# Convex Learning of Multiple Tasks and their Structure

**Carlo Ciliberto**[1,2] CCILIBER@MIT.EDU
**Youssef Mroueh**[1,2] YMROUEH@MIT.EDU
**Tomaso Poggio**[1,2] TP@AI.MIT.EDU
[1]Laboratory for Computational and Statistical Learning, Istituto Italiano di Tecnologia, Via Morego 30, Genova, Italy
[2]Center for Brains Minds and Machines, Massachusetts Institute of Technology, Cambridge, MA 02139 USA

**Lorenzo Rosasco**[1,2,3] LROSASCO@MIT.EDU
[3]DIBRIS, Università di Genova, Via Dodecaneso, 35, 16146, Genova, Italy

## Abstract

Reducing the amount of human supervision is a key problem in machine learning and a natural approach is that of exploiting the relations (structure) among different tasks. This is the idea at the core of multi-task learning. In this context a fundamental question is how to incorporate the tasks structure in the learning problem. We tackle this question by studying a general computational framework that allows to encode a-priori knowledge of the tasks structure in the form of a convex penalty; in this setting a variety of previously proposed methods can be recovered as special cases, including linear and non-linear approaches. Within this framework, we show that tasks and their structure can be efficiently learned considering a convex optimization problem that can be approached by means of block coordinate methods such as alternating minimization and for which we prove convergence to the global minimum.

## 1. Introduction

Current machine learning systems achieve remarkable results in several challenging tasks, but are limited by the amount of human supervision required. Leveraging similarity among different problems is widely acknowledged to be a key approach to reduce the need for supervised data. Indeed, this idea is at the basis of multi-task learning, where the joint solution of different problems (tasks) has the potential to exploit tasks relatedness (structure) to improve learning accuracy. This idea has motivated a variety of methods, including frequentist (Micchelli & Pontil, 2004; Argyriou et al., 2008a;b) and Bayesian methods (see e.g. (Álvarez et al., 2012) and references therein), with connections to structured learning (Bakir et al., 2007; Tsochantaridis et al., 2004).

The focus of our study is the development of a general regularization framework to learn multiple tasks as well as their structure. Following (Micchelli & Pontil, 2004; Evgeniou et al., 2005) we consider a setting where tasks are modeled as the components of a vector-valued function and their structure corresponds to the choice of suitable functional spaces. Exploiting the theory of reproducing kernel Hilbert spaces for vector-valued functions (RKHSvv) (Micchelli & Pontil, 2004), we consider and analyze a flexible regularization framework, within which a variety of previously proposed approaches can be recovered as special cases, see e.g. (Jacob et al., 2008; Lozano & Sindhwani, 2011; Minh & Sindhwani, 2011; Zhang & Yeung, 2010; Dinuzzo et al., 2011; Sindhwani et al., 2012). Our main technical contribution is a unifying study of the minimization problem corresponding to such a regularization framework. More precisely, we devise an optimization approach that can efficiently compute a solution and for which we prove convergence under weak assumptions. Our approach is based on a barrier method that is combined with block coordinate descent techniques (Tseng, 2001; Razaviyayn et al., 2013). In this sense our analysis generalizes the results in (Argyriou et al., 2008a) for which a low-rank assumption was considered; however the extension is not straightforward, since we consider a much larger class of regularization schemes (any convex penalty). Up to our knowledge, this is the first result in multi-task learning proving the convergence of alternating minimization schemes for such a general family of problems.

The RKHSvv setting allows to naturally deal both with linear and non-linear models and the approach we propose provides a general computational framework for learning

*Proceedings of the 32$^{nd}$ International Conference on Machine Learning*, Lille, France, 2015. JMLR: W&CP volume 37. Copyright 2015 by the author(s).

output kernels as formalized in (Dinuzzo et al., 2011).

The rest of the paper is organized as follows: in Sec 2 we review basic ideas of regularization in RKHSvv. In Sec. 2.3 we discuss the equivalence of different approaches to encode known structures among multiple tasks. In Sec. 3 we discuss a general framework for learning multiple tasks and their relations where we consider a wide family of structure-inducing penalties and study an optimization strategy to solve them. This setting allows us, in Sec. 4, to recover several previous methods as special cases. Finally in Sec. 5 we evaluate the performance of the optimization method proposed.

**Notation.** With $S^n_{++} \subset S^n_+ \subset S^n \subset \mathbb{R}^{n\times n}$ we denote respectively the space of positive definite, positive semidefinite (PSD) and symmetric $n \times n$ real-valued matrices. $O^n$ denotes the space of orthonormal $n \times n$ matrices. For any square matrix $M \in \mathbb{R}^{n\times n}$ and $p \geq 1$, we denote by $\|M\|_p = (\sum_{i=1}^n \sigma_i(M)^p)^{1/p}$ the $p$-Schatten norm of $M$, where $\sigma_i(M)$ is the $i$-th largest singular value of $M$. For any $M \in \mathbb{R}^{n\times m}$, $M^\top$ denotes the transpose of $M$. For any PSD matrix $A \in S^n_+$, $A^\dagger$ denotes the pseudoinverse of $A$. We denote by $I_n \in S^n_{++}$ the $n \times n$ identity matrix. The notation $\text{Ran}(M) \subseteq \mathbb{R}^m$ identifies the range of columns of a matrix $M \in \mathbb{R}^{m\times n}$.

## 2. Background

We study the problem of jointly learning multiple tasks by modeling individual task-predictors as the components of a vector-valued function. Let us assume to have $T$ supervised scalar learning problems (or tasks), each with a "training" set of input-output observations $\mathcal{S}_t = \{(x_{it}, y_{it})\}_{i=1}^{n_t}$ with $x_{it} \in \mathcal{X}$ input space and $y_{it} \in \mathcal{Y}$ output space[1]. Given a loss function $\mathcal{L} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$ that measures the per-task prediction errors, we want to solve the following joint regularized learning problem

$$\underset{f\in\mathcal{H}}{\text{minimize}} \sum_{t=1}^T \frac{1}{n_t} \sum_{i=1}^{n_t} \mathcal{L}(y_i^{(t)}, f_t(x_i^{(t)})) + \lambda \|f\|^2_{\mathcal{H}} \quad (1)$$

where $\mathcal{H}$ is an Hilbert space of vector-valued functions $f : \mathcal{X} \to \mathcal{Y}^T$with scalar components $f_t : \mathcal{X} \to \mathcal{Y}$. In order to define a suitable space of hypotheses $\mathcal{H}$, in this section we briefly recall concepts from the theory of reproducing kernel Hilbert spaces for vector-valued functions (RKHSvv) and corresponding regularization theory, which plays a key role in our work. In particular, we focus on a class of reproducing kernels (known as separable kernels) that can be designed to encode specific tasks structures (see (Evgeniou et al., 2005; Argyriou et al., 2013) and Sec. 2.3). Interestingly, separable kernels are related to ideas such as defining a metric on the output space or a label encoding in multi-label problems (see Sec. 2.3)

**Remark 2.1** (Multi-task and multi-label learning). Multi-label learning is a class of supervised learning problems in which the goal is to associate input examples with a label or a set of labels chosen from a discrete set. In general, due to discrete nature of the output space, these problems cannot be solved directly; hence, a so-called *surrogate* problem is often introduced, which is computationally tractable and whose solution allows to recover the solution of the original problem (Steinwart & Christmann, 2008; Bartlett et al., 2006; Mroueh et al., 2012).

Multi-label learning and multi-task learning are strongly related. Indeed, surrogate problems typically consist in a set of distinct supervised learning problems (or tasks) that are solved simultaneously and therefore have a natural formulation in the multi-task setting. For instance, in multi-class classification problems the "One vs All" strategy is often adopted, which consists in solving a set of multiple binary classification problems, one for each class.

### 2.1. Learning Multiple Tasks with RKHSvv

In the scalar setting, reproducing kernel Hilbert spaces have already been proved to be a powerful tool for machine learning applications. Interestingly, the theory of RKHSvv and corresponding Tikhonov regularization scheme follow closely the derivation in the scalar case.

**Definition 2.2.** *Let $(\mathcal{H}, \langle\cdot,\cdot\rangle_{\mathcal{H}})$ be a Hilbert space of functions from $\mathcal{X}$ to $\mathbb{R}^T$. A symmetric, positive definite, matrix-valued function $\Gamma : \mathcal{X} \times \mathcal{X} \to \mathbb{R}^{T\times T}$ is called a reproducing kernel for $\mathcal{H}$ if for all $x \in \mathcal{X}, c \in \mathbb{R}^T$ and $f \in \mathcal{H}$ we have that $\Gamma(x,\cdot)c \in \mathcal{H}$ and the following reproducing property holds $\langle f(x), c\rangle_{\mathbb{R}^T} = \langle f, \Gamma(x,\cdot)c\rangle_{\mathcal{H}}$.*

In analogy to the scalar setting, it can be proved (see (Micchelli & Pontil, 2004)) that the Representer Theorem holds also for regularization in RKHSvv. In particular we have that any solution of the learning problem introduced in Eq. (1) can be written in the form

$$f(x) = \sum_{t=1}^T \sum_{i=1}^{n_t} \Gamma(x, x_i^{(t)}) c_i^{(t)} \quad (2)$$

with $c_i^{(t)} \in \mathbb{R}^T$ coefficient vectors.

The choice of kernel $\Gamma$ induces a joint representation of the inputs as well as a structure among the output components (Álvarez et al., 2012); In the rest of the paper we will focus on so-called separable kernels, where these two aspects are factorized. In Section 3, we will see how separable kernels provide a natural way to learn the tasks structure as well as the tasks.

[1] To avoid clutter in the notation, we have restricted ourselves to the typical situation where all tasks share same input and output spaces, i.e. $\mathcal{X}_t = \mathcal{X}$ and $\mathcal{Y}_t \subseteq \mathbb{R}$.

### 2.2. Separable Kernels

Separable (reproducing) kernels are functions of the form $\Gamma(x, x') = k(x, x')A \;\; \forall x, x' \in \mathcal{X}$ where $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a scalar reproducing kernel and $A \in S_+^T$ is a positive semi-definite (PSD) matrix. In this case, the representer theorem allows to rewrite problem (1) in a more compact matrix notation as

$$\underset{C \in \mathbb{R}^{n \times T}}{\text{minimize}} \;\; V(Y, KCA) + \lambda \, tr(AC^\top KC). \qquad (\mathcal{P})$$

Here $Y \in \mathbb{R}^{n \times T}$ is a matrix with $n = \sum_{t=1}^T n_t$ rows containing the output points; $K \in S_+^n$ is the empirical kernel matrix associated to $k$ and $V : \mathbb{R}^{n \times T} \times \mathbb{R}^{n \times T} \to \mathbb{R}_+$ generalizes the loss in (1) and consists in a linear combination of the entry-wise application of $\mathcal{L}$. Notice that this formulation accounts also the situation where not all training outputs $y^{(t)}$ are observed when a given input $x \in \mathcal{X}$ is provided: in this case the functional $V$ weights 0 the loss values of those entries of $Y$ (and the associated entries of $KCA$) that are not available in training.

Finally, the second term in ($\mathcal{P}$) follows by observing that, for all $f \in \mathcal{H}$ of the form $f(\cdot) = \sum_{i=1}^n k(x_i, \cdot)Ac_i$, the squared norm can be written as $\|f\|^2_{\mathcal{H}} = \sum_{i,j}^n k(x_i, x_j)c_i^\top Ac_j = tr(AC^\top KC)$ where $C \in \mathbb{R}^{n \times T}$ is the matrix with $i$-th row corresponding to the coefficient vector $c_i \in \mathbb{R}^T$ of $f$. Notice that we have re-ordered the index $i$ to be in $\{1, \dots, n\}$ to ease the notation.

### 2.3. Incorporating Known Tasks Structure

Separable kernels provide a natural way to incorporate the task structure when the latter is known a priori. This strategy is quite general and indeed in the following we comment on how the matrix $A$ can be chosen to recover several multi-task methods previously proposed in contexts such as regularization, coding/embeddings or output metric learning, postponing a more detailed discussion in the supplementary material. These observations motivate the extension in Sec. 3 of the learning problem ($\mathcal{P}$) to a setting where it is possible to infer $A$ from the data.

**Regularizers.** Tasks relations can be enforced by devising suitable regularizers (Evgeniou et al., 2005). Interestingly, for a large class of such methods it can be shown that this is equivalent to the choice of the matrix $A$ (or rather its pseudoinverse) (Micchelli & Pontil, 2004). If we consider the squared norm of a function $f = \sum_{i=1}^n k(x_i, \cdot)Ac_i \in \mathcal{H}$ we have (see (Evgeniou et al., 2005))

$$\|f\|^2_{\mathcal{H}} = \sum_{t,s=1}^T A^\dagger_{ts} \langle f_t, f_s \rangle_{\mathcal{H}_k} \qquad (3)$$

where $A_t$ is the $t$-th column of $A$, $\mathcal{H}_k$ is the RKHS associated to the scalar kernel $k$ and $f_t = \sum_{i=1}^n k(x_i, \cdot)A_t^\top c_i \in \mathcal{H}_k$ is the $t$-th component of $f$. The above equation suggests to interpret $A^\dagger$ as the matrix that models the structural relations between tasks by directly coupling different predictors. For instance, by setting $A^\dagger = I_T + \gamma(\mathbf{1}\mathbf{1}^\top)/T$, with $\mathbf{1} \in \mathbb{R}^T$ the vector of all 1s, we have that the parameter $\gamma$ controls the variance $\sum_{t=1}^T \|\bar{f} - f_t\|^2_{\mathcal{H}_k}$ of the tasks with respect to their mean $\bar{f} = \frac{1}{T}\sum_{t=1}^T f_t$. If we have access to some notion of similarity among tasks in the form of a graph with adjacency matrix $W \in S^T$, we can consider the regularizer $\sum_{t,s=1}^T W_{t,s}\|f_t - f_s\|^2_{\mathcal{H}_k} + \gamma \sum_t^T \|f_t\|^2_{\mathcal{H}_k}$ which corresponds to $A^\dagger = L + \gamma I_T$ with $L$ the graph Laplacian induced by $W$.

**Output Metric.** A different approach to model tasks relatedness consists in choosing a suitable metric on the output space to reflect the tasks structure (Lozano & Sindhwani, 2011). Clearly a change of metric on the output space with the standard inner product $\langle y, y\prime \rangle_{\mathbb{R}^T}$ between two output points $y, y\prime \in \mathcal{Y}^T$ corresponds to the choice of a different inner product $\langle y, y\prime \rangle_\Theta = \langle y, \theta y\prime \rangle_{\mathbb{R}^T}$ for some positive definite matrix $\Theta \in S_{++}^T$. Indeed this can be direct related to the choice of a suitable separable kernel. In particular, for the least squares loss function a direct equivalence holds between choosing a metric deformation associated to a $\Theta \in S_{++}^T$ and a separable kernel $k(\cdot, \cdot)I_T$ or use the canonical metric (i.e. with $\Theta = I_T$ the identity) and kernel $k(\cdot, \cdot)\Theta$. The details of this equivalence can be found in the supplementary material.

**Output Representation.** The tasks structure can also be modeled by designing an ad-hoc embedding for the output space. This approach is particularly useful for multi-label scenarios, where output embedding can be designed to encode complex structures such as (e.g. trees, strings, graphs, etc.) (Fergus et al., 2010; Joachims et al., 2009; Crammer & Singer, 2000). Interestingly in these cases, or more generally whenever the embedding map $L : \mathcal{Y}^T \to \widetilde{\mathcal{Y}}$, from the original to the new output space, is linear, then it is possible to show that the learning problem with new code is equivalent to (1) for a suitable choice of separable kernel with $A = L^\top L$. We refer again to the supplementary material for the details of this equivalence.

## 3. Learning the Tasks and their Structure

Clearly, an interesting setting occurs when knowledge of the tasks structure is not available and therefore it is not possible to design a suitable separable kernel. In this case a favorable approach is to infer the tasks relations directly from the data. To this end we propose to consider the following extension of problem ($\mathcal{P}$)

$$\underset{C\in\mathbb{R}^{n\times T},A\in S_+^T}{\text{minimize}} \quad V(Y,KCA)+\lambda tr(AC^\top KC)+F(A), \tag{$\mathcal{Q}$}$$

where the penalty $F:S_+^T\to\mathbb{R}_+$ is designed to learn specific tasks structures encoded in the matrix $A$. The above regularization is general enough to encompass a large number of previously proposed approaches by simply specifying a choice of the scalar kernel and the penalty $F$. A detailed discussion of these connections is postponed to Section 4. In this section, we focus on computational aspects. Throughout, we restrict ourselves to convex loss functions $V$ and convex (and coercive) penalties $F$. In this case, the objective function in ($\mathcal{Q}$) is separately convex in $C$ and $A$ but not jointly convex. Hence, block coordinate methods, which are often used in practice, e.g. alternating minimization over $C$ and $A$, are not guaranteed to converge to a global minimum. Our study provides a general framework to provably compute a solution to problem ($\mathcal{Q}$). First, In Section 3.1, we prove our main results providing a characterization of the solutions of Problem ($\mathcal{Q}$) and studying a barrier method to cast their computation as a convex optimization problem. Second, in Section 3.2, we discuss how block coordinate methods can be naturally used to solve such a problem, analyze their convergence properties and discuss some general cases of interest.

## 3.1. Characterization of Minima and A Barrier Method

We begin, in Section 3.1.1, providing a characterization of the solutions to Problem ($\mathcal{Q}$) by showing that it has an equivalent formulation in terms of the minimization of a convex objective function, namely Problem ($\mathcal{R}$). Depending on the behavior of the objective function on the boundary of the optimization domain, Problem ($\mathcal{R}$) might not be solved using standard optimization techniques. This possible issue motivates the introduction, in Section 3.1.2, of a barrier method; a family of "perturbated" convex programs is introduced whose solutions are shown to converge to those of Problem ($\mathcal{R}$) (and hence of the original ($\mathcal{Q}$)).

### 3.1.1. An Equivalent formulation for ($\mathcal{Q}$)

The objective functional in ($\mathcal{Q}$) is not convex, therefore in principle it is hard to find a global minimizer. As it turns out however, it is possible to circumvent this issue and efficiently find a global solution to ($\mathcal{Q}$). The following result represents a first step in this direction.

**Theorem 3.1.** *Let $K\in S_+^n$ and consider the convex set*

$$\mathcal{C}=\left\{(C,A)\in\mathbb{R}^{n\times T}\times S_+^T \mid \text{Ran}(C^\top KC)\subseteq\text{Ran}(A)\right\}.$$

*Then, for any $F:S_+^T\to\mathbb{R}_+$ convex and coercive, problem*

$$\underset{(C,A)\,\in\,\mathcal{C}}{\text{minimize}} V(Y,KC)+\lambda tr\left(A^\dagger C^\top KC\right)+F(A) \tag{$\mathcal{R}$}$$

*has convex objective function and it is equivalent to ($\mathcal{Q}$). In particular, the two problems achieve the same minimum value and, given a solution $(C_R,A_R)$ for ($\mathcal{R}$), the couple $(C_RA_R^\dagger,A_R)$ is a minimizer for ($\mathcal{Q}$). Vice-versa, given a solution $(C_Q,A_Q)$ for ($\mathcal{Q}$), the couple $(C_QA_Q,A_Q)$ is a minimizer for ($\mathcal{R}$).*

The above result highlights a remarkable connection between the problems ($\mathcal{Q}$) (non-convex) and ($\mathcal{R}$) (convex). In particular, we have the following Corollary, which provides us with a useful characterization of the local minimizers of problem ($\mathcal{Q}$).

**Corollary 3.2.** *Let $Q:\mathbb{R}^{n\times T}\times S_+^T\to\mathbb{R}$ be the objective function of problem ($\mathcal{Q}$). Then, every local minimizer for $Q$ on the open set $\mathbb{R}^{n\times T}\times S_{++}^T$ is also a global minimizer.*

Corollary 3.2 follows from Theorem 3.1 and the fact that, on the restricted domain $\mathbb{R}^{n\times T}\times S_{++}^T$, the map $Q$ is the combination of the objective functional of ($\mathcal{R}$) and the invertible function $(C,A)\longmapsto(CA,A)$. Moreover, if $Q$ is differentiable, i.e. $V$ and the penalty $F$ are differentiable, this is exactly the definition of a *convexifiable* function, which in particular implies *invexity* (Craven, 1995). The latter property ensures that, in the differentiable case, all the *stationary* points (rather than only local minimizers) are global minimizers. This result was originally proved in (Dinuzzo et al., 2011) for the special case of $V$ the least-squares loss and $F(\cdot)=\|\cdot\|_F^2$ the Frobenius norm; Here we have proved its generalization to all convex losses $V$ and penalties $F$.

We end this section adding two comments. First, we note that, while the objective function in Problem ($\mathcal{R}$) is convex, the corresponding minimization problem might not be a convex program (in the sense that the feasible set $\mathcal{C}$ is not identified by a set of linear equalities and non-linear convex inequalities (Boyd & Vandenberghe, 2004)). Second, Corollary (3.2) holds only on the interior of the minimization domain $\mathbb{R}^{n\times T}\times S_+^T$ and does not characterize the behavior of the target functional on its boundary. In fact, one can see that both issues can be tackled defining a *perturbed* objective functional having a suitable behavior on the boundary of the minimization domain. This is the key motivation for the barrier method we discuss in the next section.

### 3.1.2. A Barrier Method to Optimize ($\mathcal{R}$)

Here we propose a barrier approach inspired by the work in (Argyriou et al., 2008a) by introducing a perturbation of problem ($\mathcal{R}$) that enforces the objective functions to be equal to $+\infty$ on the boundary of $\mathbb{R}^{n\times T}\times S_+^T$. As a consequence, each perturbed problem can be solved as a convex optimization constrained on a closed cone. The latter comment is made more precise in the following result that we prove in the supplementary material.

**Theorem 3.3.** *Consider the family of optimization problems*

$$\underset{\substack{C\in\mathbb{R}^{n\times T},\\ A\in S_+^T}}{\text{minimize}} V(Y,KC)+\lambda tr(A^{-1}(C^\top KC+\delta^2 I_T))+F(A) \qquad (\mathcal{S}^\delta)$$

*with $I_T \in S_{++}^T$ the identity matrix. Then, for each $\delta > 0$ the problem ($\mathcal{S}^\delta$) admits a minimum. Furthermore, the set of minimizers for ($\mathcal{S}^\delta$) converges to the set of minimizers for ($\mathcal{R}$) as $\delta$ tends to zero. More precisely, given any sequence $\delta_m > 0$ such that $\delta_m \to 0$ and a sequence of minimizers $(C_m, A_m) \in \mathbb{R}^{n\times T} \times S_+^T$ for ($\mathcal{S}^\delta$), there exists a sequence $(C_m^*, A_m^*) \in \mathbb{R}^{n\times T} \times S_+^T$ of minimizers for ($\mathcal{R}$) such that $\|C_m - C_m^*\|_F + \|A_m - A_m^*\|_F \to 0$ as $m \to +\infty$.*

The barrier $\delta^2 tr(A^{-1})$ is fairly natural and can be seen as preconditioning of the problem leading to favorable computations. The proposed barrier method is similar in spirit to the approach developed in (Argyriou et al., 2008a) and indeed Theorem 3.3 and next Corollary 3.4 are a generalization over the two main results in (Argyriou et al., 2008a) to any convex penalty $F$ on the cone of PSD matrices. However, notice that since we are considering a much wider family of penalties (than the trace norm as in (Argyriou et al., 2008a)) our results cannot directly derived from those in (Argyriou et al., 2008a). In the next section we discuss how to compute the solution of Problem ($\mathcal{S}^\delta$) considering a block coordinate approach.

### 3.2. Block Coordinate Descent Methods

The block variable structure of the objective function in ($\mathcal{S}^\delta$), suggests that it might be beneficial to use block coordinate methods (BCM) (see (Beck & Tetruashvili, 2011)) to solve it. Here with BCM we identify a large class of methods that, in our setting, iterate steps of an optimization on $C$, with $A$ fixed, followed by an optimization of $A$, for $C$ fixed.

A *meta* block coordinate algorithm to solve ($\mathcal{S}^\delta$) is reported in in Alg. 1. Here we interpret each optimization step over $C$ as a supervised step, and each optimization step over $A$ as a an unsupervised step (in the sense that it involves the inputs but not the outputs). Several optimization methods can be used as for SUPERVISEDSTEP and UNSUPERVISEDSTEP in Alg. 1. In particular, the term *Block Coordinate Descent (BCD)* identifies a wide class of iterative methods that perform (typically inexact) minimization of the objective function one block of variables at the time. Different strategies to choose which direction minimize at each step have been proposed: pre-fixed cyclic order, greedy search (Razaviyayn et al., 2013) or randomly, according to a predetermined distribution (Nesterov, 2012).

**Algorithm 1** CONVEX MULTI-TASK LEARNING

```
Input: $K, Y, \epsilon$ tolerance, $\delta$ perturbation parameter, $S$ objective functional of ($\mathcal{S}^\delta$), $V$ loss, $F$ structure penalty.
Initialize: $(C, A) = (C_0, A_0), t = 0$
repeat
    $C_{t+1} \leftarrow$ SUPERVISEDSTEP $(V, K, Y, C_t, A_t)$
    $A_{t+1} \leftarrow$ UNSUPERVISEDSTEP$(F, K, \delta, C_{t+1}, A_t)$
    $t \leftarrow t + 1$
until $|S(C_{t+1}, A_{t+1}) - S(C_t, A_t)| < \epsilon$
```

For a review of several BCD algorithms we refer the reader to (Razaviyayn et al., 2013) and references therein.

A second class of methods is called alternating minimization and corresponds to the situation where at each step in Alg. 1 and exact minimization is performed. This latter approach is favorable when a closed form solution exists for at least one block of variables (see Section 3.2.1) and has been studied extensively in (Tseng, 2001) in the abstract setting where an oracle provides a block-wise minimizer at each iteration. The following Corollary describes the convergence properties of BCD and Alternate minimization sequences provided by applying Alg. 1 to ($\mathcal{S}^\delta$).

**Corollary 3.4.** *Let the Problem ($\mathcal{S}^\delta$) be defined as in Theorem 3.3 then:*

*(a)* ***Alternating Minimization:*** *Let the two procedures in Alg. 1 each provide a block-wise minimizer of the functional with the other block held fixed. Then every limiting point of a minimization sequence provided by Alg. 1, is a global minimizer for ($\mathcal{S}^\delta$).*

*(b)* ***Block Coordinate Descent:*** *Let the two procedures in Alg. 1 each consist in a single step of a first order optimization method (e.g. Projected Gradient Descent, Proximal methods, etc.). Then every limiting point of a minimizing sequence provided by Alg. 1 is a global minimizer for ($\mathcal{S}^\delta$).*

Corollary (3.4) follows by applying previous results on BCD and Alternate minimization. In particular, for the proof of part $(a)$ we refer to Theorem 4.1 in (Tseng, 2001), while for part $(b)$ we refer to Theorem 2 in (Razaviyayn et al., 2013).

In the following we discuss the actual implementation of both SUPERVISED and UNSUPERVISED procedures in the case where $V$ is chosen to be least-squares loss and the penalty $F$ to be a spectral $p$-Schatten norm. This should provide the reader with a practical example of how the meta-algorithm introduced in this section can be specialized to a specific multi-task learning setting.

**Remark 3.5.** (Convergence of Block Coordinate Methods) Several works in multi-task learning have proposed some form of BCM strategy to solve the learning problem. However, up to our knowledge, so far only the authors in (Argyriou et al., 2008a) have considered the issue of convergence to a global optimum. Their results where proved for a specific choice of structure penalty in a framework similar to that of problem ($\mathcal{R}$) (see Section 4) but do not extend straightforwardly to other settings. Corollary 3.4 aims to fill this gap, providing convergence guarantees for block coordinate methods for a large class of multi-task learning problems.

#### 3.2.1. Closed Form solutions for Alternating Minimization: Examples

Here we focus on the alternating minimization case and discuss some settings in which it is possible to obtain a closed form solution for the procedures SupervisedStep and UnsupervisedStep.

**(SupervisedStep) Least Squares.** Let $V$ be the least squares loss and let the structure matrix $A$ be fixed. A closed form solution for the coefficient matrix $C$ returned by the SupervisedStep is (see for instance (Álvarez et al., 2012)):

$$vec(C) = (I_T \otimes K + \lambda A^{-1} \otimes I_n)^{-1} vec(Y),$$

with $\otimes$ the Kronecker product, and $\forall M \in \mathbb{R}^{n \times m}$, $vec(M) \in \mathbb{R}^{nm}$ identifies the concatenation of the columns of $M$.

**(UnsupervisedStep) $p$-Schatten penalties.** We consider the case in which $F$ is chosen to be a spectral penalty of the form $F(\cdot) = \|\cdot\|_p^p$ with $p \geq 1$. Also in this setting the optimization problem has a closed form solution, as shown in the following.

**Proposition 3.6.** *Let the penalty of problem ($\mathcal{S}^\delta$) be $F = \|\cdot\|_p^p$ with $p \geq 1$. Then, for any $C \in \mathbb{R}^{n \times T}$ fixed, the optimization problem ($\mathcal{S}^\delta$) in the block variable $A$ has a minimizer of the form*

$$A_C^\delta = \sqrt[p+1]{(C^\top K C + \delta^2 I_T)/\lambda}. \tag{4}$$

Proposition 3.6 generalizes a similar result originally proved in in (Argyriou et al., 2008a) for the special case $p = 1$ and provides an explicit formula for the UnsupervisedStep of Alg. 1. We report the proof in the supplementary material.

## 4. Previous Work: Comparison and Discussion

Framework ($\mathcal{Q}$) accounts for several choices of losses and task-structural priors. While Sec. 3 has been devoted to deriving optimization procedures to solve such a problem, here we focus on modeling aspects. In particular, we will briefly review some multi-task learning method previously proposed, discussing how they can be formulated as special cases of ($\mathcal{Q}$) (or, equivalently, ($\mathcal{R}$)).

**Spectral Penalties.** The penalty $F = \|\cdot\|_F^2$ was considered in (Dinuzzo et al., 2011), together with a least squares loss function and the non convex problem ($\mathcal{Q}$) is solved directly by alternating minimization. However, as pointed out in Sec. 3, solving the non convex problem (although invex, see the discussion on Corollary 3.2) directly could in principle become problematic when the alternating minimization sequence gets close to the boundary of $\mathbb{R}^{n \times T} \times S_{++}^T$. A related idea is that of considering $F(A) = tr(A)$ (i.e. the 1-Schatten norm). This latter approach can shown to be equivalent to the Multi-Task Feature Learning setting of (Argyriou et al., 2008a) (see supplementary material).

**Cluster Tasks Learning.** In (Jacob et al., 2008), the authors studied a multi-task setting where tasks are assumed to be organized in a fixed number $r$ of unknown disjoint clusters. While the original formulation was conceived for linear setting, it can be easily extended to non-linear kernels and cast in our framework. Let $E \in \{0,1\}^{T \times r}$ be the binary matrix whose entry $E_{st}$ has value $1$ or $0$ depending on whether task $s$ is in cluster $t$ or not. Set $M = I - E^\dagger E^\top$, and $U = \frac{1}{T} 1 1^\top$. In (Jacob et al., 2008) the authors considered a regularization setting of the form of ($\mathcal{R}$) where the structure matrix $A$ is parametrized by the matrix $M$ in order to reflect the cluster structure of the tasks. More precisely:

$$A^{-1}(M) = \epsilon_M U + \epsilon_B (M - U) + \epsilon_W (I - M)$$

where the first term characterizes a global penalty on the average of all tasks predictors, the second term penalizes the between-clusters variance, and the third term controls the tasks variance within each cluster. Clearly, it would be ideal to identify an optimal matrix $A(M)$ minimizing problem ($\mathcal{R}$). However, $M$ belongs to a discrete non convex set, therefore authors propose a convex relaxation by constraining $M$ to be in a convex set $\mathcal{S}_c = \{M \in S_+^T, 0 \preceq M \preceq I, tr(M) = r\}$. In our notations $F(A)$ is therefore the indicator function over the set of all matrices $A = A(M)$ such that $M \in \mathcal{S}_c$. The authors propose a pseudo gradient descent method to solve the problem jointly.

**Convex Multi-task Relation Learning.** Starting from a multi-task Gaussian Process setting, in (Zhang & Yeung, 2010), authors propose a model where the covariance among the coefficient vectors of the $T$ individual tasks is controlled by a matrix $A \in S_{++}^T$ in the form of a prior. The

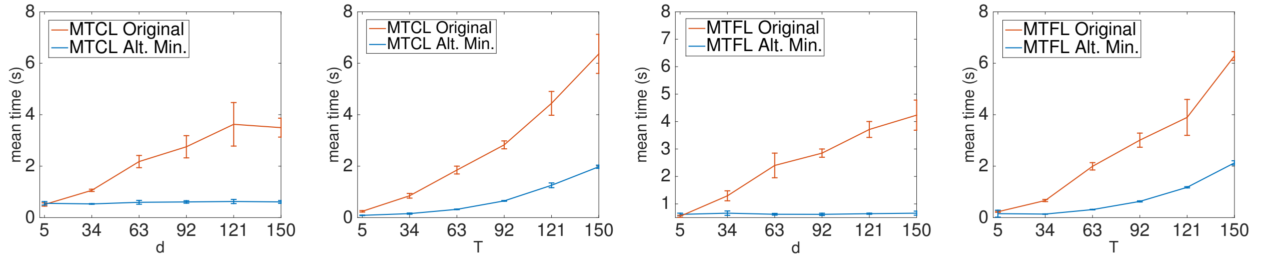

*Figure 1.* Comparison of the computational performance of the alternating minimization strategy studied in this paper with respect to the optimization methods proposed for MTCL in (Jacob et al., 2008) and MTFL (Argyriou et al., 2008a) in the original papers. Experiments are repeated for different number of tasks and input-space dimensions as described in Sec. 5.1.

initial maximum likelihood estimation problem is relaxed to a convex optimization with target functional of the form

$$\|Y - KC\|_F^2 + \lambda_1 \, tr(C^\top KC) + \lambda_2 \, tr(A^{-1}C^\top KC) \quad (5)$$

constrained to the set $\mathcal{A} = \{A \mid A \in S_{++}^T, tr(A) = 1\}$. This setting is equivalent to problem ($\mathcal{R}$) (by choosing $F$ to be the indicator function of $\mathcal{A}$) with the addition of the term $tr(C^\top KC)$.

**Non-Convex Penalties.** Often times, interesting structural assumptions require to impose non-convex penalties to recover interpretable relations among tasks. For instance (Argyriou et al., 2013) requires $A$ to be a graph Laplacian, or (Dinuzzo, 2013) imposes a low-rank factorization of $A$ in two smaller matrices. In (Mroueh et al., 2011; Kumar & Daume III, 2012) different sparsity models are proposed. Most of these methods can be naturally cast in the form ($\mathcal{Q}$) or ($\mathcal{R}$). Unfortunately our analysis of the barrier method does not necessarily hold and Alternating Minimization is not guaranteed to lead to a stationary point.

## 5. Experiments

We empirically evaluated the efficacy of the block coordinate optimization strategy proposed in this paper on both artificial and real datasets. Synthetic experiments were performed to assess the computational aspects of the approach, while we evaluated the quality of solutions found by the system on realistic settings.

### 5.1. Computational Times

As discussed in Sec. 4, several methods previously proposed in the literature, such as Multi-task Cluster Learning (MTCL) (Jacob et al., 2008) and Multi-task Feature Learning (MTFL (Argyriou et al., 2008a)]), can be formulated as special cases of problem ($\mathcal{Q}$) or ($\mathcal{R}$). It is natural to compare the proposed alternating minimization strategy with the optimization solution originally proposed for each method. To assess the system's performance with respect to varying dimensions of the feature space and an increasing number of tasks, we chose to perform this comparison in an artificial setting.

We considered a linear setting where the input data lie in $\mathbb{R}^d$ and are distributed according to a normal distribution with zero mean and identity covariance matrix. $T$ linear models $w_t \in \mathbb{R}^d$ for $t = 1, \dots, T$ were then generated according to a normal distribution in order to sample $T$ distinct training sets, each comprising of 30 examples $(x_i^{(t)}, y_i^{(t)})$ such that $y_i^{(t)} = \langle w_t, x_i^{(t)} \rangle + \epsilon$ with $\epsilon$ Gaussian noise with zero mean and $0.1$ standard deviation. On these learning problems we compared the computational performance of our alternating minimization strategy and the original optimization algorithms originally proposed for MTCL and MTFL and for which the code has been made available by the authors'. In our algorithm we used $A_0 = I$ identity matrix as initialization for the alternating minimization procedure. We used a least-squares loss for all experiments.

Figure 1 reports the comparison of computational times of alternating minimization and the original methods to converge to the same minima (of respectively the functional of MTCL and MTFL). We considered two settings: one where the number of tasks was fixed to $T = 100$ and $d$ increased from 5 to 150 and a second one wher $d$ was fixed to 100 and $T$ varied bewteen 5 and 150. To account for statistical stability we repeated the experiments for each couple $(T, d)$ and different choices of hyperparameters while generating a new random datasets at each time. We can make two observations from these results: 1) in the setting where $T$ is kept fixed we observe a linear increase in the computational times for both original MTCL and MTFL methods, while alternating minimization is almost constant with respect to the input space dimension. 2) When $d$ is fixed and the number of tasks increases, all optimization strategies require more time to converge. This shows that in general alternating minimization is a viable option to solve these problems and in particular, when $T << min(d, n)$ – which is often the case in non-linear settings –this method is particularly efficient.

| | 50 tr. samples per class | | 100 tr. samples per class | | 150 tr. samples per class | | 200 tr. samples per class | |
|---|---|---|---|---|---|---|---|---|
| | **nMSE (± std)** | **nI** | **nMSE (± std)** | **nI** | **nMSE (± std)** | **nI** | **nMSE (± std)** | **nI** |
| **STL** | $0.2436 \pm 0.0268$ | $0$ | $0.1723 \pm 0.0116$ | $0$ | $0.1483 \pm 0.0077$ | $0$ | $0.1312 \pm 0.0021$ | $0$ |
| **MTFL** | $0.2333 \pm 0.0213$ | $0.0416$ | $0.1658 \pm 0.0107$ | $0.0379$ | $0.1428 \pm 0.0083$ | $0.0281$ | $0.1311 \pm 0.0055$ | $0.0003$ |
| **MTRL** | $0.2314 \pm 0.0217$ | $0.0404$ | $0.1653 \pm 0.0112$ | $0.0401$ | $0.1421 \pm 0.0081$ | $0.0288$ | $0.1303 \pm 0.0058$ | $0.0071$ |
| **OKL** | $0.2284 \pm 0.0232$ | $0.0630$ | $0.1604 \pm 0.0123$ | $0.0641$ | $\mathbf{0.1410 \pm 0.0087}$ | $0.0350$ | $0.1301 \pm 0.0073$ | $0.0087$ |

*Table 1.* Comparison of Multi-task learning methods on the Sarcos dataset. The advantage of learning the tasks jointly decreases as more training examples became available.

### 5.2. Real dataset

We assessed the benefit of adopting multi-task learning approaches on two real dataset. In particular we considered the following algorithms: Single Task Learning (STL) as a baseline, Multi-task Feature Learning (MTFL) (Argyriou et al., 2008a), Multi-task Relation Learning (MTRL) (Zhang & Yeung, 2010), Output Kernel Learning (OKL) (Dinuzzo et al., 2011). We used least squares loss for all experiments.

**Sarcos.** Sarcos[2] is a dataset for regression problems (21-dimensional inputs and 7 outputs), which report the corresponding torques measured at each joint.

For each task, we randomly sampled $50, 100, 150$ and $200$ training examples while we kept a test set of $5000$ examples in common for all tasks. We used a linear kernel and performed 5-fold crossvalidation to find the best regularization parameter according to the normalized mean squared error (nMSE) of predicted torques. We averaged the results over $10$ repetitions of these experiments. The results, reported in Table 1, show clearly that to adopt a multi-task approach in this setting is favorable; however, in order to quantify more clearly such improvement, we report in Table 1 also the *normalized improvement* (*nI*) over single-task learning (STL). For each multi-task method MTL, the normalized improvement nI(MTL) is computed as the average

$$\mathrm{nI(MTL)} = \frac{1}{n_{exp}} \sum_{i=1}^{n_{exp}} \frac{\mathrm{nMSE}_i(\mathrm{STL}) - \mathrm{nMSE}_i(\mathrm{MTL})}{\sqrt{\mathrm{nMSE}_i(\mathrm{STL}) \cdot \mathrm{nMSE}_i(\mathrm{MTL})}}$$

over all the $n_{exp} = 10$ experiments of the normalized differences between the nMSE achieved by respectively the STL approach and the given multi-task method MTL.

15**-Scenes.** 15-Scenes[3] is a dataset designed for scene recognition, consisting in a 15-class classification problem. We represented images using LLC coding (Wang et al., 2010) and trained the system on a training set comprising 50, 100 and 150 examples per class. The test set consisted in $7500$ images evenly divided with respect to the $15$ scenes. Table 2 reports the mean classification accuracy on 20 repetitions of the experiments. It can be noticed that while all multi-task approach seem to achieve approximately similar performance, these are consistently outperforming the STL baseline.

| | **Accuracy (%) per # tr. samples per class** | | | | | |
|---|---|---|---|---|---|---|
| | 50 | | 100 | | 150 | |
| **STL** | 72.23 | ±0.04 | 76.61 | ±0.02 | 79.23 | ±0.01 |
| **MTFL** | 73.23 | ±.08 | 77.24 | ±.05 | 80.11 | ±.03 |
| **MTRL** | 73.13 | ±0.08 | 77.53 | ±0.04 | 80.21 | ±0.05 |
| **OKL** | 72.25 | ±0.03 | 77.06 | ±0.01 | 80.03 | ±0.01 |

*Table 2.* Classification results on the 15-scene dataset.

## 6. Conclusions

We have studied a general multi-task learning framework where the tasks structure can be modeled compactly in a matrix. For a wide family of models, the problem of jointly learning the tasks and their relations can be cast as a convex program, generalizing previous results for special cases (Argyriou et al., 2008a; Dinuzzo et al., 2011). Such an optimization can be naturally approached by block coordinate minimization, which can be seen as alternating between supervised and unsupervised learning steps optimizing respectively the tasks or their structure. We evaluated our method real data, confirming the benefit of multi-task learning when tasks share similar properties.

From an optimization perspective, future work will focus on studying the theoretical properties of block coordinate methods, in particular regarding convergence rates. Indeed, the empirical evidence we report suggests that similar strategies can be remarkably efficient in the multi-task setting. From a modeling perspective, future work will focus on studying wider families of matrix-valued kernels, overcoming the limitations of separable ones. Indeed, this would allow to account also for structures in the interaction space between the input and output domains jointly, which is not the case for separable models.

## 7. Acknowledgments

This work was supported by the Center for Brains, Minds and Machines (CBMM), funded by NSF STC award CCF-1231216 and by the Italian Ministry of Education, University and Research FIRB project RBFR12M3AC.

[2] urlhttp://www.gaussianprocess.org/gpml/data/

[3] http://www-cvr.ai.uiuc.edu/ponce_grp/data/

# References

Álvarez, M., Lawrence, N., and Rosasco, L. Kernels for vector-valued functions: a review. *Foundations and Trends in Machine Learning*, 4(3):195–266, 2012. URL http://dx.doi.org/10.1561/2200000036. see also http://arxiv.org/abs/1106.6251.

Argyriou, A., Evgeniou, T., and Pontil, M. Convex multi-task feature learning. *Machine Learning*, 73, 2008a.

Argyriou, Andreas, Maurer, Andreas, and Pontil, Massimiliano. An algorithm for transfer learning in a heterogeneous environment. In *ECML/PKDD (1)*, pp. 71–85, 2008b.

Argyriou, Andreas, Clémençon, Stéphan, and Zhang, Ruocong. Learning the Graph of Relations Among Multiple Tasks. Research report, October 2013. URL https://hal.inria.fr/hal-00940321.

Bakir, G. H., Hofmann, T., Scholkopf, B., Smola, A. J., Taskar, B., and Vishwanathan, S. V. N. Predicting structured data. *MIT Press*, 2007.

Bartlett, Peter L, Jordan, Michael I, and McAuliffe, Jon D. Convexity, classification, and risk bounds. *Journal of the American Statistical Association*, 101(473):138–156, 2006.

Beck, Amir and Tetruashvili, Luba. On the convergence of block coordinate descent type methods. *Technion, Israel Institute of Technology, Haifa, Israel, Tech. Rep*, 2011.

Boyd, Stephen Poythress and Vandenberghe, Lieven. *Convex optimization*. Cambridge university press, 2004.

Crammer, Koby and Singer, Yoram. On the learnability and design of output codes for multiclass problems. In *In Proceedings of the Thirteenth Annual Conference on Computational Learning Theory*, pp. 35–46, 2000.

Craven, BD. Relations between invex properties. *WORLD SCIENTIFIC SERIES IN APPLICABLE ANALYSIS*, 5: 25–34, 1995.

Dinuzzo, F., Ong, C. S., Gehler, P., and Pillonetto, G. Learning output kernels with block coordinate descent. *International Conference on Machine Learning*, 2011.

Dinuzzo, Francesco. Learning output kernels for multi-task problems. *Neurocomputing*, 118:119–126, 2013.

Evgeniou, Theodoros, Micchelli, Charles A, and Pontil, Massimiliano. Learning multiple tasks with kernel methods. In *Journal of Machine Learning Research*, pp. 615–637, 2005.

Fergus, Rob, Bernal, Hector, Weiss, Yair, and Torralba, Antonio. Semantic label sharing for learning with many categories. *European Conference on Computer Vision*, 2010.

Jacob, Laurent, Bach, Francis, and Vert, Jean-Philippe. Clustered multi-task learning: a convex formulation. *Advances in Neural Information Processing Systems*, 2008.

Joachims, Thorsten, Hofmann, Thomas, Yue, Yisong, and Yu, Chun-Nam. Predicting structured objects with support vector machines. *Commun. ACM*, 52(11):97–104, November 2009. ISSN 0001-0782. doi: 10.1145/1592761.1592783. URL http://doi.acm.org/10.1145/1592761.1592783.

Kumar, Abhishek and Daume III, Hal. Learning task grouping and overlap in multi-task learning. *arXiv preprint arXiv:1206.6417*, 2012.

Lozano, A.C. and Sindhwani, V. Block variable selection in multivariate regression and high-dimensional causal inference. *Advances in Neural Information Processing Systems*, 2011.

Micchelli, C. A. and Pontil, M. Kernels for multi-task learning. *Advances in Neural Information Processing Systems*, 2004.

Minh, H. Q. and Sindhwani, V. Vector-valued manifold regularization. *International Conference on Machine Learning*, 2011.

Mroueh, Youssef, Poggio, Tomaso, and Rosasco, Lorenzo. Multi-category and taxonomy learning: A regularization approach. In *NIPS Workshop on Challenges in Learning Hierarchical Models: Transfer Learning and Optimization*, 2011.

Mroueh, Youssef, Poggio, Tomaso, Rosasco, Lorenzo, and Slotine, Jean-jeacques. Multiclass learning with simplex coding. In *Advances in Neural Information Processing Systems*, pp. 2789–2797, 2012.

Nesterov, Yu. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

Razaviyayn, Meisam, Hong, Mingyi, and Luo, Zhi-Quan. A unified convergence analysis of block successive minimization methods for nonsmooth optimization. *SIAM Journal on Optimization*, 23(2):1126–1153, 2013.

Sindhwani, Vikas, Lozano, Aurelie C., and Minh, Ha Quang. Scalable matrix-valued kernel learning and high-dimensional nonlinear causal inference. *CoRR*, abs/1210.4792, 2012.

Steinwart, Ingo and Christmann, Andreas. *Support vector machines*. Springer, 2008.

Tseng, P. Convergence of block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109:475–494, 2001.

Tsochantaridis, Ioannis, Hofmann, Thomas, Joachims, Thorsten, and Altun, Yasemin. Support vector machine learning for interdependent and structured output spaces. *International Conference on Machine Learning*, 2004.

Wang, J., Yang, J., Yu, K., Lv, F., Huang, T., and Gong, Y. Locality-constrained linear coding for image classification. In *CVPR*, 2010.

Zhang, Yu and Yeung, Dit-Yan. A convex formulation for learning task relationships in multi-task learning. In *Proceedings of the Twenty-Sixth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-10)*, pp. 733–742, Corvallis, Oregon, 2010. AUAI Press.