to the block diagonal of a matrix.

The Allen's PRESS statistic can be expressed as:

$$\begin{aligned} PRESS &= \| \operatorname{vec}(\tilde{Y}_\ell) - \operatorname{vec}(L_{oo}) \|^2 \\ &= \| (I_{\ell d} - \operatorname{diag}_b(KB))^{-1} (I_{\ell d} - \operatorname{diag}_b(KB) - KB + \operatorname{diag}_b(KB)) \operatorname{vec}(\tilde{Y}_\ell) \|^2 \\ &= \| (I_{\ell d} - \operatorname{diag}_b(KB))^{-1} (I_{\ell d} - KB) \operatorname{vec}(\tilde{Y}_\ell) \|^2. \end{aligned}$$

This closed-form expression allows to evaluate the PRESS criterion without having to solve ℓ problems involving the inversion of a matrix of size $(\ell + n - 1)d$.

5 Input Output Kernel Regression

We now have all the needed tools to approximate vector-valued functions. In this section, we go back to Input Output Kernel Regression and consider that $\mathcal{F}_y$ is the feature space associated to some output kernel $\kappa_y : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$. Several feature spaces can be defined, including the unique RKHS associated to the kernel κ_y. This choice has direct consequences on the choice of the input operator-valued kernel $\mathcal{K}_x$. Depending on the application, we might be interested for instance on choosing $\mathcal{F}_y$ as a functional space to get integral operators or as the finite-dimensional euclidean space $\mathbb{R}^d$ to get matrices. It is important to notice that this reflects a radically new approach in machine learning where we usually focus on the choice of the input feature space and do not discuss a lot the output space. Moreover, the choice of a given triplet $(\kappa_y, \mathcal{F}_y, \mathcal{K}_x)$ has a great impact of the learning task both in terms of complexity in time and potentially of performance. In the following, we explain how Input Output Kernel Regression can be used to solve link prediction and multi-task problems.

5.1 Link Prediction

Link prediction is a challenging machine learning problem that has been defined recently in social networks as well as biological networks. Let us formulate this problem using the previous notations: $\mathcal{X} = \mathcal{Y} = \mathcal{U}$ is the set of candidate nodes we are interested in. We want to estimate some relation between these nodes, for example a social relationship between persons or some physical interaction between molecules. During the training phase we are given $\mathcal{G}_\ell = (\mathcal{U}_\ell, A_\ell)$, a non oriented graph defined by the subset $\mathcal{U}_\ell \subseteq \mathcal{U}$ and the adjacency matrix A_ℓ of size $\ell \times \ell$. Supervised link prediction is usually addressed by learning a binary pairwise classifier $f : \mathcal{U} \times \mathcal{U} \to \{0, 1\}$ that predicts if there exists a link between two objects or not, from the training information $\mathcal{G}_\ell$. One way to solve this learning task is to built a pairwise classifier. However, the link prediction problem can also be formalized as an output kernel regression task (Geurts et al., 2007a; Brouard et al., 2011).

The OKR framework for link prediction is based on the assumption that an approximation of the output kernel κ_y will provide valuable information about the proximity of the objects of $\mathcal{U}$ as nodes in the unknown graph defined on $\mathcal{U}$. Given that assumption, a classifier f_θ is defined from the approximation $\widehat{\kappa_y}$ by thresholding its output values:

$$f_\theta(u, u') = \operatorname{sgn}(\widehat{\kappa_y}(u, u') - \theta).$$

An approximation of the target output kernel κ_y is built from the scalar product between the outputs of a single variable function $h : \mathcal{U} \to \mathcal{F}_y$: $\widehat{\kappa_y}(u, u') = \langle h(u), h(u') \rangle_{\mathcal{F}_y}$. Using the kernel trick in the output space therefore allows to reduce the problem of learning a pairwise classifier to the problem of learning a single variable function with output values in a Hilbert space (the output feature space $\mathcal{F}_y$).

In the case of IOKR, the function h is learnt in an appropriate RKHS by using the operator-valued kernel regression approach presented in Section 3. In the following, we describe the output kernel and the input operator-valued kernel that we propose to use for solving the link prediction problem with IOKR.

Regarding the output kernel, we do not have a kernel κ_y defined on $\mathcal{U} \times \mathcal{U}$ in the link prediction problem but we can define a Gram matrix K_{y_ℓ} defined on the training set $\mathcal{U}_\ell$. Here, we define the output Gram matrix K_{y_ℓ} from the known adjacency matrix A_ℓ of the training graph such that it encodes the proximities in the graph between the labeled nodes. For instance, we can choose the diffusion kernel matrix (Kondor and Lafferty, 2002), which is defined as:

$$K_{y_\ell} = \exp(-\beta L_{Y_\ell}),$$

where $L_{Y_\ell} = D_\ell - A_\ell$ is the graph Laplacian, with D_ℓ the diagonal matrix of degrees.

We assume that there exists a kernel $\kappa_y : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$, such that:

$$\forall i, j \in \{1, \ldots, \ell\},\ \kappa_y(u_i, u_j) = (K_{y_\ell})_{i,j}.$$

The feature space $\mathcal{F}_y$ is assumed to be the RKHS defined by κ_y.

Regarding the operator-valued kernel, we consider here the identity decomposable kernel:

$$\forall (u, u') \in \mathcal{U} \times \mathcal{U},\ \mathcal{K}_x(u, u') = \kappa_x(u, u')I.$$

We underline that even if this kernel may seem simple, we must be aware that in this task, we do not have the explicit expressions of outputs $\varphi_y(u)$ and prediction in $\mathcal{F}_y$ is not the final target. Therefore this operator-valued kernel allows us to work properly with output Gram matrix values.

Of particular interest for us is the expression of the scalar product which is the only one we need for link prediction. When using the identity decomposable kernel, the approximation of the output kernel can be written as follows:

$$\widehat{\kappa_y}(u, u') = \langle \hat{h}(u), \hat{h}(u') \rangle_{\mathcal{F}_y} = \varphi_x(u)^T B^T K_{y_\ell} B \varphi_x(u'),$$

where B is a matrix of size $\ell \times dim(\mathcal{F}_x)$ that depends of the loss function and the learning setting used (see Table 2). We can notice that we do not need to know the explicit expressions of outputs $\varphi_y(u)$ to compute this scalar product. Besides, this formulation shows that the expression of the scalar product $\varphi_y(u)^T \varphi_y(u')$ is approximated by a modified scalar product between inputs $\varphi_x(u)$ and $\varphi_x(u')$.

5.2 Multi-Task Learning

In multi-task learning problem, it may happen that the tasks are not disjoint and are characterized by a relationship such as inclusion or similarity. Examples of multi-task

$B =$	Supervised learning	Semi-supervised learning
Ridge	$(\lambda_1 I_\ell + K_{x_\ell})^{-1}\Phi_{x_\ell}^T$	$J(\lambda_1 I_{\ell+n} + K_{x_{\ell+n}}(J^T J + 2\lambda_2 L))^{-1}\Phi_{x_{\ell+n}}^T$
MMR	$\frac{1}{2\lambda_1}\operatorname{diag}(\boldsymbol{\alpha})\Phi_{x_\ell}^T$	$\frac{1}{2}\operatorname{diag}(\boldsymbol{\alpha})J(\lambda_1 I_{\ell+n} + 2\lambda_2 K_{x_{\ell+n}} L)^{-1}\Phi_{x_{\ell+n}}^T$

Table 2: Matrix B corresponding to the different settings and loss functions for the models obtained when using the identity decomposable kernel. These models write as: $\forall u \in \mathcal{U}, h(u) = \Phi_{y_\ell} B \varphi_x(u)$.

learning problems can be found in document categorization as well as in protein functional annotation prediction. Dependencies among target variables can also be encountered in the case of multiple regression. We consider here d tasks having the same input and output domains. $\mathcal{Y} = \mathcal{F}_y = \mathbb{R}^d$ is a finite dimensional output space.

We compared three models to solve this structured regression task:

- Model 0: $\kappa_y(\mathbf{y}, \mathbf{y}') = \mathbf{y}^T\mathbf{y}'$, with the identity kernel $\mathcal{K}_x(x, x') = \kappa_x(x, x')\, I$,
- Model 1: $\kappa_y(\mathbf{y}, \mathbf{y}') = \mathbf{y}^T A_1 \mathbf{y}'$, with the identity kernel $\mathcal{K}_x(x, x') = \kappa_x(x, x')\, I$,
- Model 2: $\kappa_y(\mathbf{y}, \mathbf{y}') = \mathbf{y}^T\mathbf{y}'$, with the decomposable kernel $\mathcal{K}_x(x, x') = \kappa_x(x, x')\, A_2$.

In the first case, the different tasks are learned independently :

$$\forall x \in \mathcal{X}, \hat{h}_0(x) = Y_\ell J \left(\lambda_1 I_{\ell+n} + K_{x_{\ell+n}}(J^T J + 2\lambda_2 L)\right)^{-1} \Phi_{x_{\ell+n}}^T \varphi_x(x),$$

while in the other cases, the tasks relatedness is taken into account :

$$\forall x \in \mathcal{X},\ \hat{h}_1(x) = \sqrt{A_1} Y_\ell J \left(\lambda_1 I_{\ell+n} + K_{x_{\ell+n}}(J^T J + 2\lambda_2 L)\right)^{-1} \Phi_{x_{\ell+n}}^T \varphi_x(x),$$

$$\forall x \in \mathcal{X},\ \hat{h}_2(x) = \sum_{j=1}^{d} \gamma_j \mathbf{e}_j \mathbf{e}_j^T Y_\ell J (\lambda_1 I_{\ell+n} + \gamma_j K_{x_{\ell+n}}(J^T J + 2\lambda_2 L))^{-1} \Phi_{x_{\ell+n}}^T \varphi_x(x),$$

where γ_j and $\mathbf{e}_j$ are the eigenvalues and eigenvectors of A_2.

We consider a matrix M of size $d \times d$ that encodes the relations existing between the different tasks. This matrix can be considered as the adjacency matrix of a graph between tasks. We note L_M the graph laplacian associated to this matrix. The matrices A_1 and A_2 are defined as follow:

$$A_1 = \mu M + (1 - \mu) I_d,$$
$$A_2 = (\mu L_M + (1 - \mu) I_d)^{-1},$$

where μ is a parameter in $[0, 1]$.

The matrix A_2 was proposed by Evgeniou et al. (2005) and Sheldon (2008) for multi-task learning. Given a decomposable kernel defined with this matrix A_2, the norm of the function h_2 in $\mathcal{H}$ can be written as:

$$\|h_2\|_{\mathcal{H}}^2 = \frac{\mu}{2} \sum_{i,j=1}^{d} M_{ij} \|h_2^{(i)} - h_2^{(j)}\|^2 + (1 - \mu) \sum_{i=1}^{d} \|h_2^{(i)}\|^2,$$

where $h_2 = [h_2^{(1)}, \ldots, h_2^{(d)}]$ and $h_2^{(i)}$ corresponds to the i-th component of h. This regularization term forces two tasks $h_2^{(i)}$ and $h_2^{(j)}$ to be close to each other when the similarity value M_{ij} is high and conversely.

6 Numerical Experiments

In this section, we present the performances obtained with the IOKR approach on two different problems: link prediction and multi-task regression. In these experiments, we examine the effect of the smoothness constraint through the variation of its related hyperparameter λ_2, using supervised method as a baseline. We evaluate the method in the transductive setting, that is we assume that all the examples (labeled and unlabeled) are known at the beginning of the learning phase and the goal is to predict the correct outputs for the unlabeled examples.

6.1 Link Prediction

For the link prediction problem, we considered experiments on three datasets: a collection of synthetic networks, a co-authorship network and a protein-protein interaction (PPI) network.

6.1.1 Protocol

For different percentages of labeled nodes, we randomly selected a subsample of nodes as labeled nodes and used the remaining ones as unlabeled nodes. Labeled interactions correspond to interactions between two labeled nodes. This means that when 10% of labeled nodes are selected, it corresponds to only 1% of labeled interactions. The performances were evaluated by averaging the areas under the ROC curve and the precision-recall curve (denoted AUC-ROC and AUC-PR) over ten random choices of the training set. A gaussian kernel was used for the scalar input kernel κ_x. Its corresponding bandwidth σ was selected by a leave-one-out cross-validation procedure on the training set to maximize the AUC-ROC, jointly with the hyperparameter λ_1. In the case of the least-squares loss function, we used the leave-one-out estimates approach introduced in Section 4. The output kernel used is a diffusion kernel of parameter β. Another diffusion kernel of parameter β_2 was also used for the smoothing penalty: $\exp(-\beta_2 L) = \sum_{i=0}^{\infty} \frac{(-\beta_2 L)^i}{i!}$. Preliminary runs have shown that the values of β and β_2 have a limited influence on the performances, we then have set both parameters to 1. Finally we set W to $K_{x_{\ell+n}}$.

6.1.2 Synthetic Networks

We first illustrate our method on synthetic networks where the input kernel was chosen as a very good approximation of the output kernel. In these experiments we wanted to measure the improvement brought by the semi-supervised method in extreme cases, i.e. when the percentage of labeled nodes is very low.

The output networks were obtained by sampling random graphs containing 700 nodes from a Erdős-Renyi law with different graph densities. The graph density corresponds to

the probability of presence of edges in the graph. In this experiment we chose three densities that are representative of real network densities: 0.007, 0.01 and 0.02. For each network, we used the diffusion kernel as output kernel and chose the diffusion parameter such that it maximizes an information criterion. To built an input kernel corresponding to a good approximation of the output kernel, we applied kernel PCA on the output kernel and used the components capturing 95% of the variance as input vectors. We then build a gaussian kernel based on these inputs.

Figures 3 and 4 report respectively the averaged values and standard deviations for the AUC-ROC and AUC-PR obtained for different network densities and different percentages of labeled nodes. We observe that IOKR-ridge outperforms IOKR-margin in the supervised and in the semi-supervised cases. This improvement is particularly significant for AUC-PR, especially when the network density is strong and the percentage of labeled data is high. It is thus very significant for 10% and 20% of labeled data. In the supervised case, this observation can be explained by the difference between the complexities of the models. The solutions obtained in the supervised case for both models are written in the form $\hat{h}(u) = C_\ell \Phi_{x_\ell}^T \varphi_x(u)$. For the IOKR-ridge model, $C_\ell = \Phi_{y_\ell}(\lambda_1 I_\ell + K_{x_\ell})^{-1}$, while for the IOKR-margin model we have: $C_\ell = \frac{1}{2\lambda_1}\Phi_{y_\ell}\,\mathrm{diag}\,(\boldsymbol{\alpha})$. The synthetic networks may require a more complex predictor.

We observe an improvement of the performances in terms of AUC-ROC and AUC-PR for both approaches in the semi-supervised setting compared to the supervised setting. This improvement is more significant for IOKR-margin. This can be explained by the fact that the IOKR-margin models obtained in the supervised and in the semi-supervised cases do not have the same complexity. The solution in the supervised case writes as $\hat{h}(u) = C_\ell \Phi_{x_\ell}^T \varphi_x(u)$ with $C_\ell = \frac{1}{2\lambda_1}\Phi_{y_\ell}\,\mathrm{diag}\,(\boldsymbol{\alpha})$, while in the semi-supervised case, the solution can be written as $\hat{h}(u) = C_{\ell+n}\Phi_{x_{\ell+n}}^T\,\varphi_x(u)$, where $C_{\ell+n}$ is a much richer matrix: $C_{\ell+n} = \Phi_{y_\ell}\,\mathrm{diag}\,(\boldsymbol{\alpha})J(2\lambda_1 I_{\ell+n} + 4\lambda_2 K_{x_{\ell+n}} L)^{-1}$. For IOKR-ridge, the improvement of the performance is only observed for low percentages of labeled data. We can therefore make the assumption that for this model, using unlabeled data increases the AUCs for low percentages of labeled data. But when enough information can be found in the labeled data, semi-supervised learning does not improve the performance.

Based on these results, we can also formulate the assumption that link prediction is harder in the case of dense networks.

6.1.3 NIPS Co-authorship Network

We applied our method on a co-authorship network containing information on publications of the NIPS conferences between 1988 to 2003 (Globerson et al., 2007). In this network, vertices represent authors and an edge connects two authors if they have at least one NIPS publication in common. Among the 2865 authors, we considered the ones with at least two links in the co-authorship network in order to have a significant density and trying to keep close to the original data. We therefore focused on a network containing 2026 authors with an empirical link density of 0.002. Each author was described by a vector of 14036 values, corresponding to the frequency with which he uses each given word in his papers.

Figure 5 reports the averaged AUC-ROC and AUC-PR obtained on the NIPS co-authorship network for different values of λ_2 and different percentages of labeled nodes. As

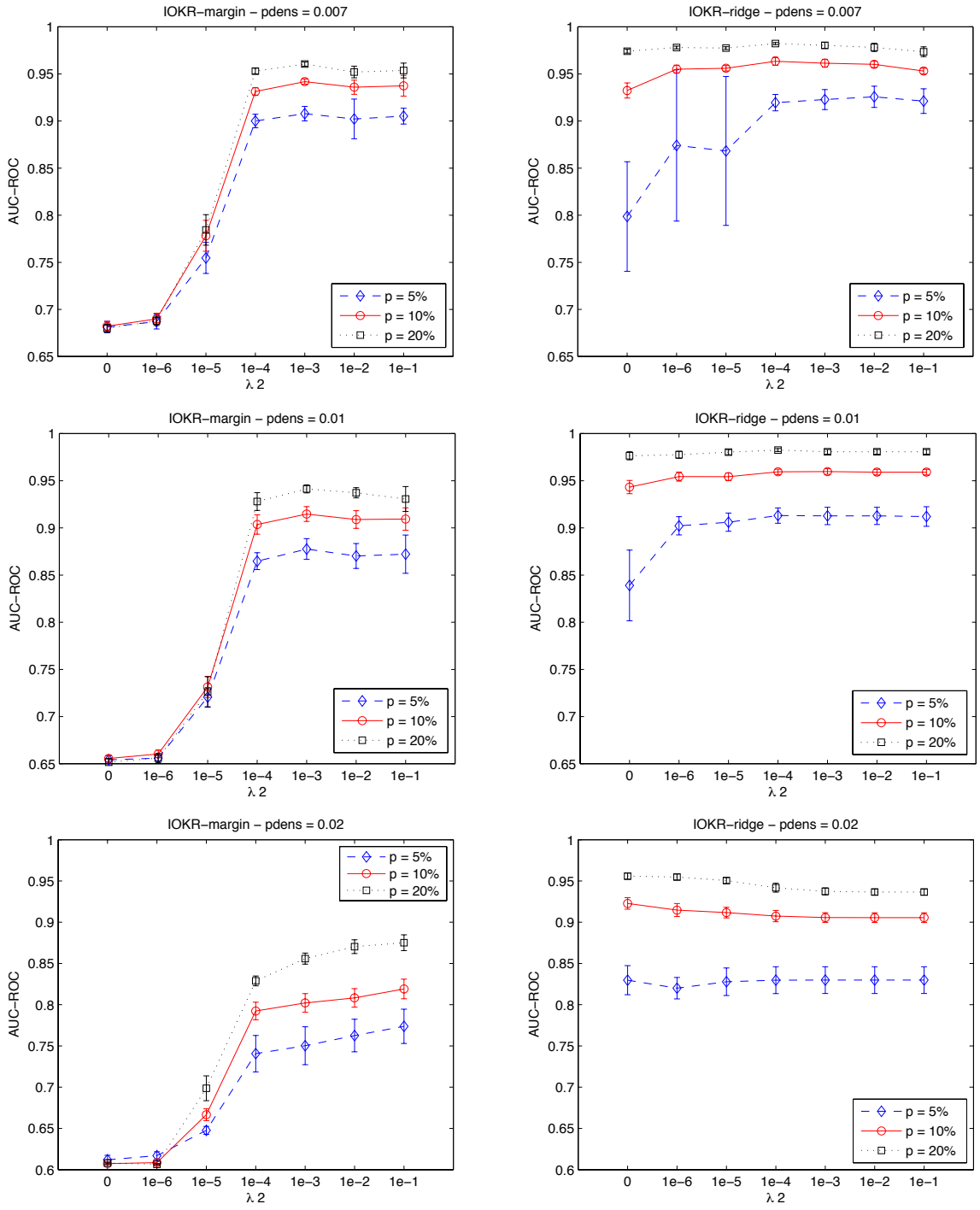

Figure 3: Averaged AUC-ROC for the reconstruction of three synthetics networks with IOKR-margin (left) and IOKR-ridge (right). The rows correspond to different graph densities (denoted pdens), which are 0.007, 0.01 and 0.02 respectively.

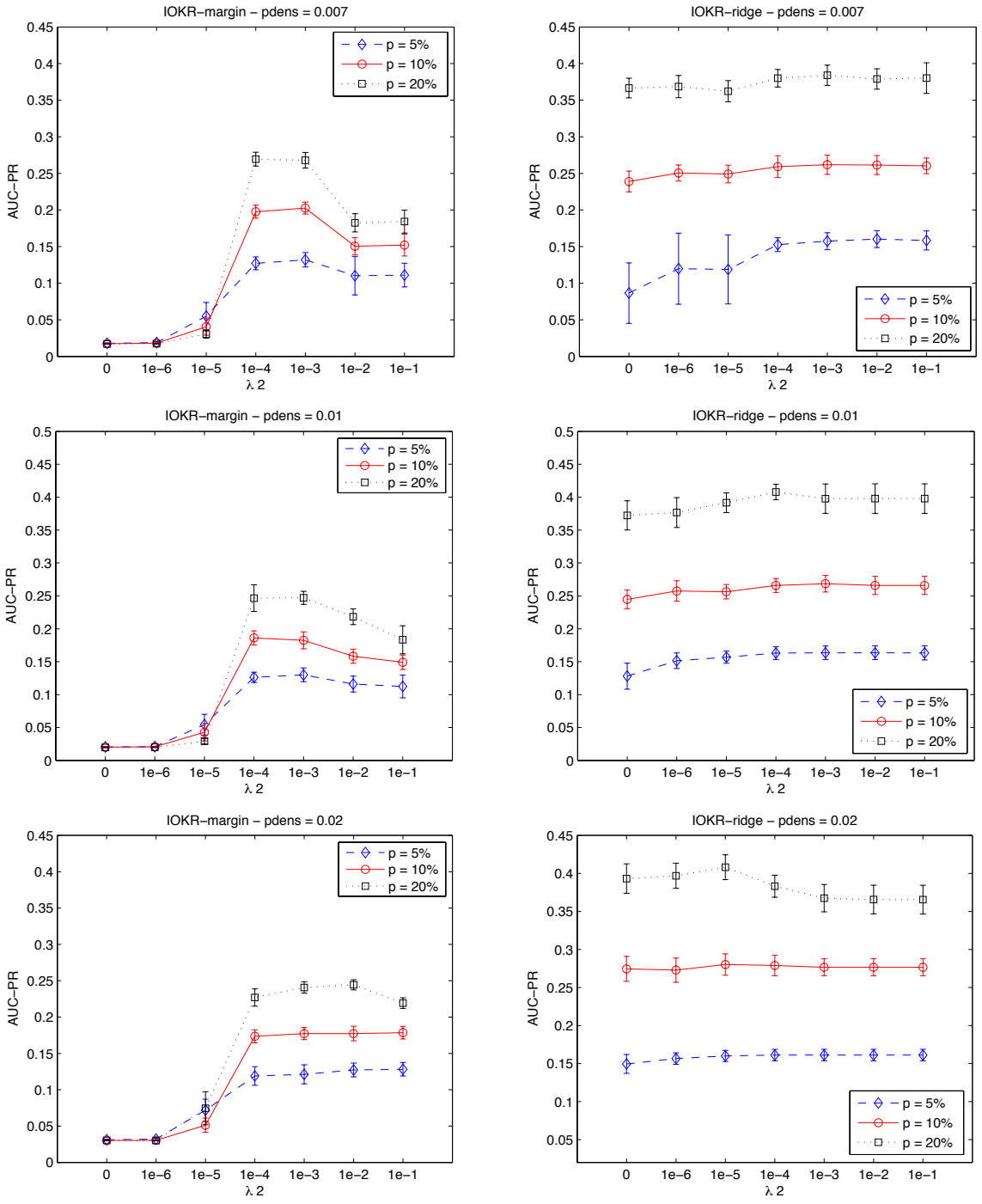

Figure 4: Averaged AUC-PR for the reconstruction of three synthetics networks with IOKR-margin (left) and IOKR-ridge (right). The rows correspond to different graph densities (denoted pdens), which are 0.007, 0.01 and 0.02 respectively.

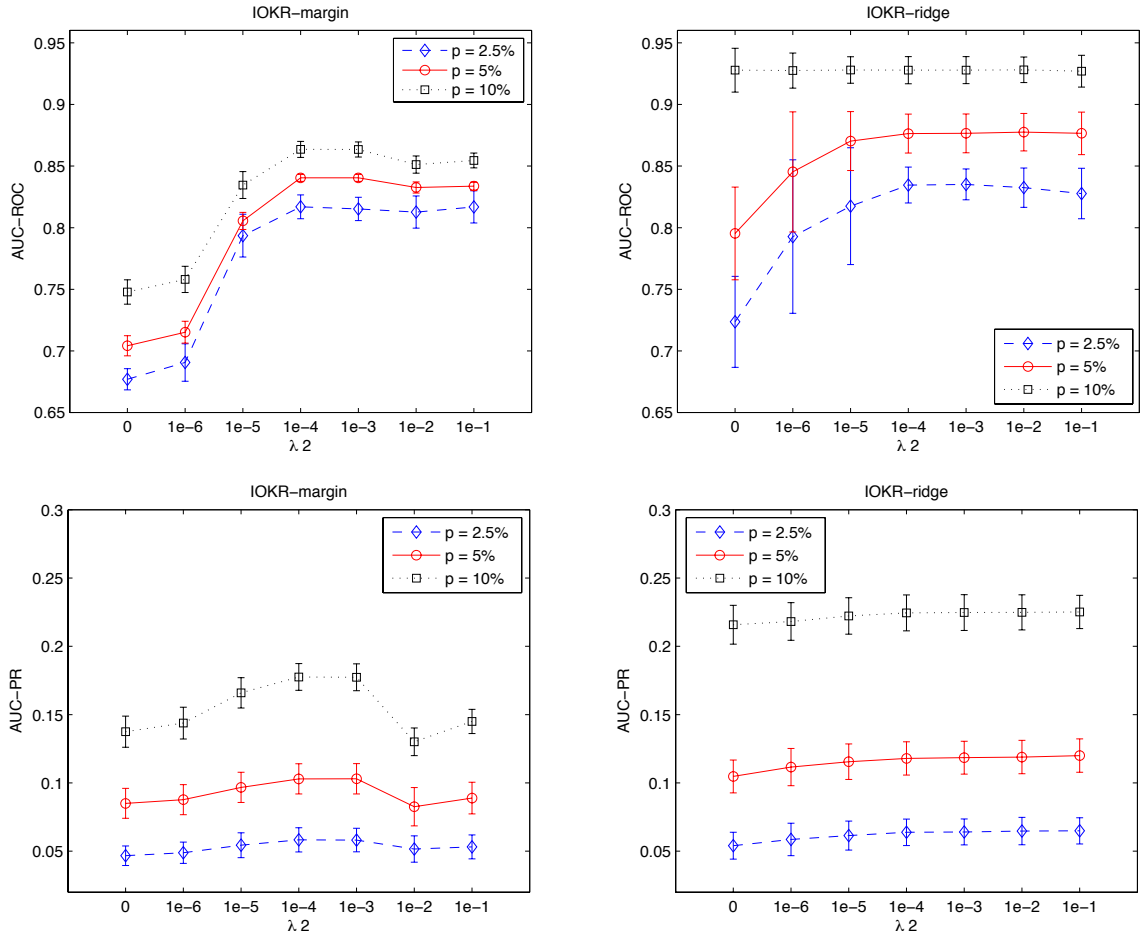

Figure 5: AUC-ROC and AUC-PR obtained for the NIPS co-authorship network inference with the IOKR-margin model (left) and the IOKR-ridge model (right).

previously, we can observe that the semi-supervised approach improves the performances compared to the supervised one for both models. For AUC-ROC values, this improvement is especially important when the percentage of labeled nodes is low. Indeed, with 2.5% of labeled nodes, the improvement can reach in average up to 0.14 points of AUC-ROC for IOKR-margin and up to 0.11 points for IOKR-ridge. As for the synthetic networks, the IOKR-ridge model outperforms IOKR-margin model in terms of AUC-ROC and AUC-PR, especially when the proportion of labeled examples is large. The explanation provided for the synthetic networks regarding the complexity of the solutions for IORK-margin and IOKR-ridge holds here also.

6.1.4 Protein-Protein Interaction Network

We also performed experiments on a protein-protein interaction (PPI) network of the yeast *Saccharomyces Cerevisiae*. This network was built using the DIP database (Salwinski et al.,

2004), which contains protein-protein interactions that have been experimentally determined and manually curated. We used more specifically the high confidence DIP core subset of interactions (Deane et al., 2002). For the input kernels, we used the annotations provided by Gene Ontology (GO) (Ashburner et al., 2000) in terms of biological processes, cellular components and molecular functions. These annotations are organized in three different ontologies. Each ontology is represented by a directed acyclic graph, where each node is a GO annotation and edges correspond to relationships between the annotations, like sub-class relationships for example. A protein can be annotated to several terms in an ontology. We chose to represent each protein u_i by a vector $\mathbf{s}_i$, whose dimension is equal to the total number of terms of the considered ontology. If a protein u_i is annotated by the term t, then :

$$\mathbf{s}_i^{(t)} = -\ln\left(\frac{\text{number of proteins annotated by } t}{\text{total number of proteins}}\right).$$

This encoding allows to take into account the specificity of a term in the ontology. We then used these representations to built a gaussian kernel for each GO ontology. By considering the set of proteins being annotated for each input kernel and being involved in at least one physical interaction, we obtained a PPI network containing 1242 proteins.

Based on the previous numerical results, we chose to consider only IOKR-ridge in the following experiments. We compared our approach to several supervised methods proposed for biological network inference:

- *Naive* (Yamanishi et al., 2004): this approach predicts an interaction between two proteins u and u' if $\kappa_x(u, u')$ is greater than a threshold θ.

- *kCCA* (Yamanishi et al., 2004): kernel CCA is used to detect correlations existing between the input kernel and a diffusion kernel derived from the adjacency matrix of the labeled PPI network.

- *kML* (Vert and Yamanishi, 2005): kernel Metric Learning consists in learning a new metric such that interacting proteins are close to each other, and conversely for non interacting proteins.

- *Local* (Bleakley et al., 2007): a local model is built for each protein in order to learn the subnetwork associated to each protein and these models are then combined together.

- *OK3+ET*(Geurts et al., 2006, 2007a): Output Kernel Tree with extra-trees is a tree-based method where the output is kernelized and is combined with ensemble methods.

The pairwise kernel method (Ben-Hur and Noble, 2005) was not considered here because this method requires to define a Gram matrix between pairs of nodes, which raises some practical issues in terms of computation time and storage.

Each method was evaluated through a 5-fold cross-validation (5-cv) experiment and the hyperparameters were tuned on the training fold using a 4-cv experiment. As the local method can not be used for predicting interactions between two proteins of the test set, AUC-ROC and AUC-PR were only computed for the prediction of interactions between proteins in the test set and proteins in the training set. Input kernel matrices were defined

a) AUC-ROC :

Methods	GO-BP	GO-CC	GO-MF	int
Naive	60.8 ± 0.8	64.4 ± 2.5	64.2 ± 0.8	67.7 ± 1.5
kCCA	82.4 ± 3.6	77.0 ± 1.7	75.0 ± 0.6	85.7 ± 1.6
kML	83.2 ± 2.4	77.8 ± 1.1	76.6 ± 1.9	84.5 ± 1.5
Local	79.5 ± 1.6	73.1 ± 1.3	66.8 ± 1.2	83.0 ± 0.5
OK3+ET	84.3 ± 2.4	81.5 ± 1.6	79.3 ± 1.8	86.9 ± 1.6
IOKR-ridge	$\mathbf{88.8 \pm 1.9}$	$\mathbf{87.1 \pm 1.3}$	$\mathbf{84.0 \pm 0.6}$	$\mathbf{91.2 \pm 1.2}$

b) AUC-PR :

Methods	GO-BP	GO-CC	GO-MF	int
Naive	4.8 ± 1.0	2.1 ± 0.6	2.4 ± 0.4	8.0 ± 1.7
kCCA	7.1 ± 1.5	7.7 ± 1.4	4.2 ± 0.5	9.9 ± 0.4
kML	7.1 ± 1.3	3.1 ± 0.6	3.5 ± 0.4	7.8 ± 1.6
Local	6.0 ± 1.1	1.1 ± 0.3	0.7 ± 0.0	22.6 ± 6.6
OK3+ET	$\mathbf{19.0 \pm 1.8}$	$\mathbf{21.8 \pm 2.5}$	$\mathbf{10.5 \pm 2.0}$	$\mathbf{26.8 \pm 2.4}$
IOKR-ridge	15.3 ± 1.2	20.9 ± 2.1	8.6 ± 0.3	22.2 ± 1.6

Table 3: AUC-ROC and AUC-PR estimated by 5-CV for the yeast PPI network reconstruction in the supervised setting with different input kernels (*GO-BP*: GO biological processes; *GO-CC*: GO cellular components; *GO-MF*: GO molecular functions; *int* : average of the different kernels).

for GO ontology and an integrated kernel, which was obtained by averaging the three input kernels, was also considered.

Table 3 reports the results obtained for the comparison of the different methods in the supervised setting. We can see that output kernel regression based methods work better on this dataset than the other methods. In terms of AUC-ROC, the IOKR-ridge method obtains the best results for the four different input kernels, while for AUC-PR, OK3 with extra-trees presents better performances.

We also compared our method with two transductive approaches: the EM-based approach (Tsuda et al., 2003; Kato et al., 2005) and Penalized Kernel Matrix Regression (PKMR) (Yamanishi and Vert, 2007). These two methods regard the link prediction problem as a kernel matrix completion problem. The *EM* method fills the missing entries of the output Gram matrix K_y by minimizing the information geometry, as measured by the Kullback-Leibler divergence, with the input Gram matrix K_x. The *PKMR* approach considers the kernel matrix regression problem as a regression problem between the labeled input Gram matrix K_{x_ℓ} and the labeled output Gram matrix K_{y_ℓ}. We did not compare our method with the *Link Propagation* framework (Kashima et al., 2009) because this framework assumes that arbitrary interactions may be considered as labeled while IOKR requires a subgraph of know interactions.

Percentage of labeled data	AUC-ROC			AUC-PR		
	EM	PKMR	IOKR	EM	PKMR	IOKR
5	$\mathbf{82.2 \pm 0.6}$	77.5 ± 2.3	80.6 ± 0.7	$\mathbf{15.7 \pm 1.4}$	6.1 ± 1.5	7.1 ± 1.1
10	82.9 ± 0.6	80.8 ± 1.1	$\mathbf{83.1 \pm 0.5}$	$\mathbf{16.5 \pm 2.7}$	9.8 ± 1.8	11.7 ± 1.1
20	$\mathbf{84.6 \pm 0.6}$	83.9 ± 1.2	83.9 ± 0.5	$\mathbf{19.7 \pm 0.7}$	13.8 ± 1.2	17.8 ± 1.5

Table 4: AUC-ROC and AUC-PR obtained for yeast PPI network inference in the transductive setting using the integrated kernel.

As for previous experiences in the transductive setting, we measured the AUC-ROC and AUC-PR values for 5%, 10% and 20% of labeled nodes, and for each percentage, we averaged the AUC over ten random training sets. The hyperparameters were selected by a 3-fold cross-validation experiment for the three methods. We used as input kernel the integrated kernel introduced in the supervised experiments.

The results obtained for the comparison in the transductive setting are reported in the Table 4. Regarding AUC-ROC, the EM approach obtains better results when the percentage of labeled data is 5%. For 10% and 20% of labeled data, the difference between EM and IOKR is not significative. In terms of AUC-PR, EM achieves rather good performances compared to the others, in particular for 5% and 10% of labeled data. For 20%, the IOKR method behaves as well as the EM method. However, we can notice that the EM-based approach is purely transductive while IOKR learns a function and can therefore be used in the semi-supervised learning, which is more general.

6.2 Application to Multi-Task Regression

In the following, we compare the behavior of the IOKR-ridge model regarding the identity and decomposable kernels presented in Section 3 on a drug activity prediction problem. The goal of this problem is to predict the activities of molecules in different cancer cell lines. In this application, $\mathcal{X}$ corresponds to the set of molecules and $\mathcal{Y} = \mathcal{F}_y = \mathbb{R}^d$, where d is the number of cell lines.

6.2.1 Dataset

We used the data set of Su et al. (2010) that contains the biological activities of molecules against a set of 59 cancer cell lines. We used the "No-Zero-Active" version of the data set: this data set contains the 2303 molecules that are all active against at least one cell line. Each molecule is represented by a graph, where nodes correspond to atoms and edges to bonds between atoms. The Tanimoto kernel (Ralaivola et al., 2005), that is based on the molecular graphs, is used for the scalar input kernel:

$$\kappa_x(x, x') = \frac{k_m(x, x')}{k_m(x, x) + k_m(x', x') - k_m(x, x')}.$$

k_m is the kernel corresponding to the feature map $\varphi_{x_m} : \mathcal{X} \to \mathcal{F}_{x_m}$:

$$k_m(x, x') = \langle \varphi_{x_m}(x), \varphi_{x_m}(x') \rangle_{\mathcal{F}_{x_m}},$$

where $\varphi_{x_m}(x)$ is a binary vector indicating the presences and absences in the molecule $x \in \mathcal{X}$ of all existing paths containing a maximum of m bonds. In this application, the value of m was set to 6.

6.2.2 Protocol

We evaluated the behavior of the IOKR-ridge model in the transductive setting. The performances were measured by computing the mean squared error (MSE) on the unlabeled set:

$$MSE = \frac{1}{n} \sum_{i=\ell+1}^{\ell+n} \|h(x_i) - \varphi_y(\mathbf{y}_i))\|^2_{\mathcal{F}_y}.$$

We estimated the similarities existing between the tasks by comparing their values on the training set:

$$M_{ij} = \exp\left(-\gamma \|Y_\ell^{(i)} - Y_\ell^{(j)}\|^2\right),\ i, j = 1, \ldots, d,$$

where $Y_\ell^{(i)} = (\mathbf{y}_1^{(i)}, \mathbf{y}_2^{(i)}, \ldots, \mathbf{y}_\ell^{(i)})$.

The parameter γ of the matrix M was chosen to maximize an information criterion and the regularization parameter λ_1 was set to 1. Regarding the matrix W used in the semi-supervised term, we sparsified the Gram matrix $K_{x_{\ell+n}}$ of the scalar input kernel κ_x using a k-nearest neighbors procedure with $k = 50$. We then computed the graph laplacian of the obtained graph and considered the laplacian iterated to degree 5.

6.2.3 Results

The results presented in Figure 6 were obtained from ten random choices of the training set. The performances obtained with model 1 and model 2 for different percentages of labeled data are represented as a function of the parameters μ and λ_2. We observe on this figure that for both models, using unlabeled data helps to improve the performances. We also observe that when μ is increased from 0 to 0.8 or 1, the mean squared errors are decreased. The obtained results therefore show the benefit of taking into account the relationships existing between the outputs for both models and both settings (supervised and semi-supervised).

We reported on Figure 7 the MSE obtained with models 1 and 2 for the best parameter μ and added the results obtained with the model 0, which corresponds to the case where $A = I$. We observe on this figure that the model 2 obtains better results than the model 1 when the percentage of labeled data is small ($p = 5\%$). For $p = 10\%$, the two models behave similarly, while for 20% of labeled data, the model 1 improves significantly the performances, compared to model 2. Therefore, we observe that using the output structure information either in the input operator-valued kernel or in the output kernel leads to different results. And depending on the amount of labeled data, one of the two models can be more interesting to use.

7 Conclusion and Perspectives

Operator-valued kernels and the associated RKHS theory provide a general framework to address approximation of functions with values in some Hilbert space. When characterizing

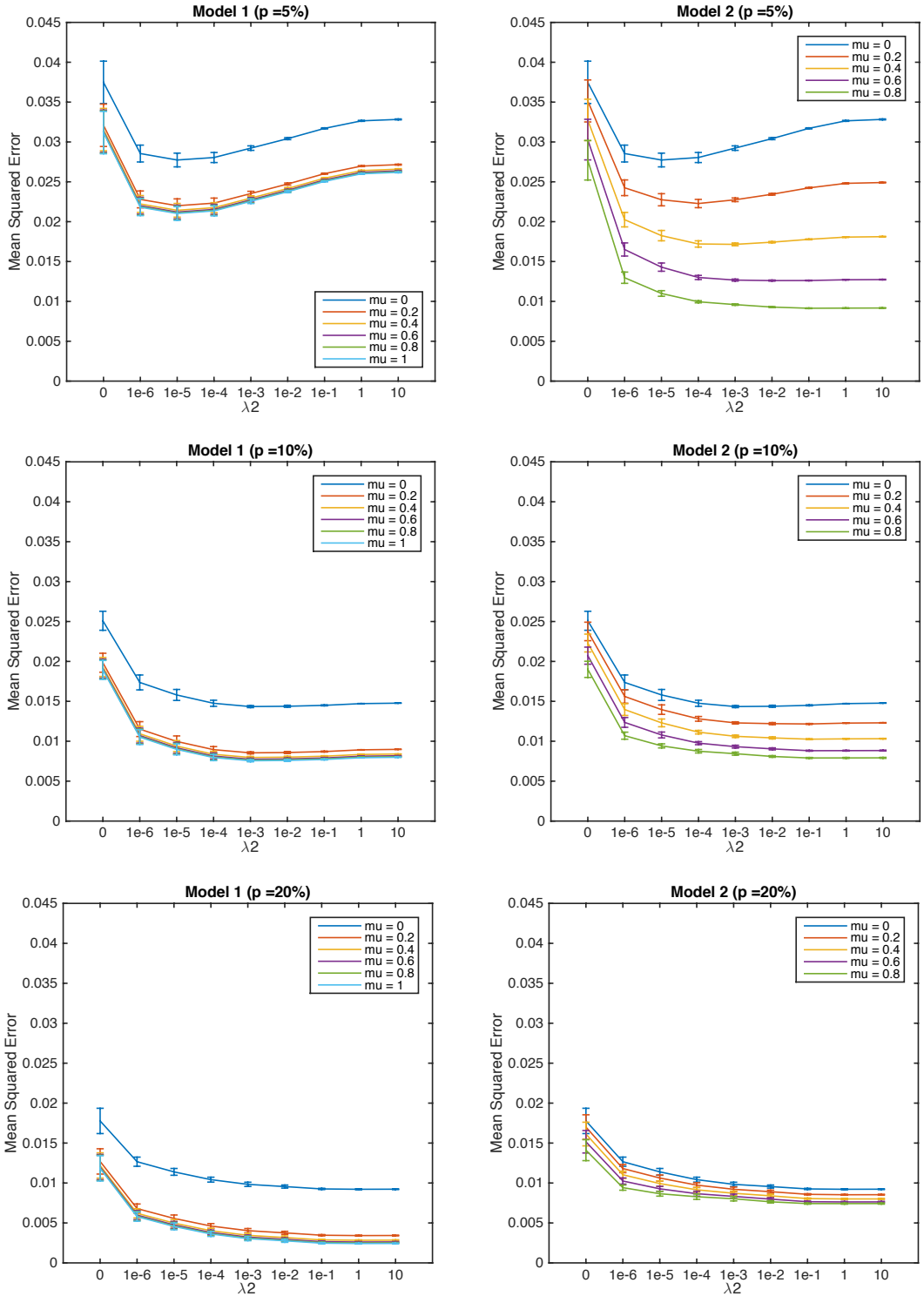

Figure 6: Mean squared errors obtained with the two models for the prediction of molecular activities. The results are averaged over ten random choices of the training set and are given for different percentages of labeled data (5%, 10% and 20%).

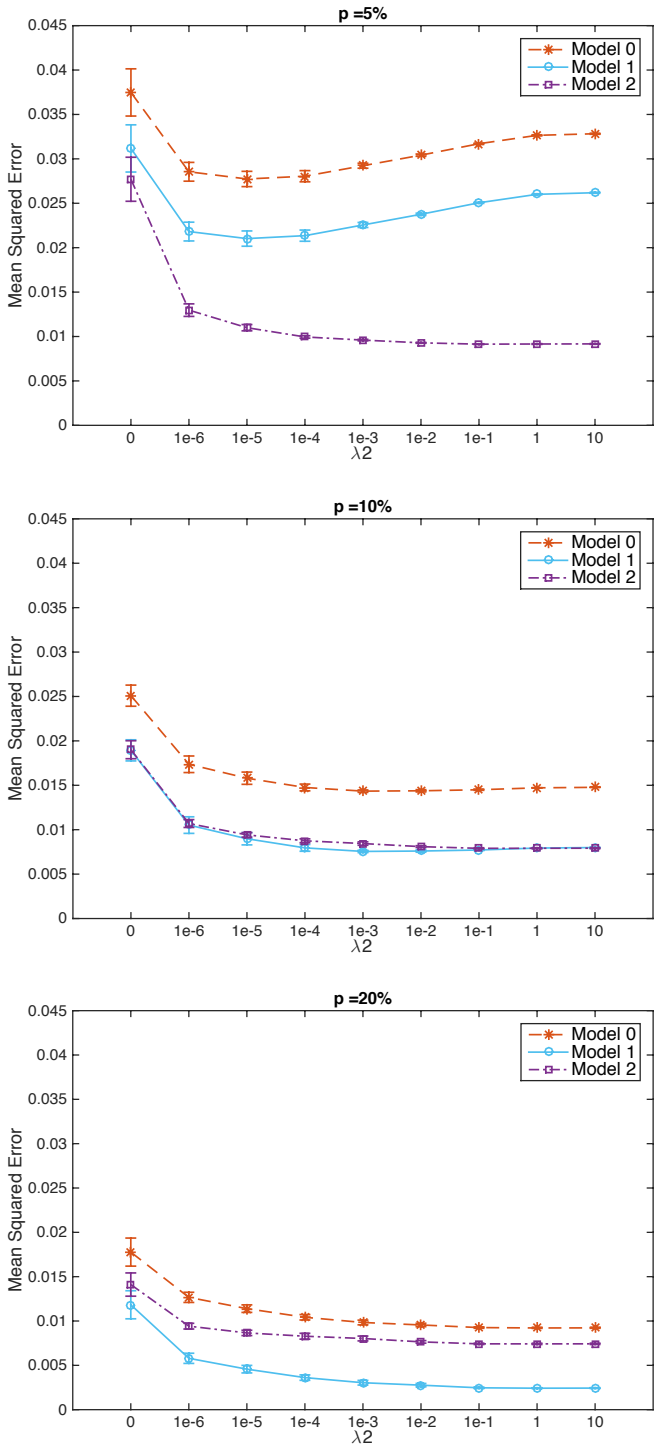

Figure 7: Mean squared errors obtained for the prediction of molecular activities for the model 0 (corresponding to $A = I$), model 1 ($\mu = 1$) and model 2 ($\mu = 0.8$). The results are averaged over ten random choices of the training set and are given for different percentages of labeled data (5%, 10% and 20%).

the output Hilbert space as a feature space related to some real-valued scalar kernel, we get an original framework to deal with structured outputs. Extending our previous work (Brouard et al., 2011) which introduced a new representer theorem for semi-supervised learning with vector-valued functions, we presented solutions of semi-supervised penalized regression developed for two empirical loss functions, the square loss and the hinge loss in the general case and in the special case of decomposable kernels using tensors. We also showed that Generalized Cross-Validation extends in the case of the closed-form solution of IOKR-ridge, providing an efficient tool for model selection. Perspectives to this work concern the construction of new models by minimizing loss functions with different penalties, for instance, penalties that enforce the parsimony of the model. For these non-smooth penalties, proximal gradient descent methods can be applied such as in Lim et al. (2013). A more general research direction is related to the design of new kernels and appropriate kernel learning algorithms. Finally, although the pre-image problem has received a lot of attention in the literature, there is still room for improvement in order to apply IOKR in other tasks than link prediction or multiple output structured regression.

Acknowledgments

We would like to acknowledge support for this project from ANR (grant ANR-009-SYSC-009-02) and University of Evry (PhD grant).

Appendix A. Technical Proofs

In this appendix section, we provide the proofs for some theorems and propositions presented in the paper.

A.1 Proof of Theorem 7

The primal can be written as:

$$\begin{aligned} \min_{h\in\mathcal{H},\{\xi_i\}\in\mathbb{R}} \quad & \lambda_1\|h\|_{\mathcal{H}}^2 + 2\lambda_2 \sum_{i,j=1}^{\ell+n} L_{ij}\langle h(x_i), h(x_j)\rangle_{\tilde{\mathcal{Y}}} + \sum_{i=1}^{\ell}\xi_i \\ \text{s.t.} \quad & \langle \tilde{\mathbf{y}}_i, h(x_i)\rangle_{\tilde{\mathcal{Y}}} \geq 1-\xi_i, i=1,\ldots,\ell \\ & \xi_i \geq 0, i=1,\ldots,\ell \end{aligned}$$

We write the Lagrangian:

$$\begin{aligned} \mathcal{L}_a(h,\boldsymbol{\xi},\boldsymbol{\alpha},\boldsymbol{\eta}) = \lambda_1\|h\|_{\mathcal{H}}^2 + 2\lambda_2 \sum_{i,j=1}^{\ell+n} L_{ij}\langle h(x_i), h(x_j)\rangle_{\tilde{\mathcal{Y}}} + \sum_{i=1}^{\ell}\xi_i \\ - \sum_{i=1}^{\ell}\alpha_i(\langle \tilde{\mathbf{y}}_i, h(x_i)\rangle_{\tilde{\mathcal{Y}}} - 1 + \xi_i) - \sum_{i=1}^{\ell}\eta_i\xi_i. \end{aligned}$$

In the following we note $K_x = \mathcal{K}_x(\cdot, x)$ and $K_x^* = \mathcal{K}_x(x,\cdot)$. By using the reproducing property the expression of the Lagrangian becomes:

$$\begin{aligned} \mathcal{L}_a &= \lambda_1\|h\|_{\mathcal{H}}^2 + 2\lambda_2 \sum_{i,j=1}^{\ell+n} L_{ij}\langle K_{x_i}^* h, K_{x_j}^* h\rangle_{\mathcal{H}} - \sum_{i=1}^{\ell}\alpha_i(\langle \tilde{\mathbf{y}}_i, K_{x_i}^* h\rangle_{\mathcal{H}} - 1) + \sum_{i=1}^{\ell}(1-\alpha_i-\eta_i)\xi_i \\ &= \langle (\lambda_1 I + 2\lambda_2 \sum_{i,j=1}^{\ell+n} L_{ij} K_{x_j} K_{x_i}^*)h, h\rangle_{\mathcal{H}} - \sum_{i=1}^{\ell}\alpha_i\langle K_{x_i}\tilde{\mathbf{y}}_i, h\rangle_{\mathcal{H}} + \sum_{i=1}^{\ell}\alpha_i + \sum_{i=1}^{\ell}(1-\alpha_i-\eta_i)\xi_i \\ &= \langle Bh, h\rangle_{\mathcal{H}} - \sum_{i=1}^{\ell}\alpha_i\langle K_{x_i}\tilde{\mathbf{y}}_i, h\rangle_{\mathcal{H}} + \sum_{i=1}^{\ell}\alpha_i + \sum_{i=1}^{\ell}(1-\alpha_i-\eta_i)\xi_i, \end{aligned}$$

where $B \in \mathcal{B}(h)$ is the operator defined as: $B = \lambda_1 I + 2\lambda_2 \sum_{i,j=1}^{\ell+n} L_{ij} K_{x_i} K_{x_j}^*$. Due to the symmetry of the Laplacian L, this operator is self-adjoint:

$$B^* = \lambda_1 I + 2\lambda_2 \sum_{i,j=1}^{\ell+n} L_{ij} K_{x_j} K_{x_i}^* = \lambda_1 I + 2\lambda_2 \sum_{i,j=1}^{\ell+n} L_{ji} K_{x_i} K_{x_j}^* = \lambda_1 I + 2\lambda_2 \sum_{i,j=1}^{\ell+n} L_{ij} K_{x_i} K_{x_j}^* = B.$$

Differentiating the Lagrangian with respect to ξ_i and h gives:

$$\begin{aligned} &\frac{\partial \mathcal{L}_a}{\partial \xi_i} = 0 \Rightarrow 1-\alpha_i-\eta_i = 0 \\ &\frac{\partial \mathcal{L}_a}{\partial h} = 0 \Rightarrow 2Bh - \sum_{i=1}^{\ell}\alpha_i K_{x_i}\tilde{\mathbf{y}}_i = 0 \Rightarrow h = \frac{1}{2}B^{-1}\left(\sum_{i=1}^{\ell}\alpha_i K_{x_i}\tilde{\mathbf{y}}_i\right). \end{aligned}$$

B is invertible as it is a positive definite operator:

$$\begin{aligned}\forall h \in \mathcal{H},\ \langle h, Bh\rangle_{\mathcal{H}} &= \lambda_1 \|h\|^2_{\mathcal{H}} + 2\lambda_2 \sum_{i,j=1}^{\ell+n} L_{ij}\langle h, K_{x_j} K^*_{x_i} h\rangle_{\mathcal{H}} \\ &= \lambda_1 \|h\|^2_{\mathcal{H}} + 2\lambda_2 \sum_{i,j=1}^{\ell+n} L_{ij}\langle h(x_j), h(x_i)\rangle_{\tilde{\mathcal{Y}}} \\ &= \lambda_1 \|h\|^2_{\mathcal{H}} + \lambda_2 \sum_{i,j=1}^{\ell+n} W_{ij}\|h(x_j) - h(x_i)\|^2_{\tilde{\mathcal{Y}}} \\ &> 0 \text{ for all non-zero function } h.\end{aligned}$$

We formulate a reduced Lagrangian :

$$\begin{aligned}\mathcal{L}_r(\boldsymbol{\alpha}) &= \frac{1}{4}\sum_{i,j=1}^{\ell} \alpha_i\alpha_j \langle BB^{-1}K_{x_i}\tilde{\mathbf{y}}_i, B^{-1}K_{x_j}\tilde{\mathbf{y}}_j\rangle - \frac{1}{2}\sum_{i,j=1}^{\ell} \alpha_i\alpha_j \langle K_{x_i}\tilde{\mathbf{y}}_i, B^{-1}K_{x_j}\tilde{\mathbf{y}}_j\rangle + \sum_{i=1}^{\ell}\alpha_i \\ &= -\frac{1}{4}\sum_{i,j=1}^{\ell} \alpha_i\alpha_j \langle K_{x_i}\tilde{\mathbf{y}}_i, B^{-1}K_{x_j}\tilde{\mathbf{y}}_j\rangle + \sum_{i=1}^{\ell}\alpha_i.\end{aligned}$$

The dual formulation of the optimization problem (6) can thus be expressed as:

$$\begin{aligned}\min_{\boldsymbol{\alpha}\in\mathbb{R}^\ell} \quad & \frac{1}{4}\sum_{i,j=1}^{\ell} \alpha_i\alpha_j \langle K_{x_i}\tilde{\mathbf{y}}_i, B^{-1}K_{x_j}\tilde{\mathbf{y}}_j\rangle - \sum_{i=1}^{\ell}\alpha_i \\ \text{s.t.} \quad & 0 \le \alpha_i \le 1, i = 1, \ldots, \ell\end{aligned}$$

A.2 Proof of Proposition 9

We start from Equation (8) and replace A by its eigenvalue decomposition:

$$\operatorname{vec}(C_{\ell+n}) = \left(\lambda_1 I_{(\ell+n)d} + M \otimes A\right)^{-1} \operatorname{vec}(\tilde{Y}_\ell J),$$

where $M = (J^T J + 2\lambda_2 L)K_{x_{\ell+n}}$.

We introduce the vec-permutation matrices P_{mn} and P_{nm} defined as:

$$\forall A \in \mathbb{R}^{m\times n},\ \operatorname{vec}(A^T) = P_{mn}\operatorname{vec}(A) \text{ and } \operatorname{vec}(A) = P_{nm}\operatorname{vec}(A^T).$$

For any $m \times n$ matrix A and $p \times q$ matrix B,

$$B \otimes A = P_{pm}(A \otimes B)P_{nq}.$$

Using these properties, we can write:

$$\begin{aligned}
\operatorname{vec}(C_{\ell+n}^T) &= P_{d(\ell+n)} \operatorname{vec}(C_{\ell+n}) \\
&= P_{d(\ell+n)} \left(\lambda_1 I_{(\ell+n)d} + P_{(\ell+n)d}(A \otimes M) P_{d(\ell+n)}\right)^{-1} \operatorname{vec}(\tilde{Y}_\ell J) \\
&= \left(\lambda_1 I_{(\ell+n)d} + P_{d(\ell+n)} P_{(\ell+n)d}(A \otimes M)\right)^{-1} P_{d(\ell+n)} \operatorname{vec}(\tilde{Y}_\ell J) \\
&= \left(\lambda_1 I_{(\ell+n)d} + A \otimes M\right)^{-1} \operatorname{vec}(J^T \tilde{Y}_\ell^T) \\
&= \left(\lambda_1 I_{(\ell+n)d} + E\Gamma E^T \otimes M\right)^{-1} \operatorname{vec}(J^T \tilde{Y}_\ell^T).
\end{aligned}$$

We multiply each side by $(E^T \otimes I_{\ell+n})$

$$\begin{aligned}
&(E^T \otimes I_{\ell+n}) \operatorname{vec}(C_{\ell+n}^T) = \\
&\quad (E^T \otimes I_{\ell+n}) \left(\lambda_1 I_{(\ell+n)d} + (E \otimes I_{\ell+n})(\Gamma \otimes M)(E^T \otimes I_{\ell+n})\right)^{-1} \operatorname{vec}(J^T \tilde{Y}_\ell^T).
\end{aligned}$$

We use the facts that $\operatorname{vec}(AXB) = (B^T \otimes A) \operatorname{vec}(X)$ and that $E^T E = I_d$ to obtain the following equation:

$$\operatorname{vec}(C_{\ell+n}^T E) = (\lambda_1 I_{(\ell+n)d} + \Gamma \otimes M)^{-1} \operatorname{vec}(J^T \tilde{Y}_\ell^T E).$$

The matrix $(\lambda_1 I_{(\ell+n)d} + \Gamma \otimes M)$ being block-diagonal, we have

$$C_{\ell+n}^T \mathbf{e}_i = (\lambda_1 I_{\ell+n} + \gamma_i M)^{-1} J^T \tilde{Y}_\ell^T \mathbf{e}_i, \text{ for } i = 1, \ldots, \ell+n.$$

Then, we can express the model h as:

$$\begin{aligned}
\forall x \in \mathcal{X}, h(x) = A C_{\ell+n} \Phi_{x_{\ell+n}}^T \varphi_x(x) &= \sum_{j=1}^d \gamma_j \mathbf{e}_j \mathbf{e}_j^T C_{\ell+n} \Phi_{x_{\ell+n}}^T \varphi_x(x) \\
&= \sum_{j=1}^d \gamma_j \mathbf{e}_j \mathbf{e}_j^T \tilde{Y}_\ell J (\lambda_1 I_{\ell+n} + \gamma_j K_{x_{\ell+n}} (J^T J + 2\lambda_2 L))^{-1} \Phi_{x_{\ell+n}}^T \varphi_x(x).
\end{aligned}$$

In the supervised setting ($\lambda_2 = 0$), the model h writes as:

$$\forall x \in \mathcal{X}, h(x) = \sum_{j=1}^d \gamma_j \mathbf{e}_j \mathbf{e}_j^T \tilde{Y}_\ell (\lambda_1 I_\ell + \gamma_j K_{x_\ell})^{-1} \Phi_{x_\ell}^T \varphi_x(x).$$

This completes the proof.

A.3 Proof of Proposition 10

Let $Z_\ell = \tilde{Y}_\ell \operatorname{diag}(\boldsymbol{\alpha}) J$. We start from the expression of the Lagrangian in the case of a general operator-valued kernel (Equation 9) and replace A by its eigenvalue decomposition:

$$\begin{aligned}
\mathcal{L}_a(\boldsymbol{\alpha}) &= -\frac{1}{4} \operatorname{vec}(Z_\ell)^T \left(\lambda_1 I_{(\ell+n)d} + 2\lambda_2 K_{x_{\ell+n}} L \otimes A\right)^{-1} (K_{x_{\ell+n}} \otimes A) \operatorname{vec}(Z_\ell) + \boldsymbol{\alpha}^T \mathbf{1} \\
&= -\frac{1}{4} \operatorname{vec}(Z_\ell)^T \left(\lambda_1 I_{(\ell+n)d} + 2\lambda_2 (I_{\ell+n} \otimes E)(K_{x_{\ell+n}} L \otimes \Gamma)(I_{\ell+n} \otimes E^T)\right)^{-1} \\
&\quad (I_{\ell+n} \otimes E)(K_{x_{\ell+n}} \otimes \Gamma)(I_{\ell+n} \otimes E^T) \operatorname{vec}(Z_\ell) + \boldsymbol{\alpha}^T \mathbf{1}. \\
&= -\frac{1}{4} \operatorname{vec}\left(E^T Z_\ell\right)^T \left(\lambda_1 I_{(\ell+n)d} + 2\lambda_2 K_{x_{\ell+n}} L \otimes \Gamma\right)^{-1} (K_{x_{\ell+n}} \otimes \Gamma) \operatorname{vec}(E^T Z_\ell) + \boldsymbol{\alpha}^T \mathbf{1}.
\end{aligned}$$

Using the vec-permutation matrices, we can show that:

$$\mathcal{L}_a(\boldsymbol{\alpha}) = -\frac{1}{4}\operatorname{vec}\left(Z_\ell^T E\right)^T \left(\lambda_1 I_{(\ell+n)d} + 2\lambda_2 \Gamma \otimes K_{x_{\ell+n}} L\right)^{-1} (\Gamma \otimes K_{x_{\ell+n}}) \operatorname{vec}\left(Z_\ell^T E\right) + \boldsymbol{\alpha}^T \mathbf{1}.$$

As $(\lambda_1 I_{(\ell+n)d} + 2\lambda_2 \Gamma \otimes K_{x_{\ell+n}} L)$ is a block diagonal matrix, we can write:

$$\begin{aligned}
\mathcal{L}_a(\boldsymbol{\alpha}) =& -\frac{1}{4}\sum_{i=1}^{d} \mathbf{e}_i^T Z_\ell (\lambda_1 I_{\ell+n} + 2\lambda_2 \gamma_i K_{x_{\ell+n}} L)^{-1} \gamma_i K_{x_{\ell+n}} Z_\ell^T \mathbf{e}_i + \boldsymbol{\alpha}^T \mathbf{1} \\
=& -\frac{1}{4}\sum_{i=1}^{d} \gamma_i \operatorname{trace}\left(\tilde{Y}_\ell^T \mathbf{e}_i \mathbf{e}_i^T \tilde{Y}_\ell \operatorname{diag}(\boldsymbol{\alpha}) J (\lambda_1 I_{\ell+n} + 2\lambda_2 \gamma_i K_{x_{\ell+n}} L)^{-1} K_{x_{\ell+n}} J^T \operatorname{diag}(\boldsymbol{\alpha})\right) \\
&+ \boldsymbol{\alpha}^T \mathbf{1}.
\end{aligned}$$

Using the fact that $\mathbf{y}^T (A \circ B)\mathbf{x} = \operatorname{trace}(\operatorname{diag}(\mathbf{y})^T A \operatorname{diag}(\mathbf{x}) B^T)$, the Lagrangian can be written as:

$$\mathcal{L}_a(\boldsymbol{\alpha}) = -\frac{1}{4}\sum_{i=1}^{d} \gamma_i \, \boldsymbol{\alpha}^T (\tilde{Y}_\ell^T \mathbf{e}_i \mathbf{e}_i^T \tilde{Y}_\ell \circ J (\lambda_1 I_{\ell+n} + 2\lambda_2 \gamma_i K_{x_{\ell+n}} L)^{-1} K_{x_{\ell+n}} J^T) \boldsymbol{\alpha} + \boldsymbol{\alpha}^T \mathbf{1}.$$

In the supervised setting ($\lambda_2 = 0$), the Lagrangian becomes:

$$\begin{aligned}
\mathcal{L}_a(\boldsymbol{\alpha}) &= -\frac{1}{4\lambda_1}\sum_{i=1}^{d} \gamma_i \, \boldsymbol{\alpha}^T (\tilde{Y}_\ell^T \mathbf{e}_i \mathbf{e}_i^T \tilde{Y}_\ell \circ K_{x_\ell}) \boldsymbol{\alpha} + \boldsymbol{\alpha}^T \mathbf{1} \\
&= -\frac{1}{4\lambda_1} \boldsymbol{\alpha}^T (\tilde{Y}_\ell^T A \tilde{Y}_\ell \circ K_{x_\ell}) \boldsymbol{\alpha} + \boldsymbol{\alpha}^T \mathbf{1},
\end{aligned}$$

which concludes the proof.

References

D. M. Allen. The relationship between variable selection and data augmentation and a method for prediction. *Technometrics*, 16(1):125–127, 1974.

M. A. Álvarez, L. Rosasco, and N. D. Lawrence. Kernels for vector-valued functions: a review. *Foundations and Trends in Machine Learning*, 4(3):195–266, 2012.

N. Aronszajn. Theory of reproducing kernels. *Transactions of the American Mathematical Society*, pages 337–404, 1950.

M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25(1): 25–29, 2000.

L. Baldassarre, L. Rosasco, A. Barla, and A. Verri. Multi-output learning via spectral filtering. *Machine Learning*, 87(3):259–301, 2012.

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

A. Ben-Hur and W. S. Noble. Kernel methods for predicting protein–protein interactions. *Bioinformatics*, 21(1):38–46, 2005. ISSN 1367-4803.

K. Bleakley, G. Biau, and J.-P. Vert. Supervised reconstruction of biological networks with local models. *Bioinformatics*, 23(13):i57–i65, 2007.

C. Brouard. *Inférence de réseaux d'interaction protéine-protéine par apprentissage statistique*. PhD thesis, University of Evry, France, feb 2013.

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

J. Burbea and P. Masani. Banach and Hilbert spaces of vector-valued functions. *Pitman Research Notes in Mathematics*, 90, 1984.

A. Caponnetto, C. A. Micchelli, M. , and Y. Ying. Universal multitask kernels. *Journal of Machine Learning Research*, 9:1615–1646, 2008.

C. Carmeli, E. De Vito, A. Toigo, and V. Umanita. Vector valued reproducing kernel Hilbert spaces and universality. *Analysis and Applications*, 8:19–61, 2010.

C. Cortes, M. Mohri, and J. Weston. A general regression technique for learning transductions. In *International Conference on Machine Learning (ICML)*, pages 153–160, 2005.

C. M. Deane, L. Salwinski, I. Xenarios, and D. Eisenberg. Protein interactions: two methods for assessment of the reliability of high throughput observations. *Molecular & Cellular Proteomics*, 1(5):349–356, 2002.

F. Dinuzzo, C.S. Ong, P. Gehler, and G. Pillonetto. Learning output kernels with block coordinate descent. In *International Conference on Machine Learning (ICML)*, pages 49–56, 2011.

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

P. Geurts, L. Wehenkel, and F. dAlché-Buc. Kernelizing the output of tree-based methods. In *International Conference on Machine Learning (ICML)*, pages 345–352, 2006.

P. Geurts, N. Touleimat, M. Dutreix, and F. d'Alché-Buc. Inferring biological networks with output kernel trees. *BMC Bioinformatics (PMSB06 special issue)*, 8(Suppl 2):S4, 2007a.

P. Geurts, L. Wehenkel, and F. d'Alché-Buc. Gradient boosting for kernelized output spaces. In *International Conference on Machine Learning (ICML)*, volume 227, pages 289–296, 2007b.

A. Globerson, G. Chechik, F. Pereira, and N. Tishby. Euclidean embedding of co-occurrence data. *Journal of Machine Learning Research*, 8:2265–2295, 2007.

G. H. Golub, M. Heath, and G. Wahba. Generalized cross-validation as a method for choosing a good ridge parameter. *Technometrics*, 21(2):215–223, 1979.

P. Honeine and C. Richard. Preimage problem in kernel-based machine learning. *IEEE Signal Processing Magazine*, 28(2):77–88, 2011.

H. Kadri, E. Duflos, P. Preux, S. Canu, and M. Davy. Nonlinear functional regression: a functional rkhs approach. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 374–380, 2010.

H. Kadri, M. Ghavamzadeh, and P. Preux. A generalized kernel approach to structured output learning. In *International Conference on Machine Learning (ICML)*, pages 471–479, 2013.

H. Kashima, T. Kato, Y. Yamanishi, M. Sugiyama, and K. Tsuda. Link propagation: a fast semi-supervised learning algorithm for link prediction. In *SIAM International Conference on Data Mining*, pages 1099–1110, 2009.

T. Kato, K. Tsuda, and K. Asai. Selective integration of multiple biological data for supervised network inference. *Bioinformatics*, 21(10):2488–2495, 2005.

R. I. Kondor and J. D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *International Conference on Machine Learning (ICML)*, pages 315–322, 2002.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning (ICML)*, pages 282–289, 2001.

J. Lafferty, X. Zhu, and Y. Liu. Kernel conditional random fields: representation and clique selection. In *International Conference on Machine Learning (ICML)*, pages 504–511, 2004.

C. H. Lampert and M. B. Blaschko. Structured prediction by joint kernel support estimation. *Machine Learning*, 77(2-3):249–269, 2009.

N. Lim, Y. Senbabaoglu, G. Michailidis, and F. d'Alché-Buc. Okvar-boost: a novel boosting algorithm to infer nonlinear dynamics and interactions in gene regulatory networks. *Bioinformatics*, 29(11):1416–1423, 2013.

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

H. Q. Minh and V. Sindhwani. Vector-valued manifold regularization. In *International Conference on Machine Learning (ICML)*, pages 57–64, 2011.

N.D. Pearce and M.P. Wand. Penalized splines and reproducing kernel methods. *The American Statistician*, 60(3):233–240, august 2006.

G. Pedrick. Theory of reproducing kernels for Hilbert spaces of vector-valued functions. Technical report, University of Kansas, Department of Mathematics, 1957.

L. Ralaivola, S. J. Swamidass, H. Saigo, and P. Baldi. Graph kernels for chemical informatics. *Neural Network*, 18(8):1093–1110, 2005.

R. M. Rifkin and R. A. Lippert. Notes on regularized least-squares. Technical report, MIT, Computer Science and Artificial Intelligence Laboratory, 2007.

L. Salwinski, C. S. Miller, A. J. Smith, F.K. Pettit, J. U. Bowie, and D. Eisenberg. The database of interacting proteins: 2004 update. *Nucleic Acids Research*, 32 (Database Issue):D449–D451, 2004.

E. Senkene and A. Tempel'man. Hilbert spaces of operator-valued functions. *Lithuanian Mathematical Journal*, 13(4):665–670, 1973.

D. Sheldon. Graphical multi-task learning. Technical report, Cornell University, 2008. URL `http://web.engr.oregonstate.edu/~sheldon/`.

H. Su, M. Heinonen, and J. Rousu. Structured output prediction of anti-cancer drug activity. In *International Conference on Pattern Recognition In Bioinformatics (PRIB)*, pages 38–49. Springer-Verlag, 2010.

S. Szedmak, J. Shawe-Taylor, and E. Parado-Hernandez. Learning via linear operators: Maximum margin regression. Technical report, University of Southampton, UK, 2005.

B. Taskar, C. Guestrin, and D. Koller. Max-margin Markov networks. In *Advances in Neural Information Processing Systems (NIPS)*, volume 16, page 25, 2004.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *International Conference on Machine Learning (ICML)*, page 104, 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.

K. Tsuda, S. Akaho, and K. Asai. The em algorithm for kernel matrix completion with auxiliary data. *Journal of Machine Learning Research*, 4:67–81, 2003.

J.-P. Vert and Y. Yamanishi. Supervised graph inference. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1433–1440, 2005.

G. Wahba. *Spline Model for Observational Data*. Philadelphia, Society for Industrial and Applied Mathematics, 1990.

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

Y. Yamanishi and J.-P. Vert. Kernel matrix regression. In *International Conference on Applied Stochastic Models and Data Analysis (ASMDA)*, 2007.

Y. Yamanishi, J.-P. Vert, and M. Kanehisa. Protein network inference from multiple genomic data: a supervised approach. *Bioinformatics*, 20:i363–i370, 2004.