Learning coordinate gradients with multi-task kernels

Yiming Ying and Colin Campbell*

Department of Engineering Mathematics, University of Bristol
Queens Building, University Walk, Bristol, BS8 1TR, UK.
{enxyy, C.Campbell}@bris.ac.uk

Abstract

Coordinate gradient learning is motivated by the problem of variable selection and determining variable covariation. In this paper we propose a novel unifying framework for coordinate gradient learning (MGL) from the perspective of multi-task learning. Our approach relies on multi-task kernels to simulate the structure of gradient learning. This has several appealing properties. Firstly, it allows us to introduce a novel algorithm which appropriately captures the inherent structure of coordinate gradient learning. Secondly, this approach gives rise to a clear algorithmic process: a computational optimization algorithm which is memory and time efficient. Finally, a statistical error analysis ensures convergence of the estimated function and its gradient to the true function and true gradient. We report some preliminary experiments to validate MGL for variable selection as well as determining variable covariation.

1 Introduction

Let $X \subseteq \mathbb{R}^d$ be compact, $Y \subseteq \mathbb{R}$, $Z = X \times Y$, and $\mathbb{N}_n = \{1, 2 \ldots, n\}$ for any $n \in \mathbb{N}$. A common theme in machine learning is to learn a target function $f_* : X \to Y$ from a finite set of input/output samples $\mathbf{z} = \{(x_i, y_i) : i \in \mathbb{N}_m\} \subseteq Z$. However, in many applications, we not only wish to learn the target function, but also want to find which variables are salient and how these variables interact with each other. This problem has practical motivations: to facilitate data visualization and dimensionality reduction, for example. Such a motivation is important when there are many redundant variables and we wish to find the salient features among these. These problems can occur in many contexts. For example, with gene expression array datasets, the vast majority of features may be redundant to a classification task and we need to find a small set of genuinely distinguishing features. These motivations have driven the design of various statistical and machine learning models [8, 11, 21, 22] for variable (feature) selection.

Here, we build on previous contributions [15, 16, 17] by addressing coordinate gradient learning and its use for variable selection and covariation learning, the interaction between variables. Specifically, for any $x \in X$, we denote x by $(x^1, x^2, \ldots, x^d)$. The target is to learn the gradient of f_* (if it exists) denoted by a vector-valued function $\nabla f_*(x) = \left(\frac{\partial f_*}{\partial x^1}, \ldots, \frac{\partial f_*}{\partial x^d}\right)$. The intuition behind gradient learning for variable selection and coordinate covariation is the following. The inner product between components of ∇f_* indicates the interaction between coordinate variables. Specific norms of $\frac{\partial f_*}{\partial x^p}$ can indicate the salience of the p-th variable: the smaller the norm is, the less important this variable will be.

In this paper we propose a novel unifying formulation of coordinate gradient learning from the perspective of multi-task learning. Learning multiple tasks together has been extensively studied both theoretically and practically in several papers [1, 2, 5, 7, 13, 14]. One way to frame this problem is to learn a vector-valued function where each of its components is a real-valued function and corresponds to a particular task. A key objective in this formulation is to capture an appropriate structure among tasks so that common information is shared across tasks. Here we follow this methodology and employ a vector-valued function $\overrightarrow{f} = (f_1, \overrightarrow{f_2}) = (f_1, f_2, \ldots, f_{d+1})$, where f_1 is used to simulate f_*, and $\overrightarrow{f_2}$ is used to simulate its gradient ∇f_*. We assume that $\overrightarrow{f}$ comes from a vector-valued reproducing kernel Hilbert space (RKHS) associated with multi-task (matrix-valued) kernels, see [14]. The rich structure of RKHS space reflects the latent structure of multi-task gradient learning, i.e. the pooling of information across components (tasks) of ∇f_* using multi-task kernels.

The paper is organized as follows. In Section 2, we first review the definition of multi-task kernels and vector-valued RKHS. Then, we propose a unifying formulation of coordinate gradient learning from the perspective of multi-task learning which is referred to as multi-task gradient learning (MGL). The choices of multi-task kernels motivate different learning models [11, 15, 16, 17]. This allows us to introduce a novel choice of multi-task kernel which reveals the inherent structure of gradient learning. Kernel methods [19, 20] usually enjoy the representer theorem which paves the way for designing efficient optimization algorithms. In Section 3 we explore a representer theorem for MGL algorithms. Subsequently, in Section 4 we discuss computational optimization approaches for MGL algorithms, mainly focusing on least square loss and the SVM counterpart for gradient learning.

*We acknowledge support from EPSRC grant EP/E027296/1.

A statistical error analysis in Section 5 ensures the convergence of the estimated function and its gradient to the true function and true gradient. Finally, in Section 6 preliminary numerical experiments are reported to validate our proposed approach.

1.1 Related work

A number of machine learning and statistical models have been proposed for variable (feature) selection. Least absolute shrinkage and selection operator (LASSO) [21] and basis pursuit denoising [8] suggest use of ℓ^1 regularization to remove redundant features. Weston *et al* [22] introduced a method for selecting features by minimizing bounds on the leave-one-out error.

Guyon *et al* [11] proposed recursive feature elimination (RFE) which used a linear kernel SVM: variables with least influence on the weights $\frac{1}{2}\|w\|^2$ are considered least important. Although these algorithms are promising, there remain unresolved issues. For example, they do not indicate variable covariation and the extension of these algorithms to the nonlinear case was marginally discussed. Our method outlined here covers variable covariation and nonlinear feature selection. As such, in Section 2, we show that RFE-SVM is a special case of our multi-task formulation.

Motivated by the Taylor expansion of a function at samples $\{x_i : i \in \mathbb{N}_m\}$, Mukherjee *et al* [15, 16, 17] proposed an algorithm for learning the gradient function. They used the norm of its components for variable (feature) selection and spectral decomposition of the covariance of the learned gradient function for dimension reduction [16]. Specifically, let $\mathcal{H}_G$ be a scalar RKHS (see e.g. [3]) and use $f_1 \in \mathcal{H}_G$ to simulate f_*. For any $p \in \mathbb{N}_d$, a function $f_{p+1} \in \mathcal{H}_G$ is used to learn $\partial f_*/\partial x^p$. The results presented by Mukherjee *et al* are quite promising both theoretically and practically, but there is no pooling information shared across the components of the gradient. This may lead to less accurate approximation to the true gradient. We will address all these issues in our unifying framework.

2 Multi-task kernels and learning gradients

In this section we formulate the gradient learning problem from the perspective of multi-task learning. Specifically, we employ a vector-valued RKHS to simulate the target function and its gradient. The abundant structure of vector-valued RKHS enables us to couple information across components of the gradient in terms of multi-task kernels.

2.1 Multi-task model for gradient learning

We begin with a review of the definition of multi-task kernels and introduce vector-valued RKHS (see [14] and the reference therein). Throughout this paper, we use the notation $\langle\cdot,\cdot\rangle$ and $\|\cdot\|$ to denote the standard Euclidean inner product and norm respectively.

Definition 1 *We say that a function $\mathcal{K} : X \times X \to R^{d+1} \times \mathbb{R}^{d+1}$ is a multi-task (matrix-valued) kernel on X if, for any $x, t \in X$, $\mathcal{K}(x,t)^T = \mathcal{K}(t,x)$, and it is* positive semi-definite*, i.e., for any $m \in \mathbb{N}$, $\{x_j \in X : j \in \mathbb{N}_m\}$ and $\{\mathbb{y}_j \in \mathbb{R}^{d+1} : j \in \mathbb{N}_m\}$ there holds*

$$\sum_{i,j\in\mathbb{N}_m} \langle \mathbb{y}_i, \mathcal{K}(x_i,x_j)\mathbb{y}_j\rangle \geq 0. \tag{1}$$

In the spirit of Moore-Aronszjain's theorem, there exists a one-to-one correspondence between the multi-task kernel $\mathcal{K}$ with property (1) and a vector-valued RKHS of functions $\overrightarrow{f} : X \to \mathbb{R}^{d+1}$ with norm $\langle\cdot,\cdot\rangle_{\mathcal{K}}$ denoted by $\mathcal{H}_{\mathcal{K}}$, see e.g. [14]. Moreover, for any $x \in X$, $\mathbb{y} \in \mathbb{R}^{d+1}$ and $\overrightarrow{f} \in \mathcal{H}_{\mathcal{K}}$, we have the reproducing property

$$\langle \overrightarrow{f}(x), \mathbb{y}\rangle = \langle \overrightarrow{f}, \mathcal{K}_x\mathbb{y}\rangle_{\mathcal{K}} \tag{2}$$

where $\mathcal{K}_x\mathbb{y} : X \to R^{d+1}$ is defined, for any $t \in X$, by $\mathcal{K}_x\mathbb{y}(t) := \mathcal{K}(t,x)\mathbb{y}$.

In the following we describe our multi-task kernel-based framework for gradient learning. Following Mukherjee *et al* [15, 17], the derivation of gradient learning can be motivated by the Taylor expansion of f_*: $f_*(x_i) \approx f_*(x_j) + \nabla f_*(x_j)(x_i - x_j)^T$. Since we wish to learn f_* with f_1 and ∇f_* with $\overrightarrow{f_2}$, replacing $f_*(x_i)$ by y_i, the error[1]

$$y_i \approx f_1(x_j) + \overrightarrow{f_2}(x_j)(x_i - x_j)^T$$

is expected to be small whenever x_i is close to x_j. To enforce the constraint that x_i is close to x_j, we introduce a weight function produced by a Gaussian with deviation s defined by $w_{ij} = \frac{1}{s^{d+2}} e^{\frac{-\|x_i - x_j\|^2}{2s^2}}$. This implies that $w_{ij} \approx 0$ if x_i is far away from x_j.

We now propose the following multi-task formulation for gradient learning (MGL):

$$\begin{aligned}\overrightarrow{f_{\mathbf{z}}} &= \arg\min_{\overrightarrow{f}\in\mathcal{H}_{\mathcal{K}}} \Big\{\frac{1}{m^2}\sum_{i,j} w_{ij} L(y_i, \\ &\quad f_1(x_j) + \overrightarrow{f_2}(x_j)(x_i - x_j)^T) + \lambda\|\overrightarrow{f}\|_{\mathcal{K}}^2\Big\}.\end{aligned} \tag{3}$$

where $L : \mathbb{R}\times\mathbb{R} \to [0,\infty)$ is a prescribed loss function and λ is usually called the *regularization parameter*. The minimum is taken over a vector-valued RKHS with multi-task kernel $\mathcal{K}$. The first component $f_{1,\mathbf{z}}$ of the minimizer $\overrightarrow{f_{\mathbf{z}}}$ of the above algorithm is used to simulate the target function and the other components $\overrightarrow{f_{2\mathbf{z}}} := (f_{2,\mathbf{z}}, \ldots, f_{d+1,\mathbf{z}})$ to learn its gradient function. In Section 6, we will discuss how to use the solution $\overrightarrow{f_{\mathbf{z}}}$ for variable selection as well as covariation measurement.

Different choice of loss functions yield different gradient learning algorithms. For instance, if the loss function $L(y,t) = (y-t)^2$ then algorithm (3) leads to the least-square multi-task gradient learning (LSMGL):

$$\begin{aligned}\arg\min_{\overrightarrow{f}\in\mathcal{H}_{\mathcal{K}}} \Big\{\frac{1}{m^2}\sum_{i,j\in\mathbb{N}_m} w_{ij}\big[y_i - f_1(x_j) \\ -\overrightarrow{f_2}(x_j)(x_i - x_j)^T\big]^2 + \lambda\|\overrightarrow{f}\|_{\mathcal{K}}^2\Big\}.\end{aligned} \tag{4}$$

In classification, the choice of loss function $L(y,t) = (1 - yt)_+$ in algorithm (3) yields the support vector machine for multi-task gradient learning (SVMMGL):

[1] Our form of Taylor expansion is slightly different from that used in [15, 17]. However, the essential idea is the same.

$$\arg \min_{\substack{\overrightarrow{f} \in \mathcal{H}_{\mathcal{K}} \\ b \in \mathbb{R}}} \Big\{ \frac{1}{m^2} \sum\nolimits_{i,j \in \mathbb{N}_m} w_{ij} \big[1 - y_i (f_1(x_j) + b + \overrightarrow{f_2}(x_j)(x_i - x_j)^T) \big]_+ + \lambda \| \overrightarrow{f} \|_{\mathcal{K}}^2 \Big\}. \tag{5}$$

Here, $f_1(x) + b$ is used to learn the target function and $\overrightarrow{f_2}$, simulating the gradient of the target function. Hence b plays the same role of offset as in the standard SVM formulation. In this case, at each point x_i the error between the output y_i and $f(x_i)$ is now replaced by the error between y_i and the first order Taylor expansion of $f(x_i)$ at x_j, i.e., $f_1(x_j) + \overrightarrow{f_2}(x_j)(x_i - x_j)^T$.

2.2 Choice of multi-task kernels

We note that if $\mathcal{K}$ is a diagonal matrix-valued kernel, then each component of a vector-valued function in the associated RKHS of $\mathcal{K}$ can be represented, independently of the other components, as a function in the RKHS of a scalar kernel. Consequently, for a scalar kernel G if we choose the multi-task kernel $\mathcal{K}$ given, for any $x, t \in X$, by $\mathcal{K}(x,t) = G(x,t) I_{d+1}$ then the MGL algorithm (3) is reduced to the gradient learning algorithm proposed in [15, 16, 17] using $(d+1)$-folds of scalar RKHS. There, under some conditions on the underlying distribution ρ, it has been proven that $f_{1,\mathbf{z}} \to f_*$ and $\overrightarrow{f_{2\mathbf{z}}} \to \nabla f_*$ when the number of samples tends to infinity. Although their results are promising both theoretically and practically, a more inherent structure would be $\overrightarrow{f_{2\mathbf{z}}} = \nabla f_{1,\mathbf{z}}$. In our MGL framework (3), we can recover this structure by choosing the multi-task kernel appropriately.

Our alternative choice of multi-task kernel is stimulated by the Hessian of Gaussian kernel proposed in [7]. For any scalar kernel G and any $x, t \in X$, we introduce the function

$$\mathcal{K}(x,t) = \begin{pmatrix} G(x,t), & (\nabla_t G(x,t))^T \\ \nabla_x G(x,t) & \nabla_{xt}^2 G(x,t) \end{pmatrix} \tag{6}$$

which we will show to be a multi-task kernel. To see this, let ℓ^2 be the Hilbert space with norm $\|w\|_{\ell^2}^2 = \sum_{j=1}^{\infty} w_j^2$. Suppose that G has a feature representation, i.e., $G(x,t) = \langle \phi(x), \phi(t) \rangle_{\ell^2}$ and, for any $f \in \mathcal{H}_G$, there exits a vector $w \in \ell^2$ such that $f(x) = \langle w, \phi(x) \rangle_{\ell^2}$ and

$$\|f\|_G = \|w\|_{\ell^2}.$$

Indeed, if the input space X is compact and $G : X \times X \to \mathbb{R}$ is a *Mercer kernel*, i.e., it is continuous, symmetric and positive semi-definite, then, according to Mercer theorem, G always has the above feature representation (see e.g. [9]).

Now we have the following proposition about $\mathcal{K}$ defined by equation (6). Let $\widetilde{e}_p$ be the p-th coordinate basis in $\mathbb{R}^{d+1}$.

Theorem 2 *For any smooth scalar Mercer kernel G, define function $\mathcal{K}$ by equation (6). Then, $\mathcal{K}$ is a multi-task kernel and, for any $\overrightarrow{f} = (f_1, \overrightarrow{f_2}) \in \mathcal{H}_{\mathcal{K}}$ there holds*

$$\overrightarrow{f_2} = \nabla f_1. \tag{7}$$

Proof: Since G is a scalar kernel, for any $x, t \in X$ we have that $G(x,t) = G(t,x)$. Therefore, $\mathcal{K}(x,t)^T = \mathcal{K}(t,x)$. Moreover, G is assumed to be a Mercer kernel which implies that it has a feature representation $G(x,t) = \langle \phi(x), \phi(t) \rangle_{\ell^2}$. Consequently, $\nabla_t G(x,t) = (\phi(x), \nabla \phi(t))$ and $\nabla_x G(x,t) = (\nabla \phi(x), \phi(t))$, and $\nabla_{xt}^2 G(x,t) = \langle \nabla \phi(x), \nabla \phi(t) \rangle_{\ell^2}$. Then, we introduce, for any $w \in \ell^2, x \in X, \mathbb{y} \in \mathbb{R}^{d+1}$, the feature map $\Phi(x) : \ell^2 \to \mathbb{R}^{d+1}$ defined by

$$\Phi(x) w := \left(\langle \phi(x), w \rangle_{\ell^2}, \langle \partial_1 \phi(x), w \rangle_{\ell^2}, \ldots, \langle \partial_d \phi(x), w \rangle_{\ell^2} \right)^T.$$

Its adjoint map Φ^* is given, for any $t \in X$ and $\mathbb{y} \in \mathbb{R}^{d+1}$, by $\Phi^*(t)\mathbb{y} := \phi(x)\mathbb{y}^1 + \sum_{p \in \mathbb{N}_d} \partial_p \phi(x) \mathbb{y}^{p+1}$. Hence, $\mathcal{K}(x,t)\mathbb{y} = \Phi(x)\Phi^*(t)\mathbb{y}$. Consequently, for any $m \in \mathbb{N}$, any $i, j \in \mathbb{N}_m$ and $\mathbb{y}_i, \mathbb{y}_j \in \mathbb{R}^{d+1}$, it follows $\sum_{i,j \in \mathbb{N}_m} \langle \mathbb{y}_i, K(x_i, x_j) \mathbb{y}_j \rangle = \| \sum_{i \in \mathbb{N}_m} \Phi^*(x_i) \mathbb{y}_i \|_{\ell^2}^2$ is nonnegative which tells us that $\mathcal{K}$ is a multi-task kernel.

We turn to the second assertion. When $\overrightarrow{f}$ is in the form of a finite combination of *kernel section* $\{\mathcal{K}_x \mathbb{y} : \mathbb{y} \in \mathbb{R}^{d+1}, x \in X\}$, the second assertion follows directly from the definition of $\mathcal{K}$. For the general case, we use the fact that the vector-valued RKHS is the closure of the span of kernel sections, see [14]. To this end, assume that there exists a sequence $\{\overrightarrow{f_j} = (f_1^j, f_2^j, \ldots, f_{d+1}^j)\}$ of finite combination of kernel sections such that $\overrightarrow{f_j} \to \overrightarrow{f} \in \mathcal{H}_{\mathcal{K}}$ w.r.t. the RKHS norm. Hence, by the reproducing property (2), for any $x \in X$ and $p \in \mathbb{N}_d$, $|f_{p+1}^j(x) - f_{p+1}(x)| = |\langle \widetilde{e}_{p+1}, \overrightarrow{f_j}(x) - \overrightarrow{f}(x) \rangle| = |\langle \overrightarrow{f_j} - \overrightarrow{f}, \mathcal{K}_x \widetilde{e}_{p+1} \rangle_{\mathcal{K}}| \le \| \overrightarrow{f_j} - \overrightarrow{f} \|_{\mathcal{K}} \sqrt{\widetilde{e}_{p+1}^T \mathcal{K}(x,x) \widetilde{e}_{p+1}}$ which tends to zeros as j tends to infinity. Consequently, it follows, for any $x \in X$,

$$f_{p+1}^j(x) \to f_{p+1}(x), \quad \text{as } j \to \infty. \tag{8}$$

Let $\delta_p \in \mathbb{R}^d$ be a vector with its p-th component $\delta > 0$ and others equal zero. Applying the reproducing property (2) yields that

$$\begin{aligned} & \left| \tfrac{[f_1^j(x+\delta_p) - f_1^j(x)] - [f_1(x+\delta_p) - f_1(x)]}{\delta} \right| \\ &= \left| \tfrac{1}{\delta} \langle \overrightarrow{f_j} - \overrightarrow{f}, \mathcal{K}_{x+\delta_p} \widetilde{e}_1 - \mathcal{K}_x \widetilde{e}_1 \rangle_{\mathcal{K}} \right| \\ &\le \| \overrightarrow{f_j} - \overrightarrow{f} \|_{\mathcal{K}} \langle \widetilde{e}_1, \tfrac{1}{\delta^2} \big[\mathcal{K}(x+\delta_p, x+\delta_p) \\ &+ \mathcal{K}(x,x) - \mathcal{K}(x, x+\delta_p) - \mathcal{K}(x+\delta_p, x) \big] \widetilde{e}_1 \rangle^{\frac{1}{2}} \\ &= \| \overrightarrow{f_j} - \overrightarrow{f} \|_{\mathcal{K}} \Big(\tfrac{1}{\delta^2} \big[G(x+\delta_p, x+\delta_p) \\ & G(x,x) - G(x, x+\delta_p) - G(x+\delta_p, x) \big] \Big)^{\frac{1}{2}}. \end{aligned}$$

Since G is smooth and X is compact there exists an absolute constant $\widetilde{c} > 0$ such that, for any $\delta > 0$, the above equation is furthermore bounded by

$$\begin{aligned} & \left| \tfrac{[f_1^j(x+\delta_p) - f_1^j(x)] - [f_1(x+\delta_p) - f_1(x)]}{\delta} \right| \\ &\le \widetilde{c} \, \| \overrightarrow{f_j} - \overrightarrow{f} \|_{\mathcal{K}}. \end{aligned}$$

Consequently, letting $\delta \to 0$ in the above equation it follows $|\partial_p f_1^j(x) - \partial_p f_1(x)| \to 0$ as j tends to infinity. Combining this with equation (8) and the fact that $f_{p+1}^j(x) = \partial_p f_1^j(x)$ implies that $\partial_p f_1(x) = f_{p+1}(x)$ which completes the theorem. ■

The scalar kernel G plays the role of a *hyper-parameter* to produce the multi-task kernel $\mathcal{K}$ given by equation (6). By the above theorem, if we choose $\mathcal{K}$ to be defined by equation (6) then any solution $f_{\mathbf{z}} = (f_{1,\mathbf{z}}, \overrightarrow{f_{2\mathbf{z}}})$ of algorithm (3) enjoys the structure $\overrightarrow{f_{2\mathbf{z}}} = \nabla f_{1,\mathbf{z}}$.
Further specifying the kernel G in the definition (6) of multi-task kernel $\mathcal{K}$, we can recover the RFE feature ranking algorithm for a linear SVM [11]. To see this, let G be a linear kernel. In the next section, we will see that, for any solution $\overrightarrow{f_{\mathbf{z}}}$ of MGL algorithm (3), there exists $\{c_{j,\mathbf{z}} \in \mathbb{R}^{d+1} : j \in \mathbb{N}_m\}$ such that $\overrightarrow{f_{\mathbf{z}}} = \sum_{j\in\mathbb{N}_m} \mathcal{K}_{x_j} c_{j,\mathbf{z}}$. Since G is linear, combining this with Theorem 2 we know that $f_{1,\mathbf{z}}(x) = W_{\mathbf{z}}^T x$ with $W_{\mathbf{z}} = \sum_j (x_j^T, 1)c_{j,\mathbf{z}} \in \mathbb{R}$ and $\overrightarrow{f_{2\mathbf{z}}} = \nabla f_{1,\mathbf{z}} = W_{\mathbf{z}}^T$. Consequently, in the case we have that

$$f_{1,\mathbf{z}}(x_j) + \overrightarrow{f_{2\mathbf{z}}}(x_j)(x_i - x_j) = W_{\mathbf{z}}^T x_i = f_{1,\mathbf{z}}(x_i).$$

Moreover, by the reproducing property (2) we can check that

$$\|\overrightarrow{f_{\mathbf{z}}}\|_{\mathcal{K}}^2 = \|f_{1,\mathbf{z}}\|_G^2 = \|W_{\mathbf{z}}\|^2.$$

Putting the above equations together, in this special case we know that the SVMMGL algorithm (5) is reduced, with the choice of $w_{ij} = 1$, to the classical learning algorithm:

$$\min_{W\in\mathbb{R}^d} \Big\{ \frac{1}{m} \sum_{i\in\mathbb{N}_m} (1 - y_i(W^T x_i + b))_+ + \lambda \|W\|^2 \Big\}.$$

Hence, our formulation of gradient learning (3) can be regarded as a generalization of RFE-SVM [11] to the nonlinear case.
In the subsequent sections we discuss a general representation theorem and computational optimization problems motivated by MGL algorithms.

3 Representer theorem

In this section we investigate the representer theorem for the MGL algorithm (3). This forms a foundation for the derivation of a computationally efficient algorithm for MGL in Section 4.
Recall that $\widetilde{e}_p$ is the p-th coordinate basis in $\mathbb{R}^{d+1}$ and, for any $x \in \mathbb{R}^d$, denote the vector $\widetilde{x}^T$ by $(0, x^T)$. By the reproducing property (2), we have that $f_1(x_j) = \langle \overrightarrow{f}(x_j), \widetilde{e}_1 \rangle = \langle \overrightarrow{f}, \mathcal{K}_{x_j}\widetilde{e}_1 \rangle_{\mathcal{K}}$ and likewise, $\overrightarrow{f_2}(x_j)(x_i - x_j)^T = \langle \overrightarrow{f}(x_j), \widetilde{x}_i - \widetilde{x}_j \rangle = \langle \overrightarrow{f}, \mathcal{K}_{x_j}(\widetilde{x}_i - \widetilde{x}_j) \rangle_{\mathcal{K}}$. Then, the algorithm (3) can be rewritten by

$$\begin{aligned} \arg\min_{\overrightarrow{f}\in\mathcal{H}_{\mathcal{K}}} \quad & \Big\{ \tfrac{1}{m^2} \sum\nolimits_{i,j\in\mathbb{N}_m} w_{ij} L\big(y_i, \langle \overrightarrow{f}, \\ & \mathcal{K}_{x_j}(\widetilde{e}_1 + \widetilde{x}_i - \widetilde{x}_j) \rangle_{\mathcal{K}}\big) + \lambda \|\overrightarrow{f}\|_{\mathcal{K}}^2 \Big\}. \end{aligned} \tag{9}$$

In analogy with standard kernel methods [19, 20], we have the following representer theorem for MGL by using the properties of multi-task kernels.

Theorem 3 *For any multi-task kernel $\mathcal{K}$, consider the gradient learning algorithm (3). Then, there exists representer coefficients $\{c_{j,\mathbf{z}} \in \mathbb{R}^{d+1} : j \in \mathbb{N}_m\}$ such that*

$$\overrightarrow{f_{\mathbf{z}}} = \sum_{j\in\mathbb{N}_m} \mathcal{K}_{x_j} c_{j,\mathbf{z}}$$

and, for every $j \in \mathbb{N}_m$, the representer coefficient $c_{j,\mathbf{z}} \in$ span$\{\widetilde{e}_1, \widetilde{x}_i : i \in \mathbb{N}_m\}$.

Proof: We can write any minimizer $\overrightarrow{f_{\mathbf{z}}} \in \mathcal{H}_{\mathcal{K}}$ as $\overrightarrow{f_{\mathbf{z}}} = \overrightarrow{f}_{\parallel} + \overrightarrow{f}_{\perp}$ where $\overrightarrow{f}_{\parallel}$ is in the span $\{\mathcal{K}_{x_j}\widetilde{e}_1, \mathcal{K}_{x_j}\widetilde{x}_i, i, j \in \mathbb{N}_m\}$ and $\overrightarrow{f}_{\perp}$ is perpendicular to this span space. By the reproducing property (2), we have that $\langle \overrightarrow{f}(x_j), \widetilde{e}_1 + \widetilde{x}_i - \widetilde{x}_j \rangle = \langle \overrightarrow{f}, \mathcal{K}_{x_j}(\widetilde{e}_1 + \widetilde{x}_i - \widetilde{x}_j) \rangle_{\mathcal{K}} = \langle \overrightarrow{f}_{\parallel}, \mathcal{K}_{x_j}(\widetilde{e}_1 + \widetilde{x}_i - \widetilde{x}_j) \rangle_{\mathcal{K}}$. Hence, $\overrightarrow{f}_{\perp}$ makes no contribution to the loss function in the MGL algorithm (9) (i.e. algorithm (3)). However, the norm $\|\overrightarrow{f}\|_{\mathcal{K}}^2 = \|\overrightarrow{f}_{\parallel}\|_{\mathcal{K}}^2 + \|\overrightarrow{f}_{\perp}\|_{\mathcal{K}}^2 > \|f_{\parallel}\|_{\mathcal{K}}^2$ unless $f_{\perp} = 0$. This implies, any solution $\overrightarrow{f_{\mathbf{z}}}$ belongs to the span space $\{\mathcal{K}_{x_j}\widetilde{e}_1, \mathcal{K}_{x_j}\widetilde{x}_i, i, j \in \mathbb{N}_m\}$ and the corresponding representer coefficients belong to the span of $\{\widetilde{e}_1, \widetilde{x}_i : i \in \mathbb{N}_m\}$. ■

The representer theorem above tells us that the optimal solution $\overrightarrow{f_{\mathbf{z}}}$ of algorithm (3) lives in the finite span of training samples which paves the way for designing efficient optimization algorithms for multi-task gradient learning.

4 Optimization and solution

In this section, by the above representer theorem, we explore efficient algorithms for computing the representer coefficients. For clarity, we mainly focus on least-square multi-task gradient learning algorithms (LSMGL). At the end of this section, the support vector machine for gradient learning (SVMMGL) in classification will be briefly discussed. One can apply the subsequent procedures to other loss functions.

4.1 Computation of representer coefficients

To specify the solution of LSMGL, we denote the column vector $C_{\mathbf{z}} \in \mathbb{R}^{m(d+1)}$ by consecutively catenating all column vectors $\{c_{j,\mathbf{z}} \in \mathbb{R}^{d+1} : j \in \mathbb{N}_m\}$ and, likewise we define a column vector $\mathbb{Y} \in \mathbb{R}^{m(d+1)}$ by catenating column vectors $\{\mathbb{y}_i \in \mathbb{R}^{d+1} : i \in \mathbb{N}_m\}$. Moreover, we introduce an $m(d+1)\times m(d+1)$ matrix by catenating all $(d+1)\times(d+1)$ matrix $\mathcal{K}(x_i, x_j)$ denoted by

$$\mathcal{K}_{\mathbf{x}} = \big(\mathcal{K}(x_i, x_j)\big)_{i,j\in\mathbb{N}_m}.$$

Finally, we introduce a system of equations

$$m^2\lambda c_j + B_j \sum\nolimits_{l\in\mathbb{N}_m} \mathcal{K}(x_j, x_l)c_l = \mathbb{y}_j, \ \ \forall j \in \mathbb{N}_m \tag{10}$$

where $B_j = \sum_{i\in\mathbb{N}_m} w_{ij}(\widetilde{e}_1 + \widetilde{x}_i - \widetilde{x}_j)(\widetilde{e}_1 + \widetilde{x}_i - \widetilde{x}_j)^T$, $\mathbb{y}_j = \sum_{i\in\mathbb{N}_m} w_{ij} y_i (\widetilde{e}_1 + \widetilde{x}_i - \widetilde{x}_j)$.
We now can solve the LSMGL algorithm by the following theorem.

Theorem 4 *For any $j \in \mathbb{N}_m$, the vectors $B_j, \mathbb{y}_j$ be defined by equation (10). Then, the representer coefficients $C_{\mathbf{z}}$ for the solution of the LSMGL algorithm are given by the following equation*

$$\mathbb{Y} = \Big(m^2\lambda I_{m(d+1)} + diag\big(B_1, \ldots, B_m\big)\mathcal{K}_{\mathbf{x}}\Big)_{i,j=1}^m C_{\mathbf{z}}. \tag{11}$$

Proof: By Theorem 3, there exists $\{c_{j,\mathbf{z}} \in \mathbb{R}^{d+1} : j \in \mathbb{N}_m\}$ such that $\overrightarrow{f_{\mathbf{z}}} = \sum_{j\in\mathbb{N}_m} \mathcal{K}_{x_j} c_{j,\mathbf{z}}$. However, taking the

functional derivative of algorithm (3) with respective to f yields that $\frac{1}{m^2}\sum_{i,j\in\mathbb{N}_m} w_{ij}\big(\langle\overrightarrow{f_{\mathbf{z}}},\mathcal{K}_{x_j}(\widetilde{e}_1+\widetilde{x}_i-\widetilde{x}_j)\rangle_{\mathcal{K}}-y_i\big)\mathcal{K}_{x_j}(\widetilde{e}_1+\widetilde{x}_i-\widetilde{x}_j)+\lambda\overrightarrow{f_{\mathbf{z}}}=0$ which means that $c_{j,\mathbf{z}}=\frac{1}{m^2\lambda}\sum_{i\in\mathbb{N}_m} w_{ij}(y_i-\langle\overrightarrow{f_{\mathbf{z}}},\mathcal{K}_{x_j}(\widetilde{e}_1+\widetilde{x}_i-\widetilde{x}_j)\rangle_{\mathcal{K}})(\widetilde{e}_1+\widetilde{x}_i-\widetilde{x}_j)$. Equivalently, equation (10) holds true, and hence completes the assertion. ■

Solving equation (11) involves the inversion of an $m(d+1)\times m(d+1)$ matrix whose time complexity is usually $O((md)^3)$. However, it is computationally prohibitive since the coordinate (feature) dimension d is very large in many applications. Fortunately, as suggested in Theorem 3, the representer coefficients $\{c_{j,\mathbf{z}}: j\in\mathbb{N}_m\}$ can be represented by the span of column vectors of matrix

$$\widetilde{M_{\mathbf{x}}}=\{\widetilde{e}_1,\widetilde{x}_1,\ldots,\widetilde{x}_{m-1},\widetilde{x}_m\}.$$

This observation suggests the possibility of reduction of the original high dimensional problem in $\mathbb{R}^{d+1}$ to the low dimensional space spanned by $\widetilde{M_{\mathbf{x}}}$. This low dimensional space can naturally be introduced by singular vectors of $\widetilde{M_{\mathbf{x}}}$.

To this end, we consider the representation of the matrix $\widetilde{M_{\mathbf{x}}}$ by its singular vectors. It will be proven to be useful to represent matrix $\widetilde{M_{\mathbf{x}}}$ from the singular value decomposition (SVD) of the data matrix defined by

$$M_{\mathbf{x}}=\big[x_1,x_2,\ldots,x_{m-1},x_m\big].$$

Apparently, the rank s of $M_{\mathbf{x}}$ is at most $\min(m,d)$. The SVD of $M_{\mathbf{x}}$ tells us that there exists orthogonal matrices $V_{d\times d}$ and $U_{m\times m}$ such that

$$\begin{aligned} M_{\mathbf{x}} &= [V_1,\ldots,V_d]\,\Sigma\begin{bmatrix} U_1\\ \vdots\\ U_m\end{bmatrix}^T\\ &= [V_1,\ldots,V_s]\,(\beta_1,\ldots,\beta_m)\end{aligned}\tag{12}$$

Here, the $d\times m$ matrix $\Sigma=\begin{bmatrix}\operatorname{diag}\{\sigma_1,\ldots,\sigma_s\} & 0\\ 0 & 0\end{bmatrix}$. For any $j\in\mathbb{N}_m$, we use the notation $U_j=(U_{1j},\ldots,U_{mj})$ and $\beta_j^T=(\sigma_1U_{1j},\sigma_2U_{2j},\ldots,\sigma_sU_{sj})\in\mathbb{R}^s$. From now on we also denote

$$\mathcal{V}=[V_1,\ldots,V_s]\tag{13}$$

Hence, we have, for any $j\in\mathbb{N}_m$, that $x_j=\mathcal{V}\beta_j$.

We are now ready to specify the representation of $\widetilde{M_{\mathbf{x}}}$ from the above SVD of $M_{\mathbf{x}}$, To see this, for any $l\in\mathbb{N}_s$ and $j\in\mathbb{N}_m$, let $\widetilde{V_l}^T=(0,V_l^T)$, $\widetilde{\beta}_j^T=(0,\beta_j^T)$. In addition, we introduce the $(d+1)\times(s+1)$ matrix

$$\widetilde{\mathcal{V}}=\begin{pmatrix}1 & 0\\ 0 & \mathcal{V}\end{pmatrix}=\big[\widetilde{e}_1,\widetilde{V}_1,\ldots,\widetilde{V}_s\big]\tag{14}$$

which induces a one-to-one mapping $\widetilde{\mathcal{V}}:\mathbb{R}^{s+1}\to\mathbb{R}^{d+1}$ defined, for any $\beta\in\mathbb{R}^{s+1}$, by $x=\widetilde{\mathcal{V}}\beta\in\mathbb{R}^{d+1}$ since column vectors in $\widetilde{\mathcal{V}}$ are orthogonal to each other. Consequently, it follows that

$$\widetilde{M_{\mathbf{x}}}=\widetilde{\mathcal{V}}\,[e_1,\widetilde{\beta}_1,\ldots,\widetilde{\beta}_m],$$

where e_1 is the standard first coordinate basis in $\mathbb{R}^{s+1}$. Equivalently, for any $i,j\in\mathbb{N}_m$,

$$\widetilde{e}_1=\widetilde{\mathcal{V}}e_1,\quad \widetilde{x}_j=\widetilde{\mathcal{V}}\widetilde{\beta}_j.\tag{15}$$

We now assemble all material to state the reduced system associated with equation (11). For this purpose, firstly we define the kernel $\widetilde{\mathcal{K}}$, for any $x,t\in X$, by

$$\widetilde{\mathcal{K}}(x,t):=\widetilde{\mathcal{V}}^T\mathcal{K}(x,t)\widetilde{\mathcal{V}},$$

and introduce the $m(s+1)\times m(s+1)$ matrix by catenating all $(s+1)\times(s+1)$ matrices $\widetilde{\mathcal{K}}(x_i,x_j)$:

$$\widetilde{\mathcal{K}}_{\mathbf{x}}=\big(\widetilde{\mathcal{K}}(x_i,x_j)\big)_{i,j\in\mathbb{N}_m}.\tag{16}$$

Secondly, for any $j\in\mathbb{N}_m$, set $\mathscr{B}_j=\sum_{i\in\mathbb{N}_m} w_{ij}(e_1+\widetilde{\beta}_i-\widetilde{\beta}_j)(e_1+\widetilde{\beta}_i-\widetilde{\beta}_j)^T$ and $\mathscr{Y}_j=\sum_{i\in\mathbb{N}_m} w_{ij}y_i(e_1+\widetilde{\beta}_i-\widetilde{\beta}_j)$. Thirdly, associated with the system (10), for any $j\in\mathbb{N}_m$ and $\gamma_j\in\mathbb{R}^{s+1}$, we define the system in reduced low dimensional space $\mathbb{R}^{s+1}$

$$m^2\lambda\gamma_j+\mathscr{B}_j\sum_{l\in\mathbb{N}_m}\widetilde{\mathcal{K}}(x_j,x_l)\gamma_l=\mathscr{Y}_j.\tag{17}$$

Finally, in analogy with the notation $\mathbb{Y}$, the column vector $\mathscr{Y}\in\mathbb{R}^{m(s+1)}$ is defined by successively catenating column vectors $\{\mathscr{Y}_i\in\mathbb{R}^{s+1}: i\in\mathbb{N}_m\}$. Likewise we can define $\gamma_{\mathbf{z}}$ by catenating column vectors $\{\gamma_{j,\mathbf{z}}\in\mathbb{R}^{s+1}: j\in\mathbb{N}_m\}$. With the above preparation we have the following result.

Theorem 5 *If the $\{\gamma_{j,\mathbf{z}}\in\mathbb{R}^{s+1}: j\in\mathbb{N}_m\}$ is the solution of system (17), i.e.,*

$$\mathscr{Y}=\Big(m^2\lambda I_{m(s+1)}+\mathit{diag}\big(\mathscr{B}_1,\ldots,\mathscr{B}_m\big)\widetilde{\mathcal{K}}_{\mathbf{x}}\Big)\gamma,\tag{18}$$

then the coefficient $C_{\mathbf{z}}$ defined, for any $j\in\mathbb{N}_m$, by $c_{j,\mathbf{z}}=\widetilde{\mathcal{V}}\gamma_{j,\mathbf{z}}$ is one of the solution of system (11), and thus yields representation coefficients of the solution $\overrightarrow{f}_{\mathbf{z}}$ for the LSMGL algorithm (3).

Proof: Let $\gamma_{\mathbf{z}}$ is the solution of system (17). Since $\widetilde{\mathcal{V}}$ is orthogonal, the system (17) is equivalent to the following equation

$$m^2\lambda\widetilde{\mathcal{V}}\gamma_{j\mathbf{z}}+\widetilde{\mathcal{V}}\mathscr{B}_j\sum_{l\in\mathbb{N}_m}\widetilde{\mathcal{K}}(x_j,x_l)\gamma_{l\mathbf{z}}=\widetilde{\mathcal{V}}\mathscr{Y}_j.$$

Recall, for any $j\in\mathbb{N}_m$, that $B_j=\widetilde{\mathcal{V}}\mathscr{B}_j\widetilde{\mathcal{V}}^T$, $\widetilde{\mathcal{V}}e_1e_1\widetilde{\mathcal{V}}^T=\widetilde{e}_1\widetilde{e}_1^T$, $\widetilde{\mathcal{V}}^T\widetilde{\mathcal{V}}=I_{s+1}$, and $Y_j=\widetilde{\mathcal{V}}\mathscr{Y}_j$. Hence, the above system is identical to system (11) (i.e. system (10)) with C_j replaced by $\widetilde{\mathcal{V}}\gamma_{j,\mathbf{z}}$ which completes the assertion. ■

We end this subsection with a brief discussion of the solution of the SVMMGL algorithm. Since the hinge loss is not differentiable, we cannot use the above techniques to derive an optimization algorithm for SVMMGL. Instead, we can consider its dual problem. To this end, we introduce slack variables $\{\xi_{ij}: i,j\in\mathbb{N}_m\}$ and rewrite SVMMGL as follows:

$$\begin{cases}\arg\quad \min\limits_{\overrightarrow{f},\xi,b}\Big\{\dfrac{1}{m^2}\displaystyle\sum_{i,j\in\mathbb{N}_m} w_{ij}\xi_{ij}+\lambda\|\overrightarrow{f}\|_{\mathcal{K}}^2\Big\}\\ \text{s.t.}\quad y_i(\langle\overrightarrow{f},\mathcal{K}_{x_j}(\widetilde{e}_1+\widetilde{x}_i-\widetilde{x}_j)\rangle_{\mathcal{K}}+b)\geq 1-\xi_{ij},\\ \qquad \xi_{ij}\geq 0,\quad \forall i,j\in\mathbb{N}_m.\end{cases}\tag{19}$$

Given a multi-task kernel $\mathcal{K}$ produced by scalar kernel G, $\lambda > 0$ and inputs $\{(x_i, y_i) : i \in \mathbb{N}_m\}$

1. Compute projection mapping $\widetilde{\mathcal{V}}$ and reduced vector $\widetilde{\beta}_j$ from equations (12), (13), and (14).
2. Compute $\widetilde{\mathcal{K}}(x_j, x_i) = \widetilde{\mathcal{V}}^T \mathcal{K}(x_j, x_i)\widetilde{\mathcal{V}}$ (i.e. equation (16))
3. Solving equation (18) to get coefficient $\gamma_{\mathbf{z}}$, see Theorem 5 (equivalently, equation (25) when G is linear or RBF kernel, see Theorem 6).
4. Output vector-valued function: $\overrightarrow{f_{\mathbf{z}}}(\cdot) = \sum_{j\in\mathbb{N}_m} \mathcal{K}(\cdot, x_j)(\widetilde{\mathcal{V}}\gamma_{j,\mathbf{z}})$.
5. Compute variable covariance and ranking variables using Proposition 1 in Section 6.

Table 1: Pseudo-code for least square multi-task gradient learning

Parallel to the derivation of the dual problem of standard SVM (e.g. [20, 23]), using Lagrangian theory we can obtain the following dual problem of SVMMGL:

$$\begin{cases} \arg\max\limits_{\alpha} \sum\limits_{i,j\in\mathbb{N}_m} \alpha_{ij} - \dfrac{1}{4m^2\lambda} \sum\limits_{i,j,i',j'\in\mathbb{N}_m} \alpha_{ij} y_i \alpha_{i'j'} y_{i'} \\ \times \left[(\widetilde{e}_1 + \widetilde{x}_i - \widetilde{x}_j)^T \mathcal{K}(x_j, x_{j'})(\widetilde{e}_1 + \widetilde{x}_{i'} - \widetilde{x}_{j'})\right] \\ \text{s.t. } \sum_{i,j\in\mathbb{N}_m} y_i \alpha_{ij} = 0,\ 0 \le \alpha_{ij} \le w_{ij},\ \ \forall i, j \in \mathbb{N}_m. \end{cases} \tag{20}$$

Moreover, if the solution of dual problem is $\alpha_{\mathbf{z}} = \{\alpha_{ij,\mathbf{z}} : i, j \in \mathbb{N}_m\}$ then the solution of SVMMGL can be represented by

$$\overrightarrow{f_{\mathbf{z}}} = \frac{1}{2m^2\lambda} \sum_{i,j\in\mathbb{N}_m} y_i \alpha_{ij,\mathbf{z}} \mathcal{K}_{x_j}(\widetilde{e}_1 + \widetilde{x}_i - \widetilde{x}_j).$$

Note that

$$\left((\widetilde{e}_1 + \widetilde{x}_i - \widetilde{x}_j)^T \mathcal{K}(x_j, x_{j'})(\widetilde{e}_1 + \widetilde{x}_{i'} - \widetilde{x}_{j'})\right)_{(i,j),(i',j')}$$

is a scalar kernel matrix with double indices (i, j) and (i', j'). Then, when the number of samples is small the dual problem of SVMMGL can be efficiently solved by quadratic programming with $\alpha \in \mathbb{R}^{m^2}$.

4.2 Further low dimensional formulation

Consider the multi-task kernel $\mathcal{K}$ defined by equation (6) with scalar kernel G. In this section, by further specifying G we show that LSMGL algorithm (4) with input/output $\{(x_i, y_i) : i \in \mathbb{N}_m\}$ can be reduced to its low dimensional formulation with input/output $\{(\beta_i, y_i) : i \in \mathbb{N}_m\}$, where β_j is defined by equation (12). This clarification will provide a more computationally efficient algorithm.

To this end, consider the scalar kernel G defined on $\mathbb{R}^d \times \mathbb{R}^d$. By definition of kernels (called restriction theorem in [3]), we can see that G is also a reproducing kernel on $\mathbb{R}^s \times \mathbb{R}^s$. Hence, $\mathcal{K}$ defined by equation (6) on $\mathbb{R}^d$ is also a multi-task kernel on the underlying space $\mathbb{R}^s$; we use the same notation $\mathcal{K}$ when no confusion arises. Therefore, associated with the LSMGL algorithm (4) in $\mathbb{R}^d$ we have an LSMGL in low dimensional input space $\mathbb{R}^s$:

$$\begin{aligned} \overrightarrow{g_{\mathbf{z}}} &= \arg\min_{\overrightarrow{g}\in\mathcal{H}_{\mathcal{K}}} \Big\{ \tfrac{1}{m^2} \sum\nolimits_{i,j} w_{ij} \big[y_i - \\ &\quad g_1(\beta_j) - \overrightarrow{g_2}(\beta_j)(\beta_i - \beta_j)^T\big]^2 + \lambda \|\overrightarrow{g}\|^2_{\mathcal{K}} \Big\}. \end{aligned} \tag{21}$$

In analogy with the derivation of the system (10), for any $j \in \mathbb{N}_m$ and $\gamma_j \in \mathbb{R}^{s+1}$, we know that the representer coefficients of the LSMGL algorithm (21) in reduced low dimensional space $\mathbb{R}^{s+1}$ satisfy that

$$m^2\lambda\gamma_j + \mathscr{B}_j \sum_{l\in\mathbb{N}_m} \mathcal{K}(\beta_j, \beta_l)\gamma_l = \mathscr{Y}_j. \tag{22}$$

We are now ready to discuss the relation between representer coefficients of the LSMGL algorithm (4) and those of reduced LSMGL algorithm (21). For this purpose, let G to satisfy, for any $d \times s$ matrix V, that $V^TV = I_s$ and $\beta, \beta' \in \mathbb{R}^s$, that

$$G(V\beta, V\beta') = G(\beta, \beta'). \tag{23}$$

There exists abundant functions G satisfying the above property. For instance, linear product kernel $G(x, t) = x^T t$, Gaussian kernel $G(x, t) = e^{-\|x-t\|^2/2\sigma}$, and sigmoid kernel $G(x, t) = \tanh(ax^T t + r)$ with parameters $a, r \in \mathbb{R}$. More generally, kernel G satisfies property (23) if it is produced by a radial basis function (RBF) $h : (0, \infty) \to \mathbb{R}$ defined, for any $x, t \in X$, by

$$G(x, t) = h(\|x - t\|^2). \tag{24}$$

We say a function $h : (0, \infty) \to \mathbb{R}$ is *complete monotone* if it is smooth and, for any $r > 0$ and $k \in \mathbb{N}$, $(-1)^k f^{(k)}(r) \ge 0$. Here $h^{(k)}$ denotes the k-th derivative of h. According to the well-known Schoenberg's theorem [18], if h is complete monotone then the function G defined by equation (24) is positive semi-definite, and hence becomes a scalar kernel. For instance, the choice of $h(t) = e^{\frac{-\|t\|}{2\sigma}}$ with standard deviation $\sigma > 0$ and $h(t) = (\sigma^2 + \|x - t\|^2)^{-\alpha}$ with parameter $\alpha > 0$ yield Laplacian kernel and inverse polynomial kernel respectively.

Now we are in a position to summarize the reduction theorem for multi-task kernels (6) produced by scalar kernels G satisfying (23). Here we also use the convention that $\mathcal{K}_\beta = \big(\mathcal{K}(\beta_i, \beta_j)\big)_{i,j\in\mathbb{N}_m}$.

Theorem 6 *Let G have the property (23) and $\mathcal{K}$ be defined by equation (6). Suppose $\{\gamma_{j,\mathbf{z}} : j \in \mathbb{N}_m\}$ are the representer coefficients of algorithm (21), i.e., $\gamma_{\mathbf{z}}$ solves the equation*

$$\mathscr{Y} = \Big(m^2\lambda I_{m(s+1)} + diag\big(\mathscr{B}_1, \ldots, \mathscr{B}_m\big)\mathcal{K}_\beta\Big)\gamma_{\mathbf{z}}, \tag{25}$$

Then, the representer coefficients $\{c_{j,\mathbf{z}} : j \in \mathbb{N}_m\}$ of algorithm (4) are given by

$$c_{j,\mathbf{z}} = \widetilde{\mathcal{V}}\gamma_{j,\mathbf{z}}.$$

Proof: Suppose the multi-task kernel $\mathcal{K}$ is produced by G with property (23). Recall that $\mathcal{V}^T\mathcal{V} = I_s$ with $\mathcal{V}$ given by (14). Then, kernel $\mathcal{K}$ satisfies, for any $x = \mathcal{V}\beta$ and $t = \mathcal{V}\beta'$ with $\mathcal{V}$, that

$$\begin{aligned}&\mathcal{K}(x,t) = \mathcal{K}(\mathcal{V}\beta_i, \mathcal{V}\beta_j)\\ &= \begin{pmatrix} G(\beta,\beta'), & (\mathcal{V}\nabla_{\beta_i} G(\beta_i,\beta_j))^T \\ \mathcal{V}\nabla_{\beta_j} G(\beta_i,\beta_j), & \mathcal{V}(\nabla_{\beta_i}\nabla_{\beta_j} G(\beta_i,\beta_j))\mathcal{V}^T \end{pmatrix}.\end{aligned}$$

Hence, it follows, for any $x_i, x_j \in X$ and $i, j \in \mathbb{N}_m$, that

$$\widetilde{\mathcal{K}}(x_i,x_j) = \begin{pmatrix} 1 & 0 \\ 0 & \mathcal{V} \end{pmatrix}^T \mathcal{K}(x_i,x_j) \begin{pmatrix} 1 & 0 \\ 0 & \mathcal{V} \end{pmatrix} = \mathcal{K}(\beta_i,\beta_j).$$

Therefore, the system (17) is identical to the system (22). Consequently, the desired assertion follows directly from Theorem 5. ■

Equipped with Theorem 6, the time complexity and computer memory can be further reduced by directly computing $m(s+1) \times m(s+1)$ matrix $\mathcal{K}_\beta$ instead of first computing $m(d+1) \times m(d+1)$ matrix $\mathcal{K}_{\mathbf{x}}$ and then $\widetilde{\mathcal{K}}_{\mathbf{x}}$ in Theorem 5. Theorem 6 also gives an appealing insight into multi-task gradient learning framework. Roughly speaking, learning gradient in the high dimensional space is equivalent to learning them in the low dimensional projection space spanned by the input data.

5 Statistical error analysis

In this section we give an error analysis for least square MGL algorithms. Our target is to show that the learned vector-valued function from our algorithm statistically converges to the true function and true gradient.

For the least square loss, denote by $\rho_X(\cdot)$ the marginal distribution on X and, for any $x \in X$, let $\rho(\cdot|x)$ to be the conditional distribution on Y. Then, the target function is the regression function f_ρ minimizing the generalization error

$$\mathcal{E}(f) = \int_Z (y - f(x))^2 d\rho(x,y).$$

Specifically, the regression function is defined, for any $x \in X$, by

$$f_\rho(x) = \arg\min_{t\in\mathbb{R}} \int_Y (y-t)^2 \rho(y|x) = \int_Y y d\rho(y|x).$$

Hence, in this case the purpose of error analysis is to show that solution $\overrightarrow{f_{\mathbf{z}}}$ of LSMGL algorithm (4) statistically converges to $\overrightarrow{f_\rho} = (f_\rho, \nabla f_\rho)$ as $m \to \infty$, $s = s(m) \to 0$ and $\lambda = \lambda(m) \to 0$.

To this end, we introduce some notations and the following hypotheses are assumed to be true throughout this section. Firstly, we assume that $Y \subseteq [-M, M]$ with $M > 0$. Since X is compact the diameter of X denoted by $D = \sup_{x,u\in X} \|x-u\|^2$ is finite. Secondly, denote by $L^2_{\rho_X}$ the space of square integral functions $\overrightarrow{f} : X \to \mathbb{R}^{d+1}$ with norm $\|\overrightarrow{f}\|^2_{\rho_X} = \int_X \|\overrightarrow{f}(x)\|^2 d\rho_X(x)$. Finally, denote the boundary of X by ∂X. We assume, for some constant $c_\rho > 0$, that the marginal distribution satisfies that

$$\rho_X\Big(x \in X : \text{dist}(x, \partial X) < s\Big) \le c_\rho s, \ \ \forall 0 < s < D. \quad (26)$$

and, for some parameter $0 < \theta \le 1$, the density function $p(x)$ of ρ_X satisfies θ-Hölder continuous condition, i.e., for any $x, u \in X$ there holds

$$|p(x) - p(u)| \le c_\rho \|x-u\|^\theta, \quad \forall x, u \in X. \quad (27)$$

Of course, p is a bounded function on X since it is continuous and X is compact. For instance, if the boundary of X is piecewise smooth and ρ_X is the uniform distribution over X then the marginal distribution ρ_X satisfies conditions (26) and (27) with parameter $\theta = 1$.

We are ready to present our statistical error analysis of LSMGL algorithms. Recall here we used the notation $\overrightarrow{f_\rho} = (f_\rho, \nabla f_\rho)$.

Theorem 7 *Suppose that the marginal distribution ρ_X satisfies (26) and (27). For any multi-task kernel $\mathcal{K}$, let $\overrightarrow{f_{\mathbf{z}}}$ be the solution of LSMGL algorithm. If $\overrightarrow{f_\rho} \in \mathcal{H}_\mathcal{K}$ then there exists a constant c such that, for any $m \in \mathbb{N}$, with the choice of $\lambda = s^{2\theta}$ and $s = m^{-\frac{1}{3(d+2)+4\theta}}$, there holds*

$$\mathbb{E}\big[\|\overrightarrow{f_{\mathbf{z}}} - \overrightarrow{f_\rho}\|^2_{\rho_X}\big] \le c m^{-\frac{\theta}{3(d+2)+4\theta}}.$$

If moreover ρ_X is a uniform distribution then, choosing $\lambda = s^\theta$ and $s = m^{-\frac{1}{3(d+2)+5\theta}}$, there holds

$$\mathbb{E}\big[\|\overrightarrow{f_{\mathbf{z}}} - \overrightarrow{f_\rho}\|^2_{\rho_X}\big] \le c m^{-\frac{\theta}{3(d+2+\theta)}}.$$

The proof of this theorem needs several steps which are postponed to the appendix. More accurate error rates in terms of probability inequality are possible using techniques in [17, 15]. It would also be interesting to extend this theorem to other loss functions such as the SVMMGL algorithm.

6 Experimental validation

In this section we will only preliminarily validate the MGL algorithm (3) on the problem of variable selection and covariance measurement.

By the representer Theorem 3 in Section 3, the solution of MGL denoted by $\overrightarrow{f_{\mathbf{z}}} = (f_{1,\mathbf{z}}, \overrightarrow{f_{2\mathbf{z}}}) = (f_{1,\mathbf{z}}, f_{2,\mathbf{z}}, \ldots, f_{d+1,\mathbf{z}})$ can be rewritten as $\overrightarrow{f_{\mathbf{z}}} = \sum_{j\in\mathbb{N}_m} \mathcal{K}_{x_j} c_{j,\mathbf{z}}$. Since it only belongs to a vector-valued RKHS $\mathcal{H}_\mathcal{K}$, we need to find a common *criterion inner product (norm)* $\langle\cdot,\cdot\rangle_r$ to measure each component of the learned gradient $\overrightarrow{f_{2\mathbf{z}}} = (f_{2,\mathbf{z}}, \ldots, f_{d+1,\mathbf{z}})$. Once we find the criterion inner product $\langle\cdot,\cdot\rangle_r$, we can use the coordinate covariance

$$\text{Cov}(\overrightarrow{f_{2\mathbf{z}}}) = \Big(\langle f_{p+1,\mathbf{z}}, f_{q+1,\mathbf{z}}\rangle_r\Big)_{p,q\in\mathbb{N}_d} \quad (28)$$

to measure how the variables covary. Also, the variable (feature) ranking can be done according to the following relative magnitude of norm of each component of $\overrightarrow{f_{2\mathbf{z}}}$:

$$s_p = \frac{\|f_{p+1,\mathbf{z}}\|_r}{(\sum_{q\in\mathbb{N}_d} \|f_{q+1,\mathbf{z}}\|_r^2)^{1/2}}. \quad (29)$$

If the scalar kernel G is a linear kernel then every component of $\overrightarrow{f_{2\mathbf{z}}}$ is a constant. In this case, we can choose the standard Euclidean inner product to be the criterion inner product (norm). When the kernel G is an RBF kernel, we show in the following proposition that we can select the criterion inner product $\langle\cdot,\cdot\rangle_r$ to be the RKHS inner product $\langle\cdot,\cdot\rangle_G$ in $\mathcal{H}_G$. The computation is summarized in the following proposition.

Proposition 1 *Suppose the scalar kernel G has a feature representation and the multi-task kernel $\mathcal{K}$ is defined by equation (6). Then, for any solution $\overrightarrow{f_{\mathbf{z}}} = \sum_{j\in\mathbb{N}_m} \mathcal{K}_{x_j} c_{j,\mathbf{z}} \in \mathcal{H}_{\mathcal{K}}$ of MGL algorithm (3), the following hold true.*
1. If G is a linear kernel then the coordinate covariance is defined by

$$Cov(\overrightarrow{f_{2\mathbf{z}}}) = \overrightarrow{f_{2\mathbf{z}}}^T \overrightarrow{f_{2\mathbf{z}}} = \sum_{i,j\in\mathbb{N}_m} (x_i, I_d) c_{i,\mathbf{z}} c_{j,\mathbf{z}}^T (x_j, I_d)^T.$$

Moreover, for LSMGL algorithm the above equation can be more efficiently computed by

$$Cov(\overrightarrow{f_{2\mathbf{z}}}) = \mathcal{V}\Big[\sum_{i,j\in\mathbb{N}_m} (\beta_i, I_s)\gamma_i\gamma_j^T(\beta_j, I_s)^T\Big]\mathcal{V}^T.$$

2. If G is a smooth RBF kernel then $f_{p+1,\mathbf{z}} \in \mathcal{H}_G$ and the coordinate covariance $Cov(\overrightarrow{f_{2\mathbf{z}}}) = \big(\langle (f_{p+1,\mathbf{z}}, f_{q+1,\mathbf{z}}\rangle_G\big)_{p,q\in\mathbb{N}_d}$ can be computed by

$$\langle f_{p+1,\mathbf{z}}, f_{q+1,\mathbf{z}}\rangle_G = C_{\mathbf{z}}^T(\mathbb{K}_{pq}(x_i - x_j))_{i,j=1}^m C_{\mathbf{z}}, \quad (30)$$

where the kernel matrix $\mathbb{K}_{pq}(x_i - x_j)$ defined, for any $i, j \in \mathbb{N}_m$, by

$$\begin{pmatrix} -(\partial_{pq}^2 G)(x_i - x_j), & ((\nabla\partial_{pq}^2 G)(x_i - x_j))^T \\ -(\nabla\partial_{pq}^2 G)(x_i - x_j), & (\nabla^2\partial_{pq}^2 G)(x_i - x_j) \end{pmatrix}.$$

The proof is postponed to the appendix where the computation of $\mathbb{K}_{pq}$ is also given if G is a Gaussian.
We run our experiment on two artificial datasets and one gene expression dataset following [17]. In the first experiment, the target function $f_\rho : \mathbb{R}^d \to \mathbb{R}$ with notation $x = (x^1, \ldots, x^d) \in \mathbb{R}^d$ and $d = 80$. The output y is contaminated by a Gaussian noise

$$y = f_\rho(x) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma_y).$$

As depicted in Figure 1 (leftmost), the thirty inputs whose relevant features are $[1, 10] \cup [11, 20] \cup [41, 50]$ are generated as follows:

1. For samples from 1 to 10, $x^p \sim \mathcal{N}(1, 0.05)$, for $p \in [1, 10]$ and $x^p \sim \mathcal{N}(0, 0.1)$, for $p \in [11, 80]$.

2. For samples from 11 to 30, $x^p \sim \mathcal{N}(1, 0.05)$, for $p \in [11, 20]$ and $x^p \sim \mathcal{N}(0, 0.1)$, for $p \in [1, 10] \cup [31, 80]$.

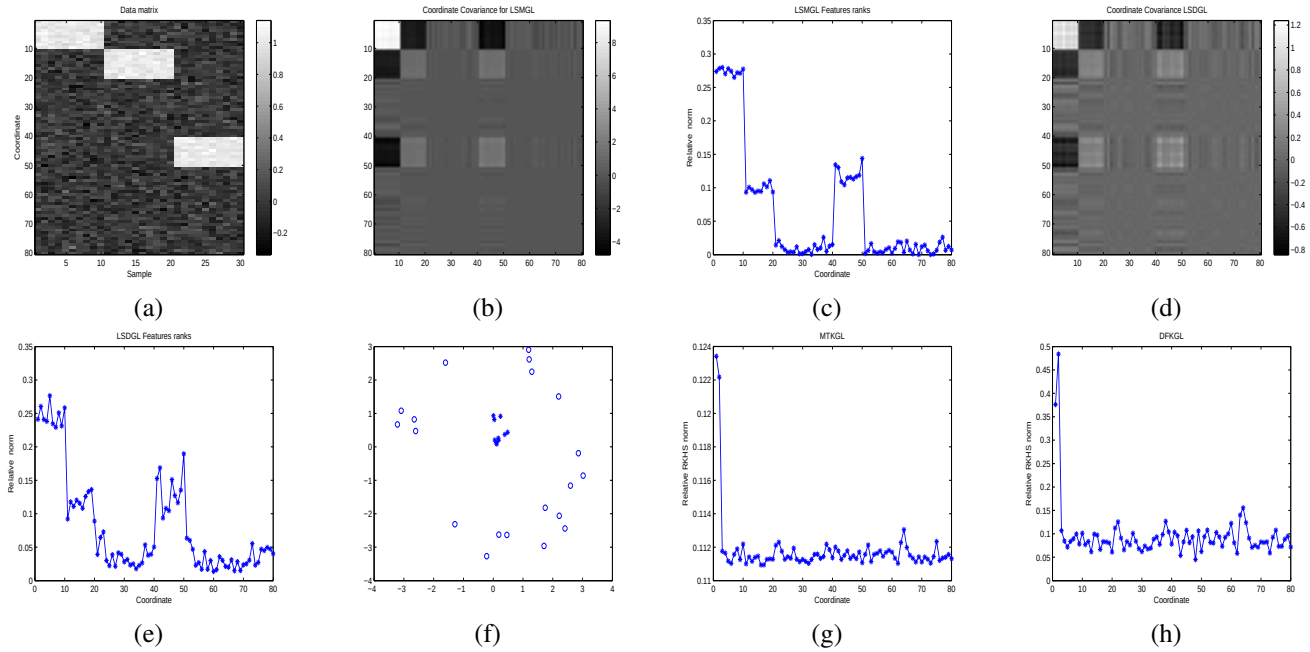
3. For samples from 11 to 30, features are in the form of $x^p \sim \mathcal{N}(1, 0.05)$, for $p \in [41, 50]$, and $x^p \sim \mathcal{N}(0, 0.1)$, for $p \in [1, 40] \bigcup [51, 80]$.

We let the regression function f_ρ to be a linear function. Specifically, we choose a noise parameter $\sigma_y = 3$ and the regression function is defined by $f_\rho(x_i) = w_1^T x_i$ for $i \in [1, 10]$, $f_\rho(x_i) = w_2^T x_i$ for $i \in [11, 20]$, and $f_\rho(x_i) = w_3^T x_i$ for $i \in [21, 30]$, where, $w_1^k = 2 + 0.5\sin(2\pi k/10)$ for $k \in [1, 10]$ and otherwise zero, $w_2^k = -2 - 0.5\sin(2\pi k/10)$ for $k \in [11, 20]$ and zero otherwise. The vector w_3 is defined by $w_3^k = -2 - 0.5\sin(2\pi k/10)$ for $k \in [41, 50]$ and zero otherwise.
In this linear case, we use the kernel $G(x, t) = x^T t$ as a basic scalar kernel and the multi-task kernel $\mathcal{K}$ defined by (6) in LSMGL algorithm (4). As in [11, 15], the regularization parameter λ is set to be a fixed number such as 0.1 (variation in this parameter made little difference to feature ranking). The parameter s in the weight coefficients w_{ij} is set to be the median pairwise distance between inputs. In Figure 1, the result of LSMGL is shown in (b) for variable covariation and in (c) for feature selection respectively. We also ran the algorithm (3) with the choice of kernel $\mathcal{K}(x, t) = G(x, t)I_{d+1}$ ([15, 16, 17]). The results are shown in (d) and (e) of Figure 1. We see that both algorithms worked well. The LSMGL algorithm works slightly better: the reason maybe be that it captures the inherent structure of gradient learning as mentioned before. We also ran LSMGL algorithm on this dataset; the result is no essentially different from SVMMGL.

In the second experiment, we use the SVMMGL algorithm for classification. For this dataset, only the first two features are relevant to the classification task. The remaining 78 redundant features are distributed according to a small Gaussian random deviate. The distribution for the first two features is shown in (f). In SVMMGL, the parameter s and λ are the same as those in the first example. The scalar kernel is set to be a Gaussian $G(x, t) = e^{-\|x-t\|^2/2\sigma^2}$ where σ is also the median pairwise distance between inputs. The feature selection results for the SVMMGL algorithm are illustrated respectively in (g) and (h) with different choices of multi-task kernels $\mathcal{K}$ given by equation (6) and $\mathcal{K}(x, t) = G(x, t)I_{d+1}$. Both algorithms picked up the two important features.

Finally, we apply our LSMGL algorithm to a well-studied expression dataset. This dataset has two classes: acute myeloid leumekia (AML) and acute lymphoblastic leukemia (ALL), see e.g. [10]. There are a total of 7129 genes (variables) and 72 patients, split into a training set of 38 examples and a test set of 34 examples. In the training set, 27 examples belong to ALL and 11 belong to AML, and the test set is composed of 20 ALL and 14 AML. Various variable selection algorithms have been applied to this dataset by choosing features based on training set, and then performing classification on the test set with the selected features. We ran LSMGL with the choice of multi-task $\mathcal{K}$ given by equation (6) where G is a linear kernel. The solution $\overrightarrow{f_{\mathbf{z}}}$ is learned from the training set for ranking the genes according to the values of s_p defined by equation (29). Then, ridge regression is run on the training set with truncated features to build a classifier to predict the labels on the test set. The regularization parameter of LSMGL is fixed to be 0.1 while the regularization parameter in ridge regression is tuned using leave-one-out cross-validation in the training set. The test error with selected top ranked genes is reported in Table 2. The classification accuracy is quite comparable to the gradient learning algorithm using individual RKHSs [15, 17]. However, [11, 15, 17] did the recursive techniques to rank features and employed SVM for classification while our method showed that ridge regression for classification and non-recursive technique for feature ranking also worked well in this data set. It would be interesting to further explore this issue.

The preliminary experiments above validated our proposed MGL algorithms. However further experiments need to be performed to evaluate our multi-task framework for gradient learning.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 1: LSMGL and SVMMGL feature ranking

genes	10	40	80	100	200	500
test error	2	1	0	0	1	1
genes	1000	2000	3000	4000	6000	7129
test error	2	1	1	1	1	1

Table 2: Number of test error using ridge regression algorithm versus the number of top ranked genes selected by LSMGL algorithm.

7 Conclusions

In this paper, our main contribution was to provide a novel unifying framework for gradient learning from the perspective of multi-task learning. Various variable selection methods in the literature can be recovered by the choice of multi-task kernels. More importantly, this framework allows us to introduce a novel choice of multi-task kernel to capture the inherent structure of gradient learning. An appealing representer theorem was presented which facilitates the design of efficient optimization algorithms, especially for datasets with high dimension and few training examples. Finally, a statistical error analysis was provided to ensure the convergence of the learned function to true function and true gradient.

Here we only preliminarily validated the method. A more extensive benchmark study remains to be pursued. In future we will explore more experiments on biomedical datasets and compare our MGL algorithms with previous related methods for feature selection, such as those in [21, 22] etc. It will be interesting to implement different loss functions in the MGL algorithms for regression and classification, apply the spectral decomposition of the gradient outer products to dimension reduction (see e.g. [16]), and possible use for network inference from the covariance of the learned gradient function.

References

[1] R. K. Ando & T. Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *J. Machine Learning Research*, 1817–1853, 2005

[2] A. Argyriou, T. Evgeniou, & M. Pontil. Multi-task feature learning. *NIPS*, 2006.

[3] N. Aronszajn. Theory of reproducing kernels. *Trans. Amer. Math. Soc.* 68: 337–404, 1950.

[4] P. L. Bartlett & S. Mendelson. Rademacher and Gaussian complexities: Risk bounds and structural results. *J. of Machine Learning Research*, 3:463–482, 2002.

[5] S. Ben-David & R. Schuller. Exploiting task relatedness for multiple task learning. *COLT*, 2003.

[6] D. R. Chen, Q. Wu, Y. Ying, & D. X. Zhou. Support vector machine soft margin classifiers: Error analysis. *J. of Machine Learning Research*, 5:1143–1175, 2004.

[7] A. Caponnetto, C.A. Micchelli, M. Pontil, & Y. Ying. Universal multi-task kernels, Preprint, 2007.

[8] S.S. Chen, D.L. Donoho & M.A. Saunders. Atomic decomposition pursuits. *SIAM J. of Scientific Computing*, 20: 33-61,1999.

[9] F. Cucker & S.Smale. On the mathematical foundations of learning, *Bull. Amer. Math. Soc.* 39: 149, 2001.

[10] T. R. Golub et. al. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, *Science*, 286: 531-537, 1999.

[11] I. Guyon, J.Weston, S. Barnhill, & V. Vapnik. Gene selection for cancer classification using support vector machines. *Machine learning* 46: 389–422, 2002.

[12] V.I. Koltchinskii & D. Panchenko. Rademacher processes and bounding the risk of function learning. In J.

Wellner E. Gine, D. Mason, editor, High Dimensional Probability II, pages 443459, 2000.
[13] T. Evgeniou, C. A. Micchelli & M. Pontil. Learning multiple tasks with kernel methods. *J. Machine Learning Research*, 6: 615–637, 2005.
[14] C. A. Micchelli & M. Pontil. On learning vector-valued functions. *Neural Computation*, 17: 177-204, 2005.
[15] S. Mukherjee & Q. Wu. Estimation of gradients and coordinate covariation in classification. *J. of Machine Learning Research* 7: 2481-2514, 2006.
[16] S. Mukherjee, Q. Wu, & D. X. Zhou. Learning gradients and feature selection on manifolds. Preprint, 2007.
[17] S. Mukherjee & D. X. Zhou. Learning coordinate covariances via gradients, *J. of Machine Learning Research* 7: 519-549, 2006.
[18] I. J. Schoenberg. Metric spaces and completely monotone functions, *Ann. of Math.* 39: 811-841, 1938.
[19] B. Schölkopf & A. J. Smola. *Learning with Kernels.* The MIT Press, Cambridge, MA, USA, 2002.
[20] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge, 2004.
[21] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Royal. Statist. Soc B.* 58: 267-288, 1996.
[22] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, & V. Vapnik. Feature selection for SVMs, *NIPS*, 2001.
[23] V. N. Vapnik. *Statistical learning theory*. Wiley, New York, 1998.

Appendix

Let G be a scalar kernel, we use the convention $\partial_p^{(2)}G$ to denote the p-th partial derivative of G with respect to the second argument, and so is the gradient $\nabla^{(2)}G$.

Proof of Proposition 1.

When G is a linear kernel, by the definition (6) of multi-task kernel $\mathcal{K}$, we have that $\overrightarrow{f_{2\mathbf{z}}} = \sum_{j\in\mathbb{N}_m}(x_j, I_d)c_{j,\mathbf{z}}$ which implies that

$$Cov(\overrightarrow{f_{2\mathbf{z}}}) = \overrightarrow{f_{2\mathbf{z}}}\overrightarrow{f_{2\mathbf{z}}}^T = \sum_{i,j\in\mathbb{N}_m}(x_i, I_d)c_{i,\mathbf{z}}c_{j,\mathbf{z}}^T(x_j, I_d)^T.$$

For the LSMGL algorithm, in Section 4 we showed that $c_{j,\mathbf{z}} = \widetilde{\mathcal{V}}\gamma_{j,\mathbf{z}}$ and, for any $j \in \mathbb{N}_m$, $x_j = \mathcal{V}\beta_j$, the above equation can be further simplified to the following:

$$\mathrm{Cov}(\overrightarrow{f_{2\mathbf{z}}}) = \mathcal{V}\Big[\sum_{i,j\in\mathbb{N}_m}(\beta_i, I_s)\gamma_{i,\mathbf{z}}\gamma_{j,\mathbf{z}}^T(\beta_j, I_s)^T\Big]\mathcal{V}^T.$$

When G is an RBF kernel, for any $x, t \in \mathbb{R}^d$ and $p, q \in \mathbb{N}_d$, $G(x,t) = G(x-t)$ and $\partial_{t_q}\partial_{x_p}G(x,t) = -(\partial_p\partial_q G)(x,t)$. Hence, for any $p \in \mathbb{N}_d$ and $x \in \mathbb{R}^d$, we have that

$$f_{p+1,\mathbf{z}}(x) = \sum_{j\in\mathbb{N}_m}\big(-\partial_p^{(2)}G(x,x_j), -\nabla^{(2)}\partial_p^{(2)}G(x,x_j)\big)c_j$$

Since G has a feature representation, i.e., for any $x, t \in X$, there holds that $G(x,t) = \langle\phi(x), \phi(t)\rangle_{\ell^2}$. Also, observe that $\partial_p^{(2)}G(x,x_j) = \langle\phi(x), (\partial_p\phi)(x_j)\rangle_{\ell^2}$ and $\partial_q^{(2)}\partial_p^{(2)}G(x,x_j) = \langle\phi(x), (\partial_p\partial_q\phi)(x_j)\rangle_{\ell^2}$. Denote c_j by $(c_j^1, \ldots, c_j^{d+1})^T$. Consequently, for any $p \in \mathbb{N}_d$, $f_{p+1,\mathbf{z}}(\cdot) = \langle w_p, \phi(\cdot)\rangle_{\ell^2}$ with $w_p = -\sum_{j\in\mathbb{N}_m}(\partial_p\phi(x_j)c_j^1 + \sum_{q\in\mathbb{N}_d}\partial_q\partial_p\phi(x_j)c_j^{q+1})$. Therefore, for any $p, q \in \mathbb{N}_d$ we have that $f_{p+1,\mathbf{z}} \in \mathcal{H}_G$ and

$$\langle f_{p+1,\mathbf{z}}, f_{q+1,\mathbf{z}}\rangle_G = \langle w_p, w_q\rangle_{\ell^2} = \sum_{i,j\in\mathbb{N}_m} c_i^T\mathbb{K}_{pq}(x_i - x_j)c_j$$

which completes the assertion. □

Computation of kernel $\mathbb{K}$.

If G is a Gaussian kernel with standard variation σ, that is, for any $x, t \in \mathbb{R}^d, G(x,t) = G(x-t) = e^{-\frac{\|x-t\|^2}{2\sigma}}$, the computation of $\mathbb{K}$ is listed as follows.

1. $(\partial_{pq}^2 G)(x) = \left[\frac{x_p x_q}{\sigma^2} - \frac{\delta_{pq}}{\sigma}\right]G(x)$

2. For any $q' \in \mathbb{N}_d$, $\partial_{q'}\partial_p\partial_q G(x) = [\frac{x_q\sigma_{pq'} + x_p\delta_{qq'} + x_{q'}\delta_{pq}}{\sigma^2} - \frac{x_p x_q x_{q'}}{\sigma^3}]G(x)$. Hence,

$$\nabla\partial_{pq}^2 G(x) = \left[\frac{e_p x_q + e_q x_p}{\sigma^2} + \frac{x\delta_{pq}}{\sigma^2} - \frac{x_p x_q x}{\sigma^3}\right]G(x)$$

3. For any $p', q' \in \mathbb{N}_d$, $\partial_{p'}\partial_{q'}\partial_{pq}^2 G(x) = G(x)\Big[\frac{\delta_{pq'}\delta_{qp'}}{\sigma^2} + \frac{\delta_{pp'}\delta_{qq'} + \delta_{p'q'}\delta_{pq}}{\sigma^2} - \frac{1}{\sigma^3}\big(x_q x_{q'}\sigma_{pp'} + x_p x_q\delta_{p'q'} + x_p x_{q'}\delta_{p'q}\big) + \frac{x_{p'}}{\sigma}\big(\frac{x_p x_q x_{q'}}{\sigma^3} - \frac{x_q\delta_{pq'} + x_{q'}\delta_{pq} + x_p\delta_{qq'}}{\sigma^2}\big)\Big]$. Hence,

$$\begin{aligned}\nabla^2\partial_{pq}^2 G(x) = G(x)\Big[&\frac{e_q^T e_p + e_p^T e_q + \delta_{pq}I_d}{\sigma^2}\\ &- \frac{1}{\sigma^3}\big(x_q(e_p x^T + x e_p^T) + x_p(x e_q^T + e_q x^T)\big)\\ &- \frac{x_p x_q I_d}{\sigma^3} + \frac{x x^T}{\sigma^3}\big(\frac{x_p x_q}{\sigma} - \sigma_{pq}\big)\Big].\end{aligned}$$

Proof of Theorem 7

We turn our attention to the proof of Theorem 7. We begin with some notations and background materials. First denote by $\mathrm{Er}_{\mathbf{z}}$ the empirical loss in LSMGL algorithm, i.e.,

$$\begin{aligned}\mathrm{Er}_{\mathbf{z}}(\overrightarrow{f}) = \ &\frac{1}{m^2}\sum_{i,j} w(x_i - x_j)\\ &\times(y_i - f_1(x_j) - \overrightarrow{f_2}(x_j)(x_i - x_j)^T)^2,\end{aligned}$$

and the modified form of its expectation

$$\begin{aligned}\mathrm{Er}(\overrightarrow{f}) = \ &\int_Z\int_X w(x-u)\big[y - f_1(u)\\ &-\overrightarrow{f_2}(u)(x-u)\big]^2 d\rho(x,y)d\rho_X(u).\end{aligned}$$

Since the Gaussian weight $w(x-u) = w_s(x-u)$ is dependent on s, the above definition of $\mathrm{Er}(\overrightarrow{f})$ is depending on the parameter s. In addition, define the Lipschitz constant $\big|\nabla f_\rho\big|_{\mathrm{Lip}}$ to be the minimum constant c such that $\|\nabla f_\rho(x+u) - \nabla f_\rho(x)\| \le c\|u\|, \ \ \forall x, u \in X$. We say that ∇f_ρ is Lipschitz continuous if $\big|\nabla f_\rho\big|_{\mathrm{Lip}}$ is finite.

The error analysis here is divided into two main steps motivated by the techniques in [15]. The first step is to bound the square error $\|\overrightarrow{f_{\mathbf{z}}} - \overrightarrow{f_\rho}\|_{\rho_X}^2$ by the excess error $\mathrm{Er}(\overrightarrow{f_{\mathbf{z}}}) - \mathrm{Er}(\overrightarrow{f_\rho})$. In the second step, we employ standard error decomposition [6] and Rademacher complexities [4, 12] to estimate the excess error. These two steps will be respectively stated in the

following two propositions. Before we do that, we introduce an auxiliary functional $\mathcal{Q}_s$ defined by

$$\mathcal{Q}_s(\overrightarrow{f}, \overrightarrow{f}_\rho) = \int\int w(x-u)\big[f_\rho(u) - f_1(u) + (\overrightarrow{f_2}(u) - \nabla f_\rho(u))(u-x)^T\big]^2 d\rho_X(x) d\rho_X(u).$$

We are ready to present the first step of the error analysis: bounding the square error $\|\overrightarrow{f_{\mathbf{z}}} - \overrightarrow{f_\rho}\|^2_{\rho_X}$ by the excess error $\mathrm{Er}(\overrightarrow{f_{\mathbf{z}}}) - \mathrm{Er}(\overrightarrow{f_\rho})$ which is stated as the following proposition.

Proposition 2 *If $0 < s, \lambda < 1$ then there exists a constant c'_ρ such that*

$$\mathbb{E}\Big[\|\overrightarrow{f_{\mathbf{z}}} - \overrightarrow{f_\rho}\|^2_{\rho_X}\Big] \le c'_\rho\Big(\min\big[s^{-\theta}, \max_{x\in X} p^{-1}(x)\big] \times \mathbb{E}\big[Er(\overrightarrow{f_{\mathbf{z}}}) - Er(\overrightarrow{f_\rho})\big] + s^\theta\big(\mathbb{E}\big[\|\overrightarrow{f_{\mathbf{z}}}\|^2_{\mathcal{K}}\big] + \|\overrightarrow{f_\rho}\|_\infty + \big|\nabla f_\rho\big|^2_{Lip}\big)\Big).$$

The proof of this proposition follows directly from the following Lemmas 8 and 9. For this purpose, let the subset X_s of X be

$$X_s = \big\{u \in X : \mathrm{dist}(u, \partial X) > s, |p(u)| \ge (1+c_\rho)s^\theta\big\} \tag{31}$$

and

$$c_p(s) := \min\{p(x) : \|u-x\| \le s, u \in X_s\}.$$

Recall that $\widetilde{e}_1$ is the first coordinate basis in $\mathbb{R}^{d+1}$ and, for any $x \in \mathbb{R}^d$, $\widetilde{x}^T = (0, x)^T \in \mathbb{R}^{d+1}$.

Lemma 8 *If $0 < s < 1$ then there exists a constant c'_ρ such that*

$$\mathbb{E}\Big[\|\overrightarrow{f_{\mathbf{z}}} - \overrightarrow{f_\rho}\|^2_{\rho_X}\Big] \le c'_\rho\Big(s^\theta\big[\mathbb{E}[\|\overrightarrow{f_{\mathbf{z}}}\|^2_{\mathcal{K}}] + \|\overrightarrow{f_\rho}\|_\infty\big] + \min\big[s^{-\theta}, \max_{x\in X} p^{-1}(x)\big]\mathbb{E}\Big[\mathcal{Q}_s(\overrightarrow{f_{\mathbf{z}}}, \overrightarrow{f_\rho})\Big]\Big).$$

Proof: Write $\|\overrightarrow{f_{\mathbf{z}}} - \overrightarrow{f_\rho}\|^2_{\rho_X}$ by

$$\begin{aligned}\|\overrightarrow{f_{\mathbf{z}}} - \overrightarrow{f_\rho}\|^2_{\rho_X} &= \int_{X\setminus X_s} \|\overrightarrow{f_{\mathbf{z}}}(u) - \overrightarrow{f_\rho}(u)\|^2 d\rho_X(u) \\ &\quad + \int_{X_s} \|\overrightarrow{f_{\mathbf{z}}}(u) - \overrightarrow{f_\rho}(u)\|^2 d\rho_X(u)\end{aligned} \tag{32}$$

By the definition of X_s, we have that $\rho_X(X\setminus X_s) \le c_\rho s + c_\rho(1+c_\rho)|X|s^\theta \le c'_\rho s^\theta$ where $|X|$ is the Lebesgue measure of X. Hence, the first term of the above equation is bounded by

$$2c'_\rho(\|\overrightarrow{f_{\mathbf{z}}}\|^2_\infty + \|\overrightarrow{f_\rho}\|^2_\infty)s^\theta \le 2c'_\rho(\|\overrightarrow{f_{\mathbf{z}}}\|^2_{\mathcal{K}} + \|\overrightarrow{f_\rho}\|^2_\infty)s^\theta.$$

For the second term on the right-hand side of equation (32), observe that, for any $u \in X_s$, $\mathrm{dist}(u, \partial X) > s$ and $\{u : \|u-x\| \le s, x \in X_s\} \subseteq X$. Moreover, for any $x \in X$ such that $\|u-x\| \le s$, by the definition of X_s there holds

$$p(x) = p(u) - (p(u) - p(x)) \ge (1+c_\rho)s^\theta - c_\rho\|u-x\|^\theta \ge s^\theta.$$

Consequently, it follows that

$$c_p(s) \ge \max(s^\theta, \min_{x\in X} p(x)), \tag{33}$$

and

$$\begin{aligned}&\mathcal{Q}_s(\overrightarrow{f_{\mathbf{z}}}, \overrightarrow{f_\rho}) = \int_X\int_X w(x-u) \\ &\times\big[(\overrightarrow{f_\rho}(u) - \overrightarrow{f_{\mathbf{z}}}(u))(\widetilde{e}_1 + \widetilde{u} - \widetilde{x})\big]^2 d\rho_X(x) d\rho_X(u) \\ &\ge \int_{X_s}\Big[\int_{\|u-x\|\le s}\big((\overrightarrow{f_\rho}(u) - \overrightarrow{f_{\mathbf{z}}}(u))(\widetilde{e}_1 + \widetilde{u} - \widetilde{x})\big)^2 \\ &\qquad \times d\rho_X(x)\Big] d\rho_X(u) \\ &\ge c_\rho(s)\int_{X_s}\Big[\int_{\|u-x\|\le s}\big((\overrightarrow{f_\rho}(u) - \overrightarrow{f_{\mathbf{z}}}(u))(\widetilde{e}_1 + \widetilde{u} - \widetilde{x})\big)^2 \\ &\qquad \times dx\Big] d\rho_X(u).\end{aligned} \tag{34}$$

The integral w.r.t. x on the right-hand side of the above inequality can be written as $(\overrightarrow{f_\rho}(u) - \overrightarrow{f_{\mathbf{z}}}(u))W(s)(\overrightarrow{f_\rho}(u) - \overrightarrow{f_{\mathbf{z}}}(u))^T$ with $(d+1)\times(d+1)$ matrix $W(s)$ defined by

$$W(s) = \int_{\|u-x\|\le s}\big[(\widetilde{e}_1 + \widetilde{u} - \widetilde{x})(\widetilde{e}_1 + \widetilde{u} - \widetilde{x})^T\big]dx.$$

Here, $(\widetilde{e}_1 + \widetilde{u} - \widetilde{x})(\widetilde{e}_1 + \widetilde{u} - \widetilde{x})^T$ equals that

$$\begin{pmatrix} 1 & (u-x)^T \\ u-x & (u-x)(u-x)^T \end{pmatrix}$$

Observe that $\int_{\|u-x\|\le s} w(x-u)dx = s^{-2}\int_{\|t\|\le 1} e^{-\frac{\|t\|^2}{2}}dt$ and $\int_{\|u-x\|\le s} w(x-u)(u-x)dx = 0$. In addition, for any $p \ne q \in \mathbb{N}_d$, $\int_{\|u-x\|\le s} w(x-u)(u^p - x^p)(u^q - x^q)dx = 0$

$\int_{\|u-x\|\le s} w(x-u)(x^p - u^p)^2 dx = \int_{\|t\|\le 1} e^{-\frac{\|t\|^2}{2}}(t^p)^2 dt$

From the above observations, there exists a constant c such that

$(\overrightarrow{f_\rho}(u) - \overrightarrow{f_{\mathbf{z}}}(u))W(s)(\overrightarrow{f_\rho}(u) - \overrightarrow{f_{\mathbf{z}}}(u))^T \ge c\|\overrightarrow{f_\rho}(u) - \overrightarrow{f_{\mathbf{z}}}(u)\|^2$.

Recalling the definition of $W(s)$ and substituting this back into equation (34) implies, for any $0 < s < 1$, that

$$c\, c_\rho(s)\int_{X_s}\|\overrightarrow{f_{\mathbf{z}}}(u) - \overrightarrow{f_\rho}(u)\|^2 d\rho_X(u) \le \mathcal{Q}_s(\overrightarrow{f_{\mathbf{z}}}, \overrightarrow{f_\rho}).$$

Plugging this into equation (34), the desired estimate follows from the estimation of $c_\rho(s)$, i.e., equation (33). ■

Now we can bound $\mathcal{Q}_s$ by the following lemma.

Lemma 9 *If $0 < s < 1$ then there exists a constant c such that, for any $\overrightarrow{f} \in \mathcal{H}_{\mathcal{K}}$, the following equations hold true.*

1. $\mathcal{Q}_s(\overrightarrow{f}, \overrightarrow{f_\rho}) \le c\Big(s^2\big|\nabla f_\rho\big|^2_{Lip} + \big[Er(\overrightarrow{f}) - Er(\overrightarrow{f_\rho})\big]\Big)$.

2. $Er(\overrightarrow{f}) - Er(\overrightarrow{f_\rho}) \le c\Big(s^2\big|\nabla f_\rho\big|^2_{Lip} + \mathcal{Q}_s(\overrightarrow{f}, \overrightarrow{f_\rho})\Big)$.

Proof: Observe that $\big[y - f_1(u) - \overrightarrow{f_2}(u)(x-u)^T\big]^2 = \big[y - f_\rho(u) - \nabla f_\rho(u)(x-u)^T\big]^2 + 2\big[y - f_\rho(u) - \nabla f_\rho(u)(x-u)^T\big]\big[f_\rho(u) - f_1(u) + (\overrightarrow{f_2}(u) - \nabla f_\rho(u))(u-x)^T\big] + \big[f_\rho(u) - f_1(u) + (\overrightarrow{f_2}(u) - \nabla f_\rho(u))(u-x)^T\big]^2$. Then, taking the integral of both sides of the above equality and using the fact that $f_\rho(x) = \int_Y y d\rho_X(x)$ we have that

$$\begin{aligned}&\mathrm{Er}(\overrightarrow{f}) - \mathrm{Er}(\overrightarrow{f_\rho}) = \mathcal{Q}_s(\overrightarrow{f}, \overrightarrow{f_\rho}) + 2\int_X\int_X w(x-u)\big[f_\rho(x) \\ &-f_\rho(u) - \nabla f_\rho(u)(x-u)^T\big]\big[f_\rho(u) - f_1(u) + \\ &(\overrightarrow{f_2}(u) - \nabla f_\rho(u))(u-x)^T\big]d\rho_X(x)d\rho_X(u) \\ &\ge \mathcal{Q}_s(\overrightarrow{f}, \overrightarrow{f_\rho}) - 2\Big(\int_X\int_X w(x-u)\big[f_\rho(x) - f_\rho(u) \\ &-\nabla f_\rho(u)(x-u)^T\big]^2 d\rho_X(x)d\rho_X(u)\Big)^{\frac{1}{2}}\Big(\mathcal{Q}_s(\overrightarrow{f}, \overrightarrow{f_\rho})|\Big)^{\frac{1}{2}}.\end{aligned}$$

Applying the inequality, for any $a, b > 0$, that $-2a^2 - \frac{1}{2}b^2 \leq -2ab$, from the above equality we further have that

$$\begin{aligned}\mathrm{Er}(\overrightarrow{f}) - \mathrm{Er}(\overrightarrow{f_\rho}) \geq \tfrac{1}{2}\mathcal{Q}_s(\overrightarrow{f}, \overrightarrow{f_\rho}) - 2\int_X\int_X w(x-u)\\ \big[f_\rho(x) - f_\rho(u) - \nabla f_\rho(u)(x-u)^T\big]^2 d\rho_X(x)d\rho_X(u).\end{aligned} \tag{35}$$

Likewise,

$$\begin{aligned}&\mathrm{Er}(\overrightarrow{f}) - \mathrm{Er}(\overrightarrow{f_\rho}) = \mathcal{Q}_s(\overrightarrow{f}, \overrightarrow{f_\rho}) + 2\int_X\int_X w(x-u)\big[f_\rho(x)\\ &-f_\rho(u) - \nabla f_\rho(u)(x-u)^T\big]\big[f_\rho(u) - f_1(u) +\\ &(\overrightarrow{f_2}(u) - \nabla f_\rho(u))(u-x)^T\big]d\rho_X(x)d\rho_X(u)\\ &\leq \mathcal{Q}_s(\overrightarrow{f}, \overrightarrow{f_\rho}) + 2\Big(\int_X\int_X w(x-u)\big[f_\rho(x) - f_\rho(u)\\ &-\nabla f_\rho(u)(x-u)^T\big]^2 d\rho_X(x)d\rho_X(u)\Big)^{\frac{1}{2}}\Big(\mathcal{Q}_s(\overrightarrow{f}, \overrightarrow{f_\rho})|\Big)^{\frac{1}{2}}.\end{aligned}$$

Applying the inequality $2ab \leq a^2 + b^2$ to the above inequality yields that

$$\begin{aligned}\mathrm{Er}(\overrightarrow{f}) - \mathrm{Er}(\overrightarrow{f_\rho}) \leq 2\mathcal{Q}_s(\overrightarrow{f}, \overrightarrow{f_\rho}) + \int_X\int_X w(x-u)\\ \big[f_\rho(x) - f_\rho(u) - \nabla f_\rho(u)(x-u)^T\big]^2 d\rho_X(x)d\rho_X(u)\end{aligned} \tag{36}$$

However, $|f_\rho(x) - f_\rho(u) - \nabla f_\rho(u)(x-u)^T| = \big|\int_0^1 \big(\nabla f_\rho(tx + (1-t)u) - \nabla f_\rho(u)\big)(x-u)^T dt\big| \leq |\nabla f_\rho|_{\mathrm{Lip}}\|x-u\|^2$ and the density $p(x)$ of ρ_X is a bounded function since we assume it is θ-Hölder continuous and X is compact. Therefore,

$$\begin{aligned}&\int_X\int_X w(x-u)\big[f_\rho(x) - f_\rho(u) - \nabla f_\rho(u)(x-u)^T\big]^2\\ &\times d\rho_X(x)d\rho_X(u)\\ &\leq \|p\|_\infty|\nabla f_\rho|^2_{\mathrm{Lip}}\Big[\int_{\mathbb{R}^d}\tfrac{1}{s^{d+2}}e^{-\frac{\|x\|^2}{2s^2}}\|x\|^4 dx\Big]\\ &\leq c\|p\|_\infty|\nabla f_\rho|^2_{\mathrm{Lip}}\, s^2.\end{aligned}$$

Putting this into Equations (35) and (36) and arranging the terms involved yields the desired result. ■

From Property (1) of Lemma 9, for any $\overrightarrow{f} \in \mathcal{H}_\mathcal{K}$ we have that

$$\mathrm{Er}(\overrightarrow{f}) - \mathrm{Er}(\overrightarrow{f_\rho}) \geq -cs^2|\nabla f_\rho|^2_{\mathrm{Lip}}. \tag{37}$$

We now turn our attention to the second step of the error analysis: the estimation of the excess error $\mathrm{Er}(\overrightarrow{f_\mathbf{z}}) - \mathrm{Er}(\overrightarrow{f_\rho}) + \lambda\|\overrightarrow{f_\mathbf{z}}\|^2_\mathcal{K}$. To do this, let

$$\overrightarrow{f_\lambda} = \arg\inf_{\overrightarrow{f}\in\mathcal{H}_\mathcal{K}}\Big\{\mathrm{Er}(\overrightarrow{f}) + \lambda\|\overrightarrow{f}\|^2_\mathcal{K}\Big\}$$

By the *error decomposition* technique in [6], we get the following estimation.

Proposition 3 *If $\overrightarrow{f_\lambda}$ is defined above then there exists a constant c such that*

$$\begin{aligned}Er(\overrightarrow{f_\mathbf{z}}) - Er(\overrightarrow{f_\rho}) + \lambda\|\overrightarrow{f_\mathbf{z}}\|^2_\mathcal{K} \leq \mathcal{S}(\mathbf{z})\\ +c\big(s^2\big|\nabla f_\rho\big|^2_{Lip} + \mathcal{A}(\lambda, s)\big),\end{aligned}$$

where

$$\mathcal{S}(\mathbf{z}) = Er(\overrightarrow{f_\mathbf{z}}) - Er_\mathbf{z}(\overrightarrow{f_\mathbf{z}}) + Er_\mathbf{z}(\overrightarrow{f_\lambda}) - Er(\overrightarrow{f_\lambda})$$

is referred to the sample error and

$$\mathcal{A}(\lambda, s) = \inf_{\overrightarrow{f}\in\mathcal{H}_\mathcal{K}}\Big\{\mathcal{Q}_s(\overrightarrow{f}, \overrightarrow{f_\rho}) + \lambda\|\overrightarrow{f}\|^2_\mathcal{K}\Big\}$$

is called the approximation error.

Proof: Note that $\mathrm{Er}(\overrightarrow{f_\mathbf{z}}) - \mathrm{Er}(\overrightarrow{f_\rho}) + \lambda\|\overrightarrow{f_\mathbf{z}}\|^2_\mathcal{K} = \big[\mathrm{Er}(\overrightarrow{f_\mathbf{z}}) - \mathrm{Er}_\mathbf{z}(\overrightarrow{f_\mathbf{z}}) + \mathrm{Er}_\mathbf{z}(\overrightarrow{f_\lambda}) - \mathrm{Er}(\overrightarrow{f_\lambda})\big] + \big[\big(\mathrm{Er}_\mathbf{z}(\overrightarrow{f_\mathbf{z}}) + \lambda\|\overrightarrow{f_\mathbf{z}}\|^2_\mathcal{K}\big) - \big(\mathrm{Er}_\mathbf{z}(\overrightarrow{f_\lambda}) + \lambda\|\overrightarrow{f_\lambda}\|^2_\mathcal{K}\big)\big] + \big[\mathrm{Er}(\overrightarrow{f_\lambda}) - \mathrm{Er}(\overrightarrow{f_\rho}) + \lambda\|\overrightarrow{f_\lambda}\|^2_\mathcal{K}\big]$. By the definition of $\overrightarrow{f_\mathbf{z}}$, we know that the second term in parenthesis on the right-hand side of the above equation is negative. Hence, by the definition of f_λ, we get that $\mathrm{Er}(\overrightarrow{f_\mathbf{z}}) - \mathrm{Er}(\overrightarrow{f_\rho}) + \lambda\|\overrightarrow{f_\mathbf{z}}\|^2_\mathcal{K} \leq \mathcal{S}(\mathbf{z}) + \inf_{f\in\mathcal{H}_\mathcal{K}}\big\{\mathrm{Er}(\overrightarrow{f}) - \mathrm{Er}(\overrightarrow{f_\rho}) + \lambda\|\overrightarrow{f}\|^2_\mathcal{K}\big\}$. By the property (2) in Lemma 9, we also have, for any $\overrightarrow{f} \in \mathcal{H}_\mathcal{K}$, that $\mathrm{Er}(\overrightarrow{f}) - \mathrm{Er}(\overrightarrow{f_\rho}) \leq c\Big((s^2\big|\nabla f_\rho\big|^2_{\mathrm{Lip}} + \mathcal{Q}_s(\overrightarrow{f}, \overrightarrow{f_\rho})\Big)$ which implies that

$$\inf\{\mathrm{Er}(\overrightarrow{f}) - \mathrm{Er}(\overrightarrow{f_\rho}) + \lambda\|\overrightarrow{f}\|^2_\mathcal{K}\} \leq c\Big(s^2\big|\nabla f_\rho\big|^2_{\mathrm{Lip}} + \mathcal{A}(\lambda, s)\Big).$$

This completes the proposition. ■

Now it suffice to estimate the sample error $\mathcal{S}(\mathbf{z})$. To this end, observe that $\mathrm{Er}_\mathbf{z}(\overrightarrow{f_\mathbf{z}}) + \lambda\|\overrightarrow{f_\mathbf{z}}\|^2_\mathcal{K} \leq \mathrm{Er}_\mathbf{z}(0) + \lambda\|0\|^2_\mathcal{K} \leq \frac{M^2}{s^{d+2}}$ which implies that $\|\overrightarrow{f_\mathbf{z}}\|_\mathcal{K} \leq Ms^{-(d+2)/2}$. Likewise, $\mathrm{Er}(\overrightarrow{f_\lambda}) + \lambda\|\overrightarrow{f_\lambda}\|^2_\mathcal{K} \leq \mathrm{Er}(0) + \lambda\|0\|^2_\mathcal{K}\|\overrightarrow{f_\lambda}\|_\mathcal{K} \leq \frac{M^2}{s^{d+2}}$ which tells us that $\|\overrightarrow{f_\mathbf{z}}\|_\mathcal{K} \leq Ms^{-(d+2)/2}$. Using these bounds on $\|\overrightarrow{f_\mathbf{z}}\|_\mathcal{K} + \|\overrightarrow{f_\lambda}\|_\mathcal{K}$, we can use the Rademacher averages (see e.g. [4, 12]) for its definition and properties) to get the following estimation for the sample error.

Lemma 10 *For any $0 < \lambda < 1$, there exists a constant c such that, for any $m \in \mathbb{N}$, there holds*

$$\mathbb{E}\big[\mathcal{S}(\mathbf{z})\big] \leq c\Big(\frac{1}{s^{2(d+2)}\lambda m} + \frac{1}{s^{3(d+2)/2}\sqrt{\lambda m}}\Big).$$

Since the proof of the above lemma is rather a standard approach and indeed parallel to the proof of Lemma 26 (replacing $r = Ms^{-(d+2)/2}$ there) in the appendix of [15], for the simplicity we omit the details here.

We have assembled all the materials to prove Theorem 7.

Proof of Theorem 7

Since we assume that $\overrightarrow{f_\rho} \in \mathcal{H}_\mathcal{K}$, for any $\big|\partial_p f_\rho(x+u) - \partial_p\overrightarrow{f_\rho}(u)\big| = \big|\langle\widetilde{e}_{p+1}, \overrightarrow{f_\rho}(x+u) - \overrightarrow{f_\rho}(u)\rangle\big| = \big|\langle\overrightarrow{f_\rho}, \mathcal{K}_{x+u}\widetilde{e}_{p+1} - \mathcal{K}_u\widetilde{e}_{p+1}\rangle_\mathcal{K}\big| \leq \|\overrightarrow{f_\rho}\|_\mathcal{K}\Big[\widetilde{e}^T_{p+1}\big(\mathcal{K}(x+u, x+u) + \mathcal{K}(u,u) - \mathcal{K}(x+u, u) - \mathcal{K}(u, x+u)\big)\widetilde{e}_{p+1}\Big]^{\frac{1}{2}} \leq c\|\overrightarrow{f_\rho}\|_\mathcal{K}\|u\|$, and hence $|\nabla f_\rho|_{\mathrm{Lip}} \leq c\|\overrightarrow{f_\rho}\|_\mathcal{K}$. Moreover,

$$\mathcal{A}(\lambda, s) \leq \mathcal{Q}(\overrightarrow{f_\rho}, \overrightarrow{f_\rho}) + \lambda\|\overrightarrow{f_\rho}\|^2_\mathcal{K} = \lambda\|\overrightarrow{f_\rho}\|^2_\mathcal{K}.$$

Hence, we know from Proposition 3 and equation (37) that $\lambda\|\overrightarrow{f_\mathbf{z}}\|^2_\mathcal{K} \leq \mathcal{S}(\mathbf{z}) + c'\big(s^2 + \lambda\big)$.

Combining the above equations with Propositions 2 and 3, there exists a constant c such that

$$\begin{aligned}\mathbb{E}\big[\|\overrightarrow{f_\mathbf{z}} - \overrightarrow{f_\rho}\|^2_{\rho_X}\big] \leq\ & c\Big(\big[\min(s^{-\theta}, \max_{x\in X}p^{-1}(x)) + \tfrac{s^\theta}{\lambda}\big]\\ &\times\big[\mathbb{E}\big[\mathcal{S}(\mathbf{z})\big] + s^2 + \lambda\big] + s^\theta\Big).\end{aligned}$$

If we choose $\lambda = s^{2\theta}$ and $s = m^{-\frac{1}{3(d+2)+4\theta}}$ yields the first assertion.

If ρ_X is the uniform distribution over X, then we have that $\min(s^{-\theta}, \min_{x\in X}p(x)) = 1$. Hence, choosing $\lambda = s^\theta$ and $s = m^{-\frac{1}{3(d+2)+5\theta}}$ we have the desired second assertion. This completes the theorem.