Incremental Learning-to-Learn with Statistical Guarantees

Giulia Denevi[1,2] **Carlo Ciliberto**[3] **Dimitris Stamos**[3] **Massimiliano Pontil** [1,3]
giulia.denevi@iit.it c.ciliberto@ucl.ac.uk d.stamos.12@ucl.ac.uk massimiliano.pontil@iit.it
[1] Computational Statistics and Machine Learning, Istituto Italiano di Tecnologia, 16163 Genova, Italy
[2] Department of Mathematics, University of Genova, 16146 Genova, Italy
[3] Department of Computer Science, University College of London, WC1E 6BT, London, United Kingdom

Abstract

In learning-to-learn the goal is to infer a learning algorithm that works well on a class of tasks sampled from an unknown meta-distribution. In contrast to previous work on batch learning-to-learn, we consider a scenario where tasks are presented sequentially and the algorithm needs to adapt incrementally to improve its performance on future tasks. Key to this setting is for the algorithm to rapidly incorporate new observations into the model as they arrive, without keeping them in memory. We focus on the case where the underlying algorithm is Ridge Regression parametrised by a symmetric positive semidefinite matrix. We propose to learn this matrix by applying a stochastic strategy to minimize the empirical error incurred by Ridge Regression on future tasks sampled from the meta-distribution. We study the statistical properties of the proposed algorithm and prove non-asymptotic bounds on its excess transfer risk, that is, the generalization performance on new tasks from the same meta-distribution. We compare our online learning-to-learn approach with a state-of-the-art batch method, both theoretically and empirically.

1 INTRODUCTION

Learning-to-learn (LTL) or meta-learning aims at finding an algorithm that is best suited to address a class of learning problems (tasks). These tasks are sampled from an unknown meta-distribution and are only partially observed via a finite collection of training examples, see (Baxter, 2000; Maurer, 2005; Thrun & Pratt, 1998) and references therein. This problem plays a large role in artificial intelligence in that it can improve the efficiency of learning from human supervision. In particular, substantial improvement over "learning in isolation" (also known as independent task learning, ITL) is to be expected when the sample size per task is small, a setting which naturally arises in many applications, see e.g. (Camoriano et al., 2017; Rebuffi et al., 2017; Rohrbach et al., 2013).

LTL is particularly appealing when considered from an online or incremental perspective. In this setting, which is sometimes referred to as lifelong learning, see e.g. (Ruvolo & Eaton, 2013), the tasks are observed sequentially – via corresponding sets of training examples – from a common environment and we aim to improve the learning ability of the underlying algorithm on future yet-to-be-seen tasks from the same environment. Practical scenarios of lifelong learning are wide ranging, including computer vision (Rebuffi et al., 2017), robotics (Camoriano et al., 2017), user modelling and many more.

Although LTL is naturally suited for the incremental setting, surprisingly, theoretical investigations are lacking. Previous studies, starting from the seminal paper (Baxter, 2000) and (Maurer, 2009; Maurer et al., 2013; 2016; Pentina & Lampert, 2014), have almost exclusively considered the setting in which the tasks are given in one batch, that is, the meta-algorithm processes multiple datasets from the environment jointly and only once as opposed to sequentially and indefinitely.

The papers (Balcan et al., 2015; Herbster et al., 2016) present results in an online framework which applies to a finite number of tasks using different performance measures. Perhaps most related to our work is (Alquier et al., 2017), where the authors consider a general PAC-Bayesian approach to lifelong learning based on the exponentially weighted aggregation procedure. Unfortunately, this approach is not efficient for large scale applications as it entails storing the entire sequence of datasets during the meta-learning process.

LTL also bears strong similarity to multi-task learning

(MTL), see e.g. (Caruana, 1997), and much work has been done on the theoretical study of both batch (Ando & Zhang, 2005; Maurer et al., 2013) and online (Cavallanti et al., 2010) multi-task learning algorithms. However multi-task learning aims to solve the different problem of learning well on a prescribed set of tasks (the learned model is tested on the same tasks used during training) whereas LTL aims to extrapolate to new tasks.

The principal contribution of this paper is to propose an incremental approach to learning-to-learn and to analyse its statistical guarantees. This incremental approach is appealing in that it efficiently processes one dataset at the time, without the need to store previously encountered datasets. We study in detail the case of linear representation learning, in which an underlying learning algorithm receives in input a sequence of datasets and incrementally updates the data representation so as to better learn future tasks. Following previous work on LTL, e.g. (Baxter, 2000; Maurer, 2009), we measure the performance of the incremental meta-algorithm by the *transfer risk*, namely the average error obtained by running the underlying algorithm with the learned representation, over tasks sampled from the meta-distribution.

Specifically, in this work we choose the underlying algorithm to be Ridge Regression parametrised by a symmetric positive semidefinite matrix. The incremental LTL approach we propose aims at optimizing the *future empirical error* (Maurer, 2009; Maurer et al., 2016) incurred by Ridge Regression over a class of linear representations. For this purpose, we propose to apply Projected Stochastic Subgradient Algorithm (PSSA). We show that the objective function of the resulting meta-algorithm is convex and we give a non-asymptotic convergence rate for the algorithm in high probability. A remarkable feature of our learning bound is that it is comparable to previous bounds for batch LTL. Our proof technique leverages previous work on learning-to-learn (Maurer, 2009) with tools from online convex optimization, see (Cesa-Bianchi et al., 2004; Hazan, 2016) and references therein.

The paper is organized as follows. In Sec. 2, we review the LTL problem and describe in detail the case of linear feature learning with Ridge Regression. In Sec. 3, we present our incremental meta-algorithm for linear feature learning. Sec. 4 contains our bound on the excess transfer risk for the proposed algorithm and in Sec. 5 we compare the bound to a previous bound for the batch setting. In Sec. 6, we report preliminary numerical experiments for the proposed algorithm and, finally, Sec. 7 summarizes the paper and highlight directions of future research. The detailed proofs of the statements in the paper are reported in the appendix.

2 PROBLEM FORMULATION

In the standard independent task learning setting the goal is to learn a functional relation between an input space $\mathcal{X}$ and an output space $\mathcal{Y}$ from a finite number of training examples. More precisely, given a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ measuring prediction errors and given a distribution μ on the joint data space $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, the goal is to find a function $f : \mathcal{X} \to \mathcal{Y}$ minimizing the *expected risk*

$$\mathcal{R}_\mu(f) = \mathbb{E}_{z\sim\mu}\, \ell(f, z) \tag{1}$$

where, with some abuse of notation, for any $z = (x, y) \in \mathcal{Z}$ we denoted $\ell(f, z) = \ell(f(x), y)$. In most practical situations the underlying distribution is *unknown* and the learner is only provided with a finite set $Z = (z_i)_{i=1}^n \in \mathcal{Z}^n$ of observations independently sampled from μ. The goal of a learning algorithm is therefore, given such a *training* dataset Z to return a "good" estimator $A(Z) = f_Z$ whose expected risk is small and tends to the minimum of Eq. (1) as n increases.

A well-established approach to tackle the learning problem is offered by *regularized empirical risk minimization*. This corresponds to the family of algorithms A_ϕ such that, for any $Z \in \mathcal{Z}^n$,

$$A_\phi(Z) = \operatorname*{argmin}_{f\in\mathcal{F}_\phi} \mathcal{R}_Z(f) + \lambda\|f\|^2_{\mathcal{F}_\phi} \tag{2}$$

where $\phi : \mathcal{X} \to \mathcal{F}_\phi$ is a feature map, $\mathcal{F}_\phi$ is the Hilbert space of functions $f : \mathcal{X} \to \mathcal{Y}$ such that $f(x) = \langle f, \phi(x)\rangle_{\mathcal{F}_\phi}$ for any $x \in \mathcal{X}$ and

$$\mathcal{R}_Z(f) = \frac{1}{n}\sum_{i=1}^n \ell(f, z_i)$$

denotes the *empirical risk* of function f on the set Z.

2.1 LINEAR FEATURE LEARNING

In this work we will focus on the case that $\mathcal{Y} \subseteq \mathbb{R}$, $\mathcal{X} \subseteq \mathbb{R}^d$, ℓ is the square loss and $\phi : \mathbb{R}^d \to \mathbb{R}^m$ is a *linear* feature map (also known as a representation), corresponding to the action $\phi(x) = \Phi x$ of a matrix $\Phi \in \mathbb{R}^{m\times d}$ on the input space. It is well known, see e.g. (Argyriou et al., 2008), that, setting $D = \frac{1}{\lambda}\Phi^\top\Phi \in \mathbb{R}^{d\times d}$, any problem of the form in Eq. (2) can be equivalently formulated as

$$A_D(Z) = \operatorname*{argmin}_{w\in\mathrm{Ran}(D)} \mathcal{R}_Z(w) + w^\top D^\dagger w \tag{3}$$

where, with some abuse of notation, we denoted with $\mathcal{R}_Z(w)$ the empirical risk of the linear function $x \mapsto w^\top x$, for any $x \in \mathcal{X}$. Here, $D^\dagger$ denotes the pseudoinverse of D, which is symmetric positive semidefinite (PSD) but not necessarily invertible; when it is not

invertible the constraint requiring w to be in the range $\mathrm{Ran}(D) \subseteq \mathbb{R}^d$ of D is needed to grant the equivalence with Eq. (2). Since for any linear feature map ϕ there exists a symmetric PSD matrix D such that Eq. (2) and Eq. (3) are equivalent, in the following we will refer to D as the *representation* used by algorithm A_D.

2.2 LEARNING TO LEARN D

A natural question is how to choose a good representation D for a given family of related learning problems. In this work we consider the approach of *learning* it from data. In particular, following the seminal work of (Baxter, 2000), we consider a setting where we are provided with an increasing number of tasks and our goal is to find a joint representation D such that the corresponding algorithm A_D is suited to address all such learning problems. The underlying assumption is that all the tasks that we observe *share a common structure* that algorithm A_D can leverage in order to achieve better prediction performance.

More formally, we assume that the tasks we observe are independently sampled from a meta-distribution ρ on the set of probability measures on $\mathcal{Z}$. According to the literature on the topic, see e.g. (Baxter, 2000; Maurer, 2005), we refer to the meta-distribution ρ as the *environment* and we identify each task sampled from ρ by its corresponding distribution μ, from which we are provided with a *training* dataset $Z \sim \mu^n$ of n points sampled independently from μ. While it is possible to consider a more general setting, for simplicity in this work we study the case where for each task we sample the same fixed number n of training points. In line with the independent task learning setting, the goal of a "learning-to-learn" algorithm is therefore to find the best parameter D minimizing the so-called *transfer risk*

$$\mathcal{E}(D) = \mathbb{E}_{\mu\sim\rho}\mathbb{E}_{Z\sim\mu^n} \ \mathcal{R}_\mu\big(A_D(Z)\big) \tag{4}$$

over a set $\mathfrak{D}$ of candidate representations. The term $\mathcal{E}(D)$ is the expected risk that the corresponding algorithm A_D, when trained on the dataset Z, would incur *on average with respect to the distribution of tasks μ induced by ρ*. That is, to compute the transfer risk, we first draw a task $\mu \sim \rho$ and a corresponding n-sample $Z \in \mathcal{Z}^n$ from μ^n, we then apply the learning algorithm to obtain an estimator $A_D(Z)$ and finally we measure the risk of this estimator on the distribution μ.

The problem of minimizing the transfer risk in Eq. (4) given a finite number T of training datasets $Z_1, \ldots, Z_T$ sampled from the corresponding tasks $\mu_1, \ldots, \mu_T$, has been subject of thorough analysis in literature, see e.g. (Baxter, 2000; Maurer, 2005; Maurer et al., 2016). Most work has been focused on the so-called "batch" setting, where all such training datasets are provided at once. However, by its nature, LTL is an ongoing (possibly never ending) process, with training datasets observed a few at the time. In such a scenario the meta-algorithm should allow for an evolving representation D, which improves over time as new datasets are observed. In the following we propose a meta-algorithm to learn D *online* with respect to the tasks, allowing us to transfer past experience about the environment in an efficient manner, *without requiring the memorization of training data*, which could be prohibitive in large scale applications. We will study the statistical guarantees of the proposed algorithm and compare it to its batch counterpart in terms of both theoretical and empirical performance.

2.3 CONNECTION WITH MULTI-TASK LEARNING

LTL is strongly related to *multi-task learning* (MTL) and in fact, as we will see later for the algorithm in Eq. (3), approaches developed for MTL can be used as inspiration to design algorithms for LTL. In multi-task learning a fixed number of tasks $\mu_1, \ldots, \mu_T$ is provided up front and, given T datasets $Z_1, \ldots, Z_T$, each sampled from its corresponding distribution, the goal is to find a joint representation D incurring a small *average expected risk* $\frac{1}{T}\sum_{t=1}^T \mathcal{R}_{\mu_t}(A_D(Z_t))$. In this sense, the main difference between LTL and MTL is that the former aims to guarantee good prediction performance on *future tasks*, while the latter aims to guarantee good prediction performance on the same tasks used to train D.

A well-established approach to MTL is *multi-task feature learning* (Argyriou et al., 2008). This method consists in solving the optimization problem

$$\min_{D\in\mathfrak{D}_\lambda} \frac{1}{T}\sum_{t=1}^T \min_{w\in\mathrm{Ran}(D)} \mathcal{R}_{Z_t}(w_t) + w_t^\top D^\dagger w_t$$

over the set

$$\mathfrak{D}_\lambda = \big\{D \in \mathbb{S}_+^d \mid \mathrm{tr(D)} \leq 1/\lambda\big\} \tag{5}$$

where $\mathbb{S}_+^d$ denotes the set of $d \times d$ symmetric PSD matrices, $\mathrm{tr}(D)$ is the trace of D and λ is a positive parameter which controls the degree of regularization. In the subsequent analysis the parameter λ must be intended as a fixed hyper-parameter, which will be chosen by cross-validation in the experiments. This choice for $\mathfrak{D}_\lambda$ is motivated by the following variational form, see e.g. (Argyriou et al., 2008, Prop. 4.2), of the square trace norm of $W = [w_1, \ldots, w_T] \in \mathbb{R}^{d\times T}$

$$\|W\|_1^2 = \frac{1}{\lambda} \inf_{D\in\mathrm{Int}(\mathfrak{D}_\lambda)} \sum_{t=1}^T w_t^\top D^{-1} w_t$$

where $\mathrm{Int}(\mathfrak{D}_\lambda)$ is the interior of $\mathfrak{D}_\lambda$, namely the set of the symmetric PSD invertible matrices with trace strictly smaller than $1/\lambda$. This leads to the equivalent problem

$$\min_{W \in \mathbb{R}^{d \times T}} \frac{1}{T} \sum_{t=1}^{T} \mathcal{R}_{Z_t}(w_t) + \gamma \|W\|_1^2 \tag{6}$$

with $\gamma = \lambda/T$. The trace norm of a matrix is defined as the sum (ℓ_1-norm) of its singular values, and it is known to induce low-rank solutions for Problem (6). Intuitively, this means that tasks are encouraged to *share a common set of features (or representation)*. In this paper, we adopt this perspective to design our online LTL approach for linear feature learning.

3 ONLINE LEARNING-TO-LEARN

Motivated by the above connection with multi-task learning, we propose an online LTL approach to approximate the solution of the learning problem

$$\min_{D \in \mathfrak{D}_\lambda} \mathcal{E}(D)$$

over the set $\mathfrak{D}_\lambda$ introduced in Eq. (5). We consider the setting in which we are provided with a stream of independent datasets $Z_1, \ldots, Z_T, \ldots$, each sampled from an individual task distribution $\mu_1, \ldots, \mu_T, \ldots$ coming from the environment ρ and our goal is to find an estimator in $\mathfrak{D}_\lambda$ that improves *incrementally* as the number of observed tasks T increases.

3.1 MINIMIZING THE EMPIRICAL TRANSFER RISK

A key observation motivating the online procedure proposed in this work, is that in the *independent task learning* setting, standard results from learning theory, see e.g. (Shalev-Shwartz & Ben-David, 2014), allow one to control the statistical performance of regularized empirical risk minimization, providing bounds on the *generalization error* of A_D as

$$\mathbb{E}_{Z \sim \mu^n} |\mathcal{R}_\mu(A_D(Z)) - \mathcal{R}_Z(A_D(Z))| \leq G(D, n) \tag{7}$$

where $G(\cdot, n)$ is a decreasing function converging to 0 as $n \to +\infty$, while $G(D, \cdot)$ is a measure of complexity of D, which is large for more "expressive" representations and smaller otherwise.

Eq. (7) suggests us to use the empirical risk $\mathcal{R}_Z$ as a proxy for the expected risk $\mathcal{R}_\mu$. Therefore, we introduce the so-called *future empirical risk* (Maurer, 2009; Maurer et al., 2016),

$$\hat{\mathcal{E}}(D) = \mathbb{E}_{\mu \sim \rho} \mathbb{E}_{Z \sim \mu^n} \mathcal{R}_Z(A_D(Z))$$

Algorithm 1 PSSA applied to $\hat{\mathcal{E}}$

Input: T number of tasks, $\lambda > 0$ hyper-parameter, $\{\gamma_t\}_{t \in \mathbb{N}}$ step sizes.
Initialization: $D^{(1)} \in \mathfrak{D}_\lambda$
For $t = 1$ to T:
Sample $\mu_t \sim \rho$, $Z_t \sim \mu_t^n$.
Choose $U_t \in \partial \mathcal{L}_{Z_t}(D^{(t)})$
Update $D^{(t+1)} = \mathrm{proj}_{\mathfrak{D}_\lambda}(D^{(t)} - \gamma_t U_t)$
Return $\bar{D}_T = \frac{1}{T} \sum_{t=1}^{T} D^{(t)}$

and consider the related problem

$$\min_{D \in \mathfrak{D}_\lambda} \hat{\mathcal{E}}(D), \tag{8}$$

which in the sequel, introducing the shorthand notation $\mathcal{L}_Z(D) = \mathcal{R}_Z(A_D(Z))$ for any $D \in \mathbb{S}_+^d$, will be rewritten as

$$\min_{D \in \mathfrak{D}_\lambda} \mathbb{E}_{\mu \sim \rho} \mathbb{E}_{Z \sim \mu^n} \mathcal{L}_Z(D) \tag{9}$$

to highlight the dependency on Z.

Problem (9) can be approached with stochastic optimization strategies. Such methods proceed by sequentially sampling a point (dataset in this case) Z and performing an update step. In recent years, stochastic optimization, finding its origin in the Stochastic Approximation method by (Robbins & Monro, 1951), has been effectively used to deal with large scale applications. We refer to (Nemirovski et al., 2009) for a more comprehensive discussion about this topic. We therefore propose to apply Projected Stochastic Subgradient Algorithm (PSSA) (Shamir & Zhang, 2013), to solve the optimization problem in Eq. (9). The candidate representation coincides in this case with the mean after T iterations $\bar{D}_T$ and it is known as Polyak-Ruppert averaging scheme (Nemirovskii & Yudin, 1985; Polyak & Juditsky, 1992) in the optimization literature. Alg. 1 reports the application of PSSA to $\hat{\mathcal{E}}$ when $\mathcal{L}_Z$ is convex on the set $\mathbb{S}_+^d$. It requires iteratively: i) sampling a dataset Z, ii) performing a step in the direction of a subgradient of $\mathcal{L}_Z$ at the current point, and iii) projecting onto the set $\mathfrak{D}_\lambda$ (which can be done in a finite number of iterations, see Lemma 16 in App. E). Note that in this case, since the function $\mathcal{L}_Z$ is convex, there is no ambiguity in the definition of the subdifferential $\partial \mathcal{L}_Z$, see e.g. (Bertsekas et al., 2003), and we can rely on the convergence of Alg. 1 to a global minimum of $\hat{\mathcal{E}}$ over $\mathfrak{D}_\lambda$ for a suitable choice of step-sizes, as discussed in Sec. 4.

3.2 LTL WITH RIDGE REGRESSION

In this work, we focus on the case that the loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ corresponds to the square loss, namely $\ell(y, y') = (y - y')^2$ for any $y, y' \in \mathcal{Y} \subseteq \mathbb{R}$. In this setting, given a dataset $Z \in \mathcal{Z}^n$, algorithm A_D is equivalent to perform the following variant to *Ridge Regression*

$$\min_{w \in \mathrm{Ran}(D)} \frac{1}{n} \|\mathbf{y} - Xw\|^2 + w^\top D^\dagger w \tag{10}$$

where $X \in \mathbb{R}^{n \times d}$ is the matrix with rows corresponding to the input points $x_i \in \mathbb{R}^d$ in the dataset Z and $\mathbf{y} \in \mathbb{R}^n$ the vector with entries equal to the corresponding output points $y_i \in \mathbb{R}$. The solution to Eq. (10) can be obtained in closed form, in particular, see e.g. (Argyriou et al., 2008; Maurer, 2009),

$$A_D(Z) = DX^\top \big(XDX^\top + nI\big)^{-1} \mathbf{y}. \tag{11}$$

Plugging this solution in the definition of $\mathcal{L}_Z(D)$, a direct computation yields that

$$\mathcal{L}_Z(D) = n \big\| (XDX^\top + nI)^{-1} \mathbf{y} \big\|^2. \tag{12}$$

The following result characterizes some key properties of the function $\mathcal{L}_Z$ in Eq. (12), which will be useful in our subsequent analysis. We denote by $\mathcal{B}_r \subseteq \mathbb{R}^d$ the ball of radius $r > 0$ centered at 0.

Proposition 1 (Properties of $\mathcal{L}_Z$ for the Square Loss)**.** *Let $\mathcal{X} \subseteq \mathcal{B}_1$, $\mathcal{Y} \subseteq [0, 1]$ and ℓ be the square loss. Then, for any dataset $Z \in \mathcal{Z}^n$ the following properties hold:*

1. *$\mathcal{L}_Z$ is convex on the set $\mathbb{S}^d_+$.*
2. *$\mathcal{L}_Z$ is $\mathcal{C}^\infty$ and, for every $D \in \mathbb{S}^d_+$,*

$$\nabla \mathcal{L}_Z(D) = -nX^\top M(D)^{-1} S(D) M(D)^{-1} X$$

where

$$M(D) = XDX^\top + nI$$
$$S(D) = \mathbf{y}\mathbf{y}^\top M(D)^{-1} + M(D)^{-1} \mathbf{y}\mathbf{y}^\top.$$

3. *$\mathcal{L}_Z$ is 2-Lipschitz w.r.t. the Frobenius norm.*
4. *$\nabla \mathcal{L}_Z$ is 6-Lipschitz w.r.t. the Frobenius norm.*
5. *$\mathcal{L}_Z(D) \in [0, 1]$, for any $D \in \mathbb{S}^d_+$.*

The proposition above establishes the convexity of Problem (8) for the case of the square loss. This fact is important in that it guarantees no ambiguity in applying Alg. 1 to our setting and moreover, since $\mathcal{L}_Z$ is differentiable, Alg. 1 becomes a *Projected Stochastic Gradient Algorithm*.

4 THEORETICAL ANALYSIS

In this section, we study the statistical properties of Alg. 1 for the case of the square loss. Below we report the main result of this work, which characterizes the non-asymptotic behavior of the estimator $\bar{D}_T$ produced by Alg. 1 with respect to a minimizer $D_* \in \mathrm{argmin}_{D \in \mathfrak{D}_\lambda} \mathcal{E}(D)$. To present our results we introduce the $d \times d$ matrix $C_\rho = \mathbb{E}_{\mu \sim \rho} \mathbb{E}_{(x,y) \sim \mu} [xx^\top]$ denoting the covariance of the input data, obtained by averaging over all input marginals sampled from ρ. We also denote with $\|C_\rho\|_\infty$ the operator norm of C_ρ, which corresponds to the largest eigen-value.

Theorem 2 (Online LTL Bound)**.** *Let $\mathcal{X} \subseteq \mathcal{B}_1$, $\mathcal{Y} \subseteq [0, 1]$ and ℓ be the square loss. Let $\mu_1, \ldots, \mu_T$ be independently sampled from ρ and Z_t sampled from μ_t^n for $t \in \{1, \ldots, T\}$. Let $\bar{D}_T$ be the output of Alg. 1 with step sizes $\gamma_t = (\lambda \sqrt{2t})^{-1}$. Then, for any $\delta \in (0, 1]$*

$$\mathcal{E}(\bar{D}_T) - \mathcal{E}(D_*) \leq \frac{4\sqrt{2\pi} \|C_\rho\|_\infty^{1/2}}{\sqrt{n}} \frac{1 + \sqrt{\lambda}}{\lambda} + \frac{4\sqrt{2}}{\lambda \sqrt{T}} + \sqrt{\frac{8 \log(2/\delta)}{T}}$$

with probability at least $1 - \delta$ with respect to the independent sampling of the tasks $\mu_t \sim \rho$ and training sets $Z_t \sim \mu_t^n$ for any $t \in \{1, \ldots, T\}$.

In Sec. 5, we will compare Thm. 2 with the statistical bound available for a state-of-the-art LTL batch procedure. We will see that the statistical behaviour of these two approaches is essentially equivalent, with the online LTL approach being more appealing given the lower requirements in terms of both number of computations and memory. In the rest of this section we give a sketch of the proof for Thm. 2. Proofs of intermediate results are reported in the appendix.

4.1 ERROR DECOMPOSITION

The statistical analysis of Alg. 1 hinges upon the following decomposition for the excess transfer risk of the estimator $\bar{D}_T$:

$$\begin{aligned} &\mathcal{E}(\bar{D}_T) - \mathcal{E}(D_*) \\ &\quad = \mathcal{E}(\bar{D}_T) \pm \hat{\mathcal{E}}(\bar{D}_T) \pm \hat{\mathcal{E}}(D_*) - \mathcal{E}(D_*) \\ &\quad \leq 2 \sup_{D \in \mathfrak{D}_\lambda} |\mathcal{E}(D) - \hat{\mathcal{E}}(D)| + \hat{\mathcal{E}}(\bar{D}_T) - \hat{\mathcal{E}}(D_*) \\ &\quad \leq \underbrace{2 \sup_{D \in \mathfrak{D}_\lambda} |\mathcal{E}(D) - \hat{\mathcal{E}}(D)|}_{\text{Uniform generalization error}} + \underbrace{\hat{\mathcal{E}}(\bar{D}_T) - \hat{\mathcal{E}}(\hat{D}_*)}_{\text{Excess future empirical risk}} \end{aligned} \tag{13}$$

where the matrix $\hat{D}_*$ denotes a minimizer of the future transfer risk over $\mathfrak{D}_\lambda$, that is, $\hat{D}_* \in \mathrm{argmin}_{D \in \mathfrak{D}_\lambda} \hat{\mathcal{E}}(D)$.

Eq. (13) decomposes $\mathcal{E}(\bar{D}_T) - \mathcal{E}(D_*)$ in a *uniform generalization error*, implicitly encoding the complexity of the class of algorithms parametrised by D and an *excess future empirical risk*, measuring the discrepancy between the estimator $\bar{D}_T$ and the minimizer $\hat{D}_*$ of $\hat{\mathcal{E}}$. In the following we describe how to bound these two terms.

4.2 BOUNDING THE UNIFORM GENERALIZATION ERROR

Results providing generalization bounds for the class of regularized empirical risk minimization algorithms A_D considered in this work are well known. The following result, which is taken from (Maurer, 2009), leverages an explicit estimate of the generalization bound $G(D, n)$ introduced in Sec. 3.1 for independent task learning, see Eq. (7), to obtain a uniform bound over the class of algorithms parametrized by $\mathfrak{D}_\lambda$.

Proposition 3 (Uniform Generalization Error Bound)**.** *Let $\mathcal{X} \subseteq \mathcal{B}_1$, $\mathcal{Y} \subseteq [0,1]$ and let ℓ be the square loss, then*

$$\sup_{D \in \mathfrak{D}_\lambda} |\mathcal{E}(D) - \hat{\mathcal{E}}(D)| \leq \frac{2\sqrt{2\pi}\|C_\rho\|_\infty^{1/2}}{\sqrt{n}} \frac{1+\sqrt{\lambda}}{\lambda}.$$

For completeness, we report the proof of this proposition in App. B.3.

4.3 BOUNDING THE EXCESS FUTURE EMPIRICAL RISK

Providing bounds for the excess future empirical risk introduced in Eq. (13) consists in studying the convergence rates of Alg. 1 to the minimum of $\hat{\mathcal{E}}$ over $\mathfrak{D}_\lambda$ *in high probability with respect to the sample of T tasks μ_t from ρ and datasets Z_t from μ_t^n for any $t \in \{1, \dots, T\}$.*

To this end, we leverage classical results from the online learning literature (Hazan, 2016). In online learning, the performance of an online algorithm returning a sequence $\{D^{(t)}\}_{t=1}^T$ over T trials is measured in terms of its *regret*, which in the context of this work corresponds to

$$R_T = \frac{1}{T}\sum_{t=1}^T \mathcal{L}_{Z_t}(D^{(t)}) - \min_{D \in \mathfrak{D}_\lambda} \frac{1}{T}\sum_{t=1}^T \mathcal{L}_{Z_t}(D).$$

Differently from the statistical setting considered in this work, in the online setting no assumption is made about the data generation process of $Z_1, \dots, Z_T$, which could be even adversely generated. Therefore, an algorithm that is able to solve the online problem (i.e. if its regret vanishes as $T \to \infty$) can be also expected to solve the corresponding problem in the statistical setting. This is indeed the case for Alg. 1, for which the following lemma provides a non-asymptotic regret bound.

Lemma 4 (Regret Bound for Alg. 1)**.** *Let $\mathcal{X} \subseteq \mathcal{B}_1$, $\mathcal{Y} \subseteq [0,1]$ and ℓ be the square loss. Then the regret of Alg. 1 with step-sizes $\gamma_t = (\lambda\sqrt{2t})^{-1}$ is such that*

$$R_T \leq \frac{4\sqrt{2}}{\lambda\sqrt{T}}.$$

The above lemma is a corollary of Prop. 1 combined with classical results on regret bounds for Projected Online Subgradient Algorithm (Hazan, 2016). We refer the reader to App. D.1 for a more in-depth discussion and for a detailed proof.

In our setting, the datasets $Z_1, \dots, Z_T$ are assumed to be independently sampled from the underlying environment. Combining this assumption with the regret bound in Lemma 4, we can control the excess future empirical risk by means of so-called *online-to-batch conversion* results (Cesa-Bianchi et al., 2004; Hazan, 2016), leading to the following proposition.

Proposition 5 (Excess Future Empirical Risk Bound for Alg. 1)**.** *Let $\mathcal{X} \subseteq \mathcal{B}_1$, $\mathcal{Y} \subseteq [0,1]$ and let ℓ be the square loss. Let $\mu_1, \dots, \mu_T$ be independently sampled from ρ and Z_t sampled from μ_t^n for $t \in \{1, \dots, T\}$. Let $\bar{D}_T$ be the output of Alg. 1 with step sizes $\gamma_t = (\lambda\sqrt{2t})^{-1}$. Then, for any $\delta \in (0,1]$*

$$\hat{\mathcal{E}}(\bar{D}_T) - \hat{\mathcal{E}}(\hat{D}_*) \leq \frac{4\sqrt{2}}{\lambda\sqrt{T}} + \sqrt{\frac{8\log(2/\delta)}{T}}$$

with probability at least $1-\delta$ with respect to the independent sampling of the tasks $\mu_t \sim \rho$ and training sets $Z_t \sim \mu_t^n$ for any $t \in \{1, \dots, T\}$.

The result above follows by combining Prop. 1 with online-to-batch results, see e.g. (Hazan, 2016, Thm. 9.3) and (Cesa-Bianchi et al., 2004). In App. D.2 we provide the complete proof of this statement together with a more detailed discussion about this topic. At this point we are ready to give the proof of Thm. 2.

Proof of Thm. 2. The claim follows by combining Prop. 3 and Prop. 5 in the decomposition of the error $\mathcal{E}(\bar{D}_T) - \mathcal{E}(D_*)$ given in Eq. (13). ■

5 ONLINE LTL VERSUS BATCH LTL

In this section, we compare the statistical guarantees obtained for our online meta-algorithm with a state-of-the-art batch LTL method for linear feature learning. We also comment on the computational cost of both procedures.

5.1 STATISTICAL COMPARISON

Given a finite collection $\mathbf{Z} = \{Z_1, \dots, Z_T\}$ of datasets, a standard approach to approximate a minimizer of the

future empirical risk $\hat{\mathcal{E}}$ is to take a representation $\hat{D}_T$ minimizing the multi-task empirical risk

$$\hat{\mathcal{E}}_{\mathbf{Z}}(D) = \frac{1}{T}\sum_{t=1}^{T} \mathcal{R}_{Z_t}(A_D(Z_t)) \tag{14}$$

over the set $\mathfrak{D}_\lambda$. Such a choice has been extensively studied in the LTL literature (Baxter, 2000; Maurer, 2009; Maurer et al., 2013; 2016). Here we report a result analogous to Thm. 2, characterizing the discrepancy between the transfer risks of $\hat{D}_T$ and D_*.

Theorem 6 (Batch LTL Bound). *Let $\mathcal{X} \subseteq \mathcal{B}_1$, $\mathcal{Y} \subseteq [0,1]$ and let ℓ be the square loss. Let tasks $\mu_1, \dots, \mu_T$ be independently sampled from ρ and Z_t sampled from μ_t^n for $t \in \{1, \dots, T\}$. Let $\hat{D}_T$ be a minimizer of the multi-task empirical risk in Eq. (14) over the set $\mathfrak{D}_\lambda$. Then, for any $\delta \in (0,1]$*

$$\mathcal{E}(\hat{D}_T) - \mathcal{E}(D_*) \leq \frac{4\sqrt{2\pi}\|C_\rho\|_\infty^{1/2}}{\sqrt{n}} \frac{1+\sqrt{\lambda}}{\lambda} + \frac{2\sqrt{2\pi}}{\lambda\sqrt{T}} + \sqrt{\frac{2\log(2/\delta)}{T}}$$

with probability at least $1-\delta$ with respect to the independent sampling of the tasks $\mu_t \sim \rho$ and training sets $Z_t \sim \mu_t^n$ for any $t \in \{1, \dots, T\}$.

The result above is obtained by further decomposing the error $\mathcal{E}(\hat{D}_T) - \mathcal{E}(D_*)$ as done in Eq. (13). In particular, since the multi-task empirical error provides an estimate for the future empirical risk, it is possible to control the overall error by further bounding the term $|\hat{\mathcal{E}}(D) - \hat{\mathcal{E}}_{\mathbf{Z}}(D)|$ uniformly with respect to $D \in \mathfrak{D}_\lambda$. This last result was originally presented in (Maurer, 2009); in App. C we report the complete analysis of such decomposition, leading to the bound in Thm. 6.

5.2 STATISTICAL CONSIDERATIONS

For a fixed value of λ, we can now compare the bounds on the excess transfer risk for the representations resulting from the application of the online procedure (see Thm. 2) and the batch one (see Thm. 6). Since the approximation error due to the choice of λ will be the same for both approaches, this comparison provides a first indication of their statistical behavior. However, it should be kept in mind that we are comparing upper bounds, hence our considerations are not conclusive and further analysis by means of lower bounds for both algorithms would be valuable.

Thm. 2 and Thm. 6 are both composed of three terms. The first term is exactly the same for both procedures and this is obvious looking at the decompositions used to deduce both results. This term can be interpreted as a within-task-estimation error, that depends on the number of points n used to train the underlying learning algorithm (in our case Ridge Regression with a linear feature map). This term, similarly to the MTL setting, highlights the advantage of exploiting the relatedness of the tasks in the learning process in comparison to independent task learning (ITL). Indeed, if the inputs are distributed on a high dimensional manifold, then $\|C_\rho\|_\infty \ll 1$, while upper bounds for ITL have a leading constant of 1. In particular, $\|C_\rho\|_\infty = 1/d$ if the marginal distributions of the tasks are uniform on the $d-1$ dimensional unit sphere; see (Maurer, 2009; Maurer et al., 2016) for a more detailed discussion about this point. The last term in the bounds expresses the dependency on the confidence parameter δ and it is again approximately the same for the batch and the online case. It follows that the main role in the comparison between the online and batch bounds is driven by the middle term, which expresses the dependency of the bound on the number of tasks T. This term originates in different ways: in the batch approach it is derived from the application of uniform bounds and it can be interpreted as an inter-task estimation error, while in the online approach, it plays the role of an optimization error. Despite the different derivations, we can ascertain from the explicit formula of the bounds that this term is approximately the same for both procedures. This is remarkable since it implies that the representation resulting from our online procedure enjoys the same statistical guarantees than the batch one, despite its more parsimonious memory and computational requirements.

5.3 COMPUTATIONAL CONSIDERATIONS

After discussing the theoretical comparison between the online and the batch LTL approach, in this section we point out some key aspects regarding the computational costs of both procedures.

Memory. The batch LTL estimator corresponds to a minimizer of the multi-task empirical risk in Eq. (14) over *all tasks observed so far*. The corresponding approach therefore requires storing in memory all training datasets as they arrive in order to perform the optimization. This is clearly not sustainable in the incremental setting, since tasks are observed sequentially and, possibly indefinitely, inevitably leading to a memory overflow. On the contrary, in line with stochastic methods, online LTL has a small memory footprint, since it requires to store only one dataset at the time, allowing to "forget" it as soon as one gradient step is performed.

Time. Online LTL is also advantageous in terms of the number of iterations performed whenever a new task is observed. Indeed, for every new task, online LTL per-

forms *only one step* of gradient descent for a total of T steps after T tasks. On the contrary, batch LTL requires finding a minimizer for Eq. (14), which cannot be obtained in closed form but requires adopting an iterative method such as Projected Gradient Descent, see e.g. (Combettes & Wajs, 2005). These methods typically require k iterations to achieve an error of the order of $O(1/k)$ from the optimum (better rates are possible adopting accelerated schemes). However, since for any new task batch LTL needs to find a minimizer for the multi-task empirical error from scratch, this leads to a total of Tk iterations after T tasks. Noting that every such iteration requires to compute T gradients of $\mathcal{L}_Z$ in contrast to the single one of PSSA, this shows that online LTL requires much less operations. In the batch case, a "warm-restart" strategy can be adopted to initialize the Projected Gradient Descent with the representation learned during the previous step, however, as we empirically observed in Sec. 6, online LTL is still significantly faster than batch.

6 EXPERIMENTS

In this section, we report preliminary empirical evaluations of the online LTL strategy proposed in this work; the Python implementation of our algorithm is available at *https://github.com/dstamos*. In particular we compare our method with its batch (or offline) counterpart and independent task learning (ITL), i.e. standard Ridge Regression, which does not leverage any shared structure among the tasks.

In all experiments, we obtain the online and batch estimators $\bar{D}_{\lambda,T_{\mathrm{tr}}}$ and $\hat{D}_{\lambda,T_{\mathrm{tr}}}$ by learning them on a dataset $\mathbf{Z}_{\mathrm{tr}}$ of T_{tr} *training* tasks, each comprising n input-output pairs $(x, y) \in \mathcal{X} \times \mathcal{Y}$. Below to simplify our notation we omit the subscript T_{tr} in these estimators. We perform this training for different values of $\lambda \in \{\lambda_1, \ldots, \lambda_p\}$ and select the best estimator based on the prediction error measured on a separate set $\mathbf{Z}_{\mathrm{va}}$ of T_{va} *validation* tasks. Once such optimal λ value has been selected, we report the generalization performance of the corresponding estimator on a set $\mathbf{Z}_{\mathrm{te}}$ of T_{te} *test* tasks. Note that the tasks in the test and validation sets $\mathbf{Z}_{\mathrm{te}}$ and $\mathbf{Z}_{\mathrm{va}}$ are all provided with both a training and test datasets $Z, Z' \in \mathcal{Z}^n$. Indeed, in order to evaluate the performance of a representation D, we need to first train the corresponding algorithm A_D on Z, and then test its performance on Z' (sampled from the same distribution), by computing the empirical risk $\mathcal{R}_{Z'}(A_D(Z))$. For all methods considered in this setting, we perform parameter selection over $p = 30$ candidate values of λ over the range $[10^{-6}, 10^3]$ with logarithmic spacing. In the online setting the training datasets arrive one at the time, therefore model selection is performed *online*: the system keeps track of all candidate representation matrices $\bar{D}_{\lambda_1}, \ldots, \bar{D}_{\lambda_m}$ and whenever a new training task is presented, these matrices are all updated by incorporating the corresponding new observations. The best representation is then returned at each iteration, based on its performance on the validation set $\mathbf{Z}_{\mathrm{va}}$. Finally, in the subsequent experiments, we set the step sizes of the online LTL method in Alg. 1 equal to $\gamma_t = c/\sqrt{t}$, for some constant $c > 0$ chosen by model selection. Moreover, we computed the batch LTL estimator by classical Projected Gradient Descent method up to convergence, within 10^{-6} relative descent of the objective function.

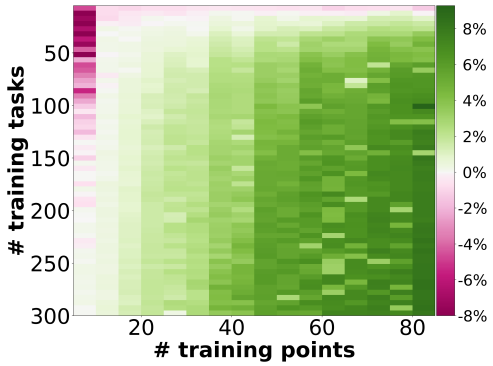

Figure 1: Relative improvement (in %) of our online LTL algorithm over the ITL baseline for a varying range of training tasks T_{tr} and number of samples n per task, during 30 trials.

Synthetic Data. We considered a regression problem on $\mathcal{X} \subseteq \mathbb{R}^d$ with $d = 50$ and a variable number of training tasks T_{tr} and training points n. We also generated $T_{\mathrm{te}} = 300$ test tasks and we sampled a number T_{va} of validation tasks equal to 50% of T_{tr}. For each task, the corresponding dataset $(x_i, y_i)_{i=1}^n$ was generated according to the linear regression equation $y = w^\top x + \epsilon$, with x sampled uniformly on the unit sphere in $\mathbb{R}^d$ and ϵ sampled from a Normal distribution, $\epsilon \sim \mathcal{N}(0, 0.2)$. The tasks predictors w were generated as $P\tilde{w}$ with the components of $\tilde{w} \in \mathbb{R}^{d/2}$ sampled from $\mathcal{N}(0, 1)$ and then $\tilde{w}$ normalized to have unit norm, with $P \in \mathbb{R}^{d \times d/2}$ a matrix with orthonormal rows. In this way, the tasks reflect the assumption of sharing a low dimensional representation, which needs to be inferred by the LTL algorithm.

Fig. 1 reports the comparison between the baseline ITL and the proposed online LTL approach in terms of the relative difference of the prediction error on test tasks for the two methods. More precisely, given the mean squared errors (MSE) R_{oLTL} of online LTL and R_{ITL} of ITL averaged across the test tasks, we report the ratio $(R_{\mathrm{ITL}} - R_{\mathrm{oLTL}})/R_{\mathrm{ITL}}$ as a percentage improvement. Results are reported across a range of T_{tr} and n. We note that the regime considered for these experiments is par-

Table 1: Time (in seconds) for computing online and batch LTL for T_{tr} training tasks and n of samples per task.

T_{tr}	50		100		150	
n	20	50	20	50	20	50
Batch	85	227	246	617	428	2003
Online	36	86	108	273	227	776

ticularly favorable to LTL, almost always outperforming ITL. However, when the number of training points per task is small, the LTL algorithm, as expected, is unable to capture the underlying representation, unless several tasks are used in training.

To provide further evidence of the performance of online LTL, Fig. 2 (Top) compares the prediction error of online LTL, batch LTL, and ITL as the number of training tasks T_{tr} increases one at the time and the different methods update their corresponding representation accordingly. In this case, the number of samples per task is fixed to $n = 40$. We also added to the comparison the multi-task algorithm (MTL) described in Sec. 2.3, performing trace norm regularization *on the test set*. As expected, the performance of both ITL and MTL does not depend on the number of training tasks. Consistently to what observed before, ITL is outperformed by both LTL methods, which tend to converge to the MTL method as more training tasks are provided. In general, when, as in this case, the number of test tasks is large enough, the MTL method is expected to outperform LTL, since MTL optimizes the representation directly on the test tasks. Concerning the LTL methods, consistently with the theory presented in Sec. 4, the performance of the online method is equivalent to that of its batch counterpart, which is, as already stressed in Sec. 5.3, less appealing from the computational point of view. To confirm this aspect, we report in Tab. 1 the computational times required on average by online LTL and batch LTL as T_{tr} and n vary. Online LTL is faster than batch LTL.

Schools Dataset. We evaluated online LTL on the Schools dataset, consisting of examination records from 139 schools, see (Argyriou et al., 2008). Each school is associated to a regression task, individual students correspond to the input and their exam score to the output. In this case, the sample size n varies across the tasks and the features belong to an input space $\mathcal{X} \subseteq \mathbb{R}^d$, with $d = 26$. We randomly sampled 25% and 50% of the 139 tasks for LTL training and validation respectively and the remaining tasks were used as test set. Fig. 2 (Bottom) reports the performance of online LTL, batch LTL, ITL and MTL. Performance is reported in terms of the Explained Variance on the tasks (Argyriou et al., 2008), higher values correspond to better performance. Results are consistent with synthetic experiments; in particular, online and batch LTL are comparable.

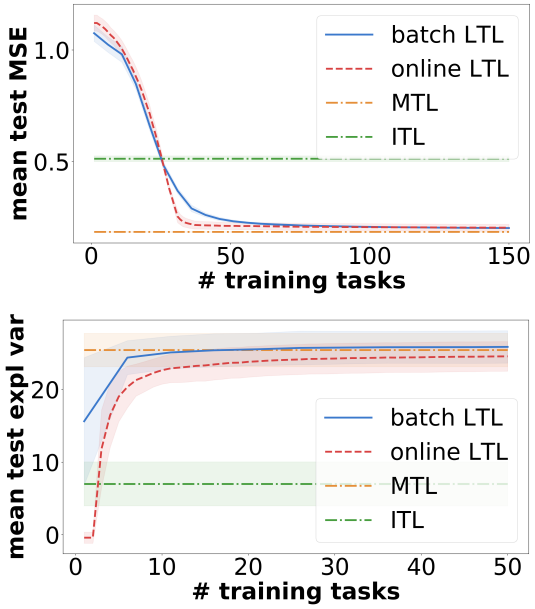

Figure 2: Performance of online LTL, batch LTL, ITL and MTL (on the test set) during 30 trials on the synthetic dataset (**Top**) and the Schools dataset (**Bottom**) as the number of training tasks increases incrementally.

7 CONCLUSION AND FUTURE WORK

We proposed an on-line (incremental) approach to LTL for linear data representation learning. Compared with its batch counterpart, this approach is computationally more efficient both in terms of memory and number of operations, while enjoying the same generalization properties. Preliminary experiments have highlighted the favorable learning capability of the proposed LTL strategy. Our analysis opens several future research directions. First, it would be valuable to investigate whether the same statistical guarantees hold for a projection-free meta-algorithm which does not require the computation of the entire SVD (e.g. certain variants of Frank Wolfe algorithm (Hazan & Kale, 2012), which do not require memorizing the sequence of datasets). Second, from a modeling perspective, we could take inspiration from the vast MTL literature to design new LTL methods in order to deal with tasks that are not necessarily spanning a low-rank subspace but are for instance organized into clusters (Jacob et al., 2009) or share a sparse set of relations (Ciliberto et al., 2015a;b). Finally, extending our analysis to non-convex settings would allow one to tackle more general families of learning algorithms as well as recent empirical meta-learning approaches (e.g. Franceschi et al., 2018) which implicitly attempt to directly minimize the transfer risk.

References

Alquier, Pierre, Mai, The Tien, and Pontil, Massimiliano. Regret bounds for lifelong learning. In *International Conference on Artificial Intelligence and Statistics*, 2017.

Ando, Rie Kubota and Zhang, Tong. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 2005.

Argyriou, Andreas, Evgeniou, Theodoros, and Pontil, Massimiliano. Convex multi-task feature learning. *Machine Learning*, 2008.

Balcan, Maria-Florina, Blum, Avrim, and Vempala, Santosh. Efficient representations for lifelong learning and autoencoding. In *Conference on Learning Theory*, 2015.

Bartlett, Peter L and Mendelson, Shahar. Rademacher and gaussian complexities: Risk bounds and structural results. *Journal of Machine Learning Research*, 3:463–482, 2002.

Bauschke, Heinz H, Combettes, Patrick L, et al. *Convex analysis and monotone operator theory in Hilbert spaces*. Springer.

Baxter, Jonathan. A model of inductive bias learning. *J. Artif. Intell. Res.*, 12(149–198):3, 2000.

Bertsekas, Dimitri P, Nedi, Angelia, and Ozdaglar, Asuman. *Convex analysis and optimization*. Athena Scientific, 2003.

Bhatia, Rajendra. Matrix analysis, volume 169 of graduate texts in mathematics, 1997.

Boucheron, Stéphane, Lugosi, Gábor, and Bousquet, Olivier. Concentration inequalities. In *Advanced Lectures on Machine Learning*, pp. 208–240. Springer, 2004.

Camoriano, Raffaello, Pasquale, Giulia, Ciliberto, Carlo, Natale, Lorenzo, Rosasco, Lorenzo, and Metta, Giorgio. Incremental robot learning of new objects with fixed update time. In *ICRA*, 2017.

Caruana, Rich. Multitask learning. *Machine Learning*, 1997.

Cavallanti, Giovanni, Cesa-Bianchi, Nicolo, and Gentile, Claudio. Linear algorithms for online multitask classification. *Journal of Machine Learning Research*, 2010.

Cesa-Bianchi, Nicolo, Conconi, Alex, and Gentile, Claudio. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 2004.

Ciliberto, Carlo, Mroueh, Youssef, Poggio, Tomaso, and Rosasco, Lorenzo. Convex learning of multiple tasks and their structure. In *ICML*, 2015a.

Ciliberto, Carlo, Rosasco, Lorenzo, and Villa, Silvia. Learning multiple visual tasks while discovering their structure. In *CVPR*, 2015b.

Combettes, Patrick L and Wajs, Valérie R. Signal recovery by proximal forward-backward splitting. *Multiscale Modeling & Simulation*, 4(4):1168–1200, 2005.

Franceschi, Luca, Frasconi, Paolo, Grazzi, Riccardo, Salzo, Saverio, and Pontil, Massi. Bilevel programming for hyperparameter optimization and meta-learning. In *ICML*, 2018.

Grimmett, Geoffrey and Stirzaker, David. *Probability and random processes*. Oxford university press, 2001.

Hazan, Elad. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2016.

Hazan, Elad and Kale, Satyen. Projection-free online learning. *arXiv preprint arXiv:1206.4657*, 2012.

Herbster, Mark, Pasteris, Stephen, and Pontil, Massimiliano. Mistake bounds for binary matrix completion. In *Advances in Neural Information Processing Systems*, 2016.

Jacob, Laurent, Vert, Jean-philippe, and Bach, Francis R. Clustered multi-task learning: A convex formulation. In *Advances in neural information processing systems*, 2009.

Kollo, Tonu and von Rosen, Dietrich. *Advanced Multivariate Statistics with Matrices*, volume 579. Springer Science & Business Media, 2006.

Maurer, Andreas. Algorithmic stability and meta-learning. *Journal of Machine Learning Research*, 2005.

Maurer, Andreas. Transfer bounds for linear feature learning. *Machine learning*, 75(3):327–350, 2009.

Maurer, Andreas, Pontil, Massi, and Romera-Paredes, Bernardino. Sparse coding for multitask and transfer learning. In *ICML*, 2013.

Maurer, Andreas, Pontil, Massimiliano, and Romera-Paredes, Bernardino. The benefit of multitask representation learning. *The Journal of Machine Learning Research*, 17(1): 2853–2884, 2016.

McDonald, Andrew M, Pontil, Massimiliano, and Stamos, Dimitris. New perspectives on k-support and cluster norms. *Journal of Machine Learning Research*, 2016.

Nemirovski, Arkadi, Juditsky, Anatoli, Lan, Guanghui, and Shapiro, Alexander. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 2009.

Nemirovskii, A. and Yudin, D. B. Problem complexity and method efficiency in optimization. *SIAM Review*, 1985.

Pentina, Anastasia and Lampert, Christoph. A PAC-Bayesian bound for lifelong learning. In *International Conference on Machine Learning*, pp. 991–999, 2014.

Petersen, Kaare Brandt and Pedersen, Michael Syskind. The matrix cookbook. *Technical University of Denmark*, 2008.

Polyak, Boris T and Juditsky, Anatoli B. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992.

Rebuffi, Sylvestre-Alvise, Kolesnikov, Alexander, and Lampert, Christoph H. iCaRL: Incremental classifier and representation learning. In *Proc. CVPR*, 2017.

Robbins, Herbert and Monro, Sutton. A stochastic approximation method. *The Annals of Mathematical Statistics*, 1951.

Rohrbach, Marcus, Ebert, Sandra, and Schiele, Bernt. Transfer learning in a transductive setting. In *Advances in Neural Information Processing Systems*, pp. 46–54, 2013.

Ruvolo, Paul and Eaton, Eric. Ella: An efficient lifelong learning algorithm. In *ICML*, 2013.

Shalev-Shwartz, Shai and Ben-David, Shai. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014.

Shamir, Ohad and Zhang, Tong. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In *ICML*, 2013.

Thrun, Sebastian and Pratt, Lorien. *Learning to Learn*. Springer, 1998.

APPENDIX

A PROOF OF Prop. 1

We denote by $\mathbb{S}^d$, $\mathbb{S}^d_+$ and $\mathbb{S}^d_{++}$ the sets of symmetric, positive semidefinite (PSD) and positive definite $d \times d$ real matrices, respectively. We denote by $\langle \cdot, \cdot \rangle$ the standard inner product in $\mathbb{R}^d$ (or $\mathbb{R}^n$, depending on the context) and by $\|\cdot\|$ the associated norm. For any $p \in [1, \infty]$, the p-Schatten norm of a matrix will be denoted by $\|\cdot\|_p$. Note that $\|\cdot\|_1$, $\|\cdot\|_2$ and $\|\cdot\|_\infty$ are the trace, Frobenius and spectral norms, respectively.

Recall the definition of the function $\mathcal{L}_Z$ in Eq. (12). In order to provide the proof of Prop. 1 we need the following Lemma, see (Maurer, 2005, Lemma 11).

Lemma 7. *If $G_1, G_2 \in \mathbb{S}^d_+$, then for any $\gamma > 0$ and for $i = 1, 2$, the following points hold.*

(a) *$G_i + \gamma I$ is invertible.*

(b) $\left\|\left(G_i + \gamma I\right)^{-1}\right\|_\infty \leq \gamma^{-1}$.

(c) $\left\|\left(G_1 + \gamma I\right)^{-1} - \left(G_2 + \gamma I\right)^{-1}\right\|_\infty \leq \gamma^{-2}\left\|G_1 - G_2\right\|_\infty$.

(d) *Let w_1 and w_2 satisfy $\left(G_i + \gamma I\right) w_i = \mathbf{y}$ for some $\mathbf{y}$, for $i = 1, 2$. Then we have that*

$$\left| \|w_1\|^2 - \|w_2\|^2 \right| \leq 2\gamma^{-3}\left\|G_1 - G_2\right\|_\infty \|\mathbf{y}\|^2.$$

Proof of Prop. 1. We now prove each point in turn.

1. Recall that a function $h : \mathbb{S}^d \to \mathbb{S}^d$ is matrix-convex if for every $A, B \in \mathbb{S}^d$ and $\lambda \in [0, 1]$, $h(\lambda A + (1 - \lambda)B) \preceq \lambda h(A) + (1 - \lambda)h(B)$, see e.g. (Bhatia, 1997, Chap. V). The function $h(A) = A^{-2}$ is matrix convex on $\mathbb{S}^d_{++}$. It follows, for every $\mathbf{y} \in \mathbb{R}^n$, that the real-valued function $g_\mathbf{y} : \mathbb{S}^d_{++} \to \mathbb{R}$ defined at $A \in \mathbb{S}^d_{++}$ as $g_\mathbf{y}(A) = \langle \mathbf{y}, A^{-2}\mathbf{y} \rangle$ is convex. By Eq. (12), we have that $\mathcal{L}_Z(D) = g_\mathbf{y}(XDX^\top + nI)$, hence it is convex because it is the composition of the convex function $g_\mathbf{y}$ with an affine function.

2. Since the function $\mathcal{L}_Z$ in Eq. (12) is the composition of $\mathcal{C}^\infty$ functions, it is itself $\mathcal{C}^\infty$ on $\mathbb{S}^d_+$; therefore, as soon as we restrict it to a bounded subset of $\mathbb{S}^d_+$, all its derivatives[1] become Lipschitz. In this section we will use formula deriving from matrix calculus, we refer to the books (Kollo & von Rosen, 2006; Petersen & Pedersen, 2008) for more details. Recalling the notation $M(D) = XDX^\top + nI \in \mathbb{R}^{n \times n}$, we now compute the Jacobian of the function $\mathcal{L}_Z$. Denoting by x^k the k-th column of the matrix X (it will be a column vector) for $k = 1, \ldots, d$, we first show, for every $i, j \in \{1, \ldots, d\}$, that

$$\begin{aligned} \left[\nabla \mathcal{L}_Z(D)\right]_{i,j} &= -n \operatorname{tr}\Big(\mathbf{y}\mathbf{y}^\top M(D)^{-1}\Big(x^i {x^j}^\top M(D)^{-1} + M(D)^{-1} x^i {x^j}^\top\Big) M(D)^{-1}\Big) \\ &= -n\Big\langle \mathbf{y}, M(D)^{-1}\Big(x^i {x^j}^\top M(D)^{-1} + M(D)^{-1} x^i {x^j}^\top\Big) M(D)^{-1}\mathbf{y}\Big\rangle. \end{aligned} \tag{15}$$

   To see this, we first exploit the cyclic property of the trace to rewrite, for any $Z \in \mathcal{Z}^n$ and $D \in \mathbb{S}^d_+$, the function $\mathcal{L}_Z$ in Eq. (12) as

$$\mathcal{L}_Z(D) = n\left\langle \mathbf{y}, M(D)^{-2}\mathbf{y}\right\rangle = n \operatorname{tr}\Big(\mathbf{y}^\top M(D)^{-2}\mathbf{y}\Big) = n \operatorname{tr}\Big(\mathbf{y}\mathbf{y}^\top M(D)^{-2}\Big) = n\, f\big(U(D)\big)$$

   where for any matrix $V \in \mathbb{R}^{n \times n}$ we have introduced the function $f(V) = \operatorname{tr}(\mathbf{y}\mathbf{y}^\top V)$ and the symmetric matrix $U(D) = M(D)^{-2} \in \mathbb{R}^{n \times n}$. Hence, since $\dfrac{\partial f(V)}{\partial V} = \mathbf{y}\mathbf{y}^\top$ for any symmetric V (Petersen & Pedersen, 2008, Eq. (93)), thanks to the chain rule (Petersen & Pedersen, 2008, Eq. (126)), for any $i, j \in \{1, \ldots, d\}$, we have that

$$\frac{\partial \mathcal{L}_Z(D)}{\partial D_{ij}} = n \operatorname{tr}\Big(\frac{\partial f(U(D))}{\partial U(D)}^\top \frac{\partial U(D)}{\partial D_{ij}}\Big) = n \operatorname{tr}\Big(\mathbf{y}\mathbf{y}^\top \frac{\partial U(D)}{\partial D_{ij}}\Big) = n\left\langle \mathbf{y}, \frac{\partial U(D)}{\partial D_{ij}}\mathbf{y}\right\rangle.$$

[1] On the boundary of the set we define the derivatives by continuity.

Moreover, the following formula, which is a direct consequence of (Petersen & Pedersen, 2008, Eq. (33),Eq. (53)), holds:

$$\frac{\partial M(D)^{-2}}{\partial D_{i,j}} = -M(D)^{-1}\Big(\frac{\partial M(D)}{\partial D_{i,j}}M(D)^{-1} + M(D)^{-1}\frac{\partial M(D)}{\partial D_{i,j}}\Big)M(D)^{-1} \tag{16}$$

and, for every $k, h \in \{1, \dots, n\}$, we have that

$$\Big[\frac{\partial M(D)}{\partial D_{i,j}}\Big]_{kh} = \Big[\frac{\partial \big(XDX^\top\big)}{\partial D_{i,j}}\Big]_{kh} = x^i_k x^j_h = \big[x^i x^{j\top}\big]_{kh}.$$

Hence, substituting in Eq. (16) we obtain:

$$\frac{\partial U(D)}{\partial D_{i,j}} = \frac{\partial M(D)^{-2}}{\partial D_{i,j}} = -M(D)^{-1}\Big(x^i x^{j\top} M(D)^{-1} + M(D)^{-1} x^i x^{j\top}\Big)M(D)^{-1}$$

and this conclude the proof of Eq. (15). Now, using the fact that for two $n \times 1$ vectors v and w, we have that $x^{i\top} v, x^{j\top} w \in \mathbb{R}$ and

$$\big(x^{i\top} v\big)\big(x^{j\top} w\big) = \big[X^\top v\big]_i \big[X^\top w\big]_j = \big[X^\top v w^\top X\big]_{ij},$$

and exploiting the symmetry of $M(D)$, we can rewrite:

$$\begin{aligned}
\big[\nabla \mathcal{L}_Z(D)\big]_{i,j} &= -n\Big\langle \mathbf{y}, M(D)^{-1}\Big(x^i x^{j\top} M(D)^{-1} + M(D)^{-1} x^i x^{j\top}\Big)M(D)^{-1}\mathbf{y}\Big\rangle \\
&= -n\Big\langle \mathbf{y}, M(D)^{-1} x^i x^{j\top} M(D)^{-2}\mathbf{y}\Big\rangle - n\Big\langle \mathbf{y}, M(D)^{-2} x^i x^{j\top} M(D)^{-1}\mathbf{y}\Big\rangle \\
&= -n\Big(x^{i\top}\underbrace{M(D)^{-1}\mathbf{y}}_{v}\Big)\Big(x^{j\top}\underbrace{M(D)^{-2}\mathbf{y}}_{w}\Big) - n\Big(x^{i\top}\underbrace{M(D)^{-2}\mathbf{y}}_{v}\Big)\Big(x^{j\top}\underbrace{M(D)^{-1}\mathbf{y}}_{w}\Big) \\
&= -n\Big[X^\top M(D)^{-1}\mathbf{y}\mathbf{y}^\top M(D)^{-2} X\Big]_{ij} - n\Big[X^\top M(D)^{-2}\mathbf{y}\mathbf{y}^\top M(D)^{-1} X\Big]_{ij} \\
&= -n\Big[X^\top M(D)^{-1}\Big(\mathbf{y}\mathbf{y}^\top M(D)^{-1} + M(D)^{-1}\mathbf{y}\mathbf{y}^\top\Big)M(D)^{-1} X\Big]_{ij}.
\end{aligned}$$

This last equation contains the elements of the Jacobian in the statement of the proposition.

3. In order to compute the Lipschitz constant of the function $\mathcal{L}_Z$ we first recall, for any $D \in \mathbb{S}^d_+$ and $Z \in \mathcal{Z}^n$, the expression $\mathcal{L}_Z(D) = n\big\|\big(XDX^\top + nI\big)^{-1}\mathbf{y}\big\|^2$ in Eq. (12). Consequently, for any $D_1, D_2 \in \mathbb{S}^d_+$ we have that

$$\begin{aligned}
\big|\mathcal{L}_Z(D_1) - \mathcal{L}_Z(D_2)\big| &= n\Big|\big\|\big(XD_1X^\top + nI\big)^{-1}\mathbf{y}\big\|^2 - \big\|\big(XD_2X^\top + nI\big)^{-1}\mathbf{y}\big\|^2\Big| \\
&\le \frac{2n}{n^3}\big\|XD_1X^\top - XD_2X^\top\big\|_\infty \|\mathbf{y}\|^2 \\
&= \frac{2}{n^2}\big\|X\big(D_1 - D_2\big)X^\top\big\|_\infty \|\mathbf{y}\|^2 \\
&\le \frac{2}{n^2}\|X\|^2_\infty \|\mathbf{y}\|^2 \big\|D_1 - D_2\big\|_\infty \\
&\le \frac{2}{n^2}\|X\|^2_\infty \|\mathbf{y}\|^2 \big\|D_1 - D_2\big\|_2,
\end{aligned}$$

where in the first inequality we have applied Lemma 7-(d) with $G_i = XD_iX^\top$, for $i = 1, 2$. The statement now follows observing that if $\mathcal{Y} \subseteq [0, 1]$, then $\|\mathbf{y}\|^2 \le n$ and if $\mathcal{X} \subseteq \mathcal{B}_1$, then $\|X\|^2_\infty \le n$.

4. We now compute the Lipschitz constant of the gradient $\nabla \mathcal{L}_Z$. In the following we will use the more compact notation $M_1 = M(D_1)$ and $M_2 = M(D_2)$, for any $D_1, D_2 \in \mathbb{S}^d_+$, and $R = \mathbf{y}\mathbf{y}^\top$. Exploiting the following facts:

   (a) $\|AB\|_2 \le \|A\|_\infty \|B\|_2$ for any two matrices A and B,
   (b) by Lemma 7-(b): $\|M_i^{-1}\|_\infty \le 1/n$ for $i = 1, 2$,

(c) by Lemma 7-(c):

$$\begin{aligned}\left\|M_1^{-1}-M_2^{-1}\right\|_\infty &= \left\|\left(XD_1X^\top+nI\right)^{-1}-\left(XD_2X^\top+nI\right)^{-1}\right\|_\infty \\ &\le \frac{1}{n^2}\left\|XD_1X^\top-XD_2X^\top\right\|_\infty \\ &\le \frac{1}{n^2}\|X\|_\infty^2\left\|D_1-D_2\right\|_\infty,\end{aligned}$$

(d) $\left\|M_1^{-2}-M_2^{-2}\right\|_\infty = \left\|M_1^{-1}\left(M_1^{-1}-M_2^{-1}\right)+\left(M_1^{-1}-M_2^{-1}\right)M_2^{-1}\right\|_\infty \le \frac{2}{n}\left\|M_1^{-1}-M_2^{-1}\right\|_\infty$,

(e) if $\mathcal{X}\subseteq\mathcal{B}_1$ and $\mathcal{Y}\subseteq[0,1]$, then $\|X\|_2\le\sqrt{n}$ and $\|R\|_\infty=\|\mathbf{y}\mathbf{y}^\top\|_\infty\le n$,

we can write the following relations:

$$\begin{aligned}&\left\|\nabla\mathcal{L}_Z(D_1)-\nabla\mathcal{L}_Z(D_2)\right\|_2 = \\ &n\left\|X^\top\left(M_1^{-1}\left(\mathbf{y}\mathbf{y}^\top M_1^{-1}+M_1^{-1}\mathbf{y}\mathbf{y}^\top\right)M_1^{-1}-M_2^{-1}\left(\mathbf{y}\mathbf{y}^\top M_2^{-1}+M_2^{-1}\mathbf{y}\mathbf{y}^\top\right)M_2^{-1}\right)X\right\|_2 \le \\ &n\|X\|_\infty\|X\|_2\left\|M_1^{-1}\left(\mathbf{y}\mathbf{y}^\top M_1^{-1}+M_1^{-1}\mathbf{y}\mathbf{y}^\top\right)M_1^{-1}-M_2^{-1}\left(\mathbf{y}\mathbf{y}^\top M_2^{-1}+M_2^{-1}\mathbf{y}\mathbf{y}^\top\right)M_2^{-1}\right\|_\infty = \\ &n\|X\|_\infty\|X\|_2\left\|M_1^{-1}RM_1^{-2}+M_1^{-2}RM_1^{-1}-M_2^{-1}RM_2^{-2}-M_2^{-2}RM_2^{-1}\right\|_\infty = \\ &n\|X\|_\infty\|X\|_2\Big\|M_1^{-1}RM_1^{-2}+M_1^{-2}RM_1^{-1}-M_2^{-1}RM_2^{-2}-M_2^{-2}RM_2^{-1} \\ &\qquad\qquad \pm M_2^{-1}RM_1^{-2}\pm M_1^{-2}RM_2^{-1}\Big\|_\infty = \\ &n\|X\|_\infty\|X\|_2\Big\|\left(M_1^{-1}-M_2^{-1}\right)RM_1^{-2}+M_1^{-2}R\left(M_1^{-1}-M_2^{-1}\right)+M_2^{-1}R\left(M_1^{-2}-M_2^{-2}\right)+ \\ &\qquad\qquad \left(M_1^{-2}-M_2^{-2}\right)RM_2^{-1}\Big\|_\infty \le \\ &2\|X\|_\infty\|X\|_2\|R\|_\infty\left(\frac{1}{n}\left\|M_1^{-1}-M_2^{-1}\right\|_\infty+\left\|M_1^{-2}-M_2^{-2}\right\|_\infty\right) \le \\ &2\|X\|_\infty\|X\|_2\left(\left\|M_1^{-1}-M_2^{-1}\right\|_\infty+2\left\|M_1^{-1}-M_2^{-1}\right\|_\infty\right) = \\ &6\|X\|_\infty\|X\|_2\left\|M_1^{-1}-M_2^{-1}\right\|_\infty \le \frac{6}{n^2}\|X\|_2\|X\|_\infty^3\left\|D_1-D_2\right\|_\infty \\ &\le 6\left\|D_1-D_2\right\|_2.\end{aligned}$$

5. The last point is contained in (Maurer, 2009, Prop. 1-(i)); we report here the proof for completeness. To this end, we require some additional notation, which will be also used in the next section of the appendix.

Remark 1 (Notation)**.** *According to our actual notation, $X\in\mathbb{R}^{n\times d}$ is the matrix having as rows the points x_i for $i=1,\dots n$. In the sequel, since the analysis will be extended also to the infinite dimension case, we will need to introduce the notation $\mathbf{x}=(x_i)_{i=1}^n\in\mathcal{X}^n$, to indicate the collection of these points; to remark this difference, we will denote the complete dataset $(\mathbf{x},\mathbf{y})$ by $\mathbf{z}$ and no more by Z. For any symmetric PSD matrix/ linear operator D and any dataset $\mathbf{z}$, with some abuse of notation, we let $D^{1/2}\mathbf{z}=(D^{1/2}x_i,y_i)_{i=1}^n$. Moreover, according to the notation introduced in the paper, we will denote the empirical error of a linear function $x\mapsto\langle w,x\rangle$ over the dataset $\mathbf{z}$ as*

$$\hat{\mathcal{R}}(\mathbf{z},w)=\mathcal{R}_{\mathbf{z}}(w).$$

Coming back to the proof of the proposition, as observed in (Argyriou et al., 2008; Maurer, 2009), it is possible to rewrite the algorithm defined in Eq. (11) in the equivalent form

$$A_D(\mathbf{z})=D^{1/2}A^{\mathrm{Rid}}(D^{1/2}\mathbf{z}), \tag{17}$$

where $A^{\mathrm{Rid}}(\mathbf{z})\in\mathbb{R}^d$ is the solution of Ridge Regression on the dataset $\mathbf{z}$, that is

$$A^{\mathrm{Rid}}(\mathbf{z})=\arg\min_{w\in\mathbb{R}^d}\left\{\hat{\mathcal{R}}(\mathbf{z},w)+\|w\|^2\right\}.$$

From Eq. (17), we have that $\langle A_D(\mathbf{z}), x\rangle = \langle A^{\mathrm{Rid}}(D^{1/2}\mathbf{z}), D^{1/2}x\rangle$, for any $x \in \mathcal{X}$ and any dataset $\mathbf{z}$. Consequently

$$\hat{\mathcal{R}}\big(\mathbf{z}, A_D(\mathbf{z})\big) = \hat{\mathcal{R}}\big(D^{1/2}\mathbf{z}, A^{\mathrm{Rid}}(D^{1/2}\mathbf{z})\big). \tag{18}$$

Due to the definition of A^{Rid}, assuming $\mathcal{Y} \subseteq [0,1]$, the following relations hold:

$$\begin{aligned}\hat{\mathcal{R}}\big(D^{1/2}\mathbf{z}, A^{\mathrm{Rid}}(D^{1/2}\mathbf{z})\big) &\leq \hat{\mathcal{R}}\big(D^{1/2}\mathbf{z}, A^{\mathrm{Rid}}(D^{1/2}\mathbf{z})\big) + \big\|A^{\mathrm{Rid}}(D^{1/2}\mathbf{z})\big\|^2 \\ &\leq \frac{1}{n}\sum_{i=1}^n \ell(0, y_i) = \frac{1}{n}\sum_{i=1}^n y_i^2 \leq 1.\end{aligned}$$

The claim now follows by combining the last inequality with Eq. (18).

■

B UNIFORM BOUNDS FOR LINEAR FEATURE LEARNING

In this section, we provide the uniform bounds on $\mathcal{E}(D) - \hat{\mathcal{E}}(D)$ and $\hat{\mathcal{E}}(D) - \hat{\mathcal{E}}_{\mathbf{Z}}(D)$ (and the corresponding symmetric quantities) for the family of linear feature learning algorithms. Our observations are essentially taken from (Maurer, 2009), we report them for clarity of exposition. We start from recalling some tools from empirical processes, then we state the uniform bounds for a more general class of learning algorithms and finally we specialize the bounds to linear feature learning. We ignore issues of measurability throughout.

B.1 PRELIMINARIES

Let m be a positive integer. In the following, we denote by $(\sigma_j)_{j=1}^m$ a sequence of i.i.d. Rademacher random variables, that is σ_j takes values on -1 or 1 with equal probabilities. We also denote by $(\gamma_j)_{j=1}^m$ a sequence of i.i.d. standard Gaussian random variables. For a set $S \subseteq \mathbb{R}^m$ we define the Rademacher average of S as

$$\mathfrak{R}(S) = \mathbb{E}_{\sigma_j}\Big[\sup_{v\in S}\frac{2}{m}\sum_{j=1}^m \sigma_j v_j\Big]$$

and the Gaussian average

$$\mathcal{G}(S) = \mathbb{E}_{\gamma_j}\Big[\sup_{v\in S}\frac{2}{m}\sum_{j=1}^m \gamma_j v_j\Big].$$

For more details about these quantities, we refer to (Bartlett & Mendelson, 2002). Given a class $\mathcal{F}$ of real-valued functions on a set $\mathcal{V}$, and given a point $V = (v_1, \ldots, v_m) \in \mathcal{V}^m$, we let

$$\mathcal{F}(V) = \Big\{\big(f(v_1), \ldots, f(v_m)\big) : f \in \mathcal{F}\Big\} \subset \mathbb{R}^m$$

so that $\mathfrak{R}(\mathcal{F}(V))$ and $\mathcal{G}(\mathcal{F}(V))$ are the corresponding Rademacher and Gaussian averages.

The following theorem is taken from (Maurer, 2009, Thm. 4), where the author considers only the inequality for the function Φ_1. Considering both inequalities allows us to obtain symmetric uniform bounds. The proof follows the same pattern as in (Maurer, 2009).

Theorem 8. *Let η be a probability distribution over the space $\mathcal{V}$, let $\mathcal{F}$ be a real-valued function class on $\mathcal{V}$ and let $V = (v_1, \ldots, v_m) \in \mathcal{V}^m$. Define the random functions:*

$$\Phi_1(V) = \sup_{f\in\mathcal{F}}\Big\{\mathbb{E}_{v\sim\eta}\big[f(v)\big] - \frac{1}{m}\sum_{j=1}^m f(v_j)\Big\}$$

$$\Phi_2(V) = \sup_{f\in\mathcal{F}}\Big\{\frac{1}{m}\sum_{j=1}^m f(v_j) - \mathbb{E}_{v\sim\eta}\big[f(v)\big]\Big\}.$$

Then the following statements hold.

1. $\mathbb{E}_{V\sim\eta^m}\big[\Phi_k(V)\big] \leq \mathbb{E}_{V\sim\eta^m}\big[\mathfrak{R}(\mathcal{F}(V))\big]$, *for* $k = 1, 2$.
2. *If* $\mathcal{F}$ *is* $[0,1]$*-valued, then, for any* $\delta \in (0,1]$*, we have that*

$$\Phi_k(V) \leq \mathbb{E}_{V\sim\eta^m}\big[\mathfrak{R}(\mathcal{F}(V))\big] + \sqrt{\frac{\log(1/\delta)}{2m}}$$

with probability at least $1-\delta$ *in* $V \sim \eta^m$*, for* $k = 1, 2$.

3. *In the previous two points we can replace* $\mathfrak{R}(\mathcal{F}(V))$ *with* $\sqrt{\pi/2}\mathcal{G}(\mathcal{F}(V))$.

Proof. The proof for the symmetric term Φ_2 proceeds in the same way as the one for Φ_1 in (Maurer, 2009, Thm. 4), more precisely, since the proof is based on symmetric arguments, the statement does not change if we flip the order of $\mathbb{E}_{v\sim\eta}\big[f(v)\big]$ and $\frac{1}{m}\sum_{j=1}^m f(v_j)$. The last inequality is a standard result, see e.g. (Boucheron et al., 2004). ■

B.2 UNIFORM BOUNDS FOR A MORE GENERAL FAMILY OF ALGORITHMS

The results presented in this sub-section hold for the infinite dimension case. In the sequel, we let $\mathcal{X}$ be a generic Hilbert space and we denote by $\langle\cdot,\cdot\rangle$ and $\|\cdot\|$ its scalar product and the induced norm. We let $\mathcal{S}_+(\mathcal{X})$ be the set of positive semidefinite bounded linear operators on $\mathcal{X}$ and, for any operator $D \in \mathcal{S}_+(\mathcal{X})$, we denote its p-Schatten norm by $\|D\|_p$, where $p \in [1,\infty]$. We continue to use the notation introduced in the paper and in Remark 1, in particular, $\mathbf{Z} = \{\mathbf{z}_t\}_{t=1}^T$ is the meta-sample and, for any $D \in \mathcal{S}_+(\mathcal{X})$, we denote $D^{1/2}\mathbf{z} = \big(D^{1/2}x_i, y_i\big)_{i=1}^n$. Throughout this section we will consider linear models and a learning algorithm $A(\mathbf{z})$ processing a training set $\mathbf{z} \in \mathcal{Z}^n$ of n points:

$$\begin{aligned} A : \mathcal{Z}^n &\to \mathcal{X} \\ \mathbf{z} &\mapsto A(\mathbf{z}), \end{aligned}$$

hence, according to our notation, we have that $A(\mathbf{z})(x) = \langle A(\mathbf{z}), x\rangle$ for any $x \in \mathcal{X}$. For any $D \in \mathcal{S}_+(\mathcal{X})$, define now the more general family of modified algorithms

$$A_D(\mathbf{z}) = D^{1/2}A(D^{1/2}\mathbf{z}).$$

By this definition, as we have already observed in the proof of Prop. 1-(5) in App. A, we have that

$$\big\langle A_D(\mathbf{z}), x\big\rangle = \big\langle A(D^{1/2}\mathbf{z}), D^{1/2}x\big\rangle$$

for any $x \in \mathcal{X}$ and consequently

$$\hat{\mathcal{R}}\big(\mathbf{z}, A_D(\mathbf{z})\big) = \hat{\mathcal{R}}\big(D^{1/2}\mathbf{z}, A(D^{1/2}\mathbf{z})\big).$$

In this way, we can consider the family of learning algorithms $\big\{\mathbf{z} \mapsto A_D(\mathbf{z}) : D \in \mathcal{S}_+(\mathcal{X})\big\}$, parametrised by the operators D. Recall now, for every $D \in \mathcal{S}_+(\mathcal{X})$, the notion of transfer risk

$$\mathcal{E}(D) = \mathbb{E}_{\mu\sim\rho}\mathbb{E}_{\mathbf{z}\sim\mu^n}\mathbb{E}_{(x,y)\sim\mu}\big[\ell(\langle A_D(\mathbf{z}), x\rangle, y)\big],$$

future empirical risk

$$\hat{\mathcal{E}}(D) = \mathbb{E}_{\mu\sim\rho}\mathbb{E}_{\mathbf{z}\sim\mu^n}\big[\hat{\mathcal{R}}\big(\mathbf{z}, A_D(\mathbf{z})\big)\big]$$

and multi-task empirical risk

$$\hat{\mathcal{E}}_{\mathbf{Z}}(D) = \frac{1}{T}\sum_{t=1}^T \hat{\mathcal{R}}\big(\mathbf{z}_t, A_D(\mathbf{z}_t)\big).$$

The following two theorems are taken from (Maurer, 2009), where the author does not consider the symmetric case, which immediately follows from Thm. 8. More precisely, The first theorem is taken from (Maurer, 2009, Thm. 6) and second one from (Maurer, 2009, Thm. 8). In the sequel, the symbol C_ρ, already introduced in the paper, denotes the covariance of the input data, obtained by averaging over all input marginals sampled from ρ, that is, $C_\rho = \mathbb{E}_{\mu\sim\rho}\mathbb{E}_{(x,y)\sim\mu}\,[C(x)]$, where for any $x \in \mathcal{X}$, and for any $v \in \mathcal{X}$, $C(x)v = \langle v, x\rangle x$.

Theorem 9. *Let p and q be conjugate exponents in $[1,\infty]$ and assume $\mathcal{X} \subseteq \mathcal{B}_1$. Consider a learning algorithm A such that $\|A(D^{1/2}\mathbf{z})\| \leq 1$ for any $\mathbf{z} \in \mathcal{Z}^n$ and any $D \in \mathcal{S}_+(\mathcal{X})$, and let ℓ be a loss function such that, for any $y \in \mathbb{R}$, $\ell(\cdot, y)$ has Lipschitz constant $L(K)$ on the interval $[-K, K]$, for any $K \geq 0$. Then for any meta-distribution ρ on $\mathcal{Z}$ and for any $D \in \mathcal{S}_+(\mathcal{X})$ we have that:*

$$\left|\mathcal{E}(D) - \hat{\mathcal{E}}(D)\right| \leq \sqrt{\frac{2\pi}{n}} L\big(\|D\|_\infty^{1/2}\big) \|C_\rho\|_p^{\,1/2} \|D\|_q^{\,1/2}.$$

In order to give the next theorem, we need to introduce the Gramian matrix defined by the entries $[G(\mathbf{x})]_{i,j} = \langle x_i, x_j \rangle$ for $i, j = 1, \ldots, n$.

Theorem 10. *Let $\mathcal{X} \subseteq \mathcal{B}_1$ and $\mathfrak{D} \subseteq \mathcal{S}_+(\mathcal{X})$ be a bounded set. Consider a function $f : \mathcal{Z}^n \to [0,1]$ satisfying the condition*

$$\left|f(\mathbf{z}) - f(\mathbf{z}')\right| \leq \frac{L_K}{n} \left\|G(\mathbf{x}) - G(\mathbf{x}')\right\|_2$$

for any $\mathbf{z}, \mathbf{z}' \in \mathcal{Z}^n$ and for some $L_K \geq 0$. Let $\mu_1, \ldots, \mu_T$ tasks independently sampled from ρ and $\mathbf{z}_t$ sampled from μ_t^n for $t \in \{1, \ldots, T\}$. Then, for any $\delta \in (0,1]$, we have that

$$\sup_{D \in \mathfrak{D}} \left| \mathbb{E}_{\mu \sim \rho} \mathbb{E}_{\mathbf{z} \sim \mu^n} \left[f(D^{1/2}\mathbf{z}) \right] - \frac{1}{T} \sum_{t=1}^{T} f(D^{1/2}\mathbf{z}_t) \right| \leq \Big(\sup_{D \in \mathfrak{D}} \|D\|_2 \Big) \frac{\sqrt{2\pi} L_K}{\sqrt{T}} + \sqrt{\frac{\log(1/\delta)}{2T}}$$

with probability at least $1-\delta$ with respect to the independent sampling of the tasks $\mu_t \sim \rho$ and training sets $\mathbf{z}_t \sim \mu_t^n$ for any $t \in \{1, \ldots, T\}$.

B.3 APPLICATION TO THE FAMILY OF LINEAR FEATURE LEARNING ALGORITHM

Similarly to what observed in Prop. 1-(5) in App. A, also in the infinite dimension case, we can cast the family of linear feature learning algorithms in the framework described in the previous sub-section, taking the original vanilla algorithm $A(\mathbf{z})$ as Ridge Regression with regularization parameter equal to 1:

$$A(\mathbf{z}) = A^{\mathrm{Rid}}(\mathbf{z}) = \arg\min_w \left\{ \hat{\mathcal{R}}(\mathbf{z}, w) + \|w\|^2 \right\}, \tag{19}$$

we refer to (Maurer, 2009) for more details. Thus, we can apply the results in the previous sub-section to this specific case, in order to obtain the results stated in the paper for the uniform bounds. In fact, in the paper we have analyzed the finite dimension case, but from this analysis, we deduced that they still hold in the infinite dimension setting. The following definition, see (Maurer, 2009, Def. 1), will be used in the sequel.

Definition 11. *Relative to a loss function ℓ, a learning algorithm $A : \mathcal{Z}^n \to \mathcal{X}$ is said to*

1. *be 1-bounded if $\|A(\mathbf{z})\| \leq 1$ and $\hat{\mathcal{R}}(\mathbf{z}, A(\mathbf{z})) \leq 1$ for any $\mathbf{z} \in \mathcal{Z}^n$;*
2. *have kernel stability L_K if $\left|\hat{\mathcal{R}}(\mathbf{z}, A(\mathbf{z})) - \hat{\mathcal{R}}(\mathbf{z}', A(\mathbf{z}'))\right| \leq \frac{L_K}{n} \left\|G(\mathbf{x}) - G(\mathbf{x}')\right\|_2$, for any $\mathbf{z}, \mathbf{z}' \in \mathcal{Z}^n$ and for some $L_K \geq 0$.*

The following two lemmas are essentially taken from (Maurer, 2009) and they are respectively immediate consequences of Thm. 9 and Thm. 10 applied to the family of linear feature learning algorithms with restriction to the set

$$\mathfrak{D} = \mathfrak{D}_\lambda = \left\{ D \in \mathcal{S}_+(\mathcal{X}) : \mathrm{tr}(D) \leq 1/\lambda \right\}.$$

Proposition 3 (Uniform Generalization Error Bound)**.** *Let $\mathcal{X} \subseteq \mathcal{B}_1$, $\mathcal{Y} \subseteq [0,1]$ and let ℓ be the square loss, then*

$$\sup_{D \in \mathfrak{D}_\lambda} |\mathcal{E}(D) - \hat{\mathcal{E}}(D)| \leq \frac{2\sqrt{2\pi} \|C_\rho\|_\infty^{1/2}}{\sqrt{n}} \, \frac{1 + \sqrt{\lambda}}{\lambda}.$$

Proof. Thanks to the assumption $\mathcal{Y} \subseteq [0,1]$, by (Maurer, 2009, Prop. 1), $A^{\mathrm{Rid}}(\mathbf{z})$, is 1-bounded – and in particular, $\|A^{\mathrm{Rid}}(D^{1/2}\mathbf{z})\| \leq 1$ for any $D \in \mathcal{L}_+(\mathcal{X})$ and any dataset $\mathbf{z}$ – with respect to the square loss. Hence, we can apply Thm. 9 to A^{Rid}. We restrict to the set $\mathfrak{D}_\lambda$, we choose $q = 1$ and $p = \infty$ and we observe that the square loss is $M(K) = 2(K+1)$-Lipschitz on the interval $[-K, K]$. ■

Proposition 12. *Let $\mathcal{X} \subseteq \mathcal{B}_1$, $\mathcal{Y} \subseteq [0,1]$ and let ℓ be the square loss. Let $\mu_1, \dots, \mu_T$ be independently sampled from ρ and Z_t sampled from μ_t^n for $t \in \{1, \dots, T\}$. Then, for any $\delta \in (0,1]$,*

$$\sup_{D \in \mathfrak{D}_\lambda} \left| \hat{\mathcal{E}}(D) - \hat{\mathcal{E}}_{\mathbf{Z}}(D) \right| \leq \frac{2\sqrt{2\pi}}{\lambda\sqrt{T}} + \sqrt{\frac{\log(1/\delta)}{2T}}$$

with probability at least $1-\delta$ with respect to the independent sampling of the tasks $\mu_t \sim \rho$ and training sets $Z_t \sim \mu_t^n$ for any $t \in \{1, \dots, T\}$.

Proof. Thanks to the assumption that $\mathcal{Y} \subseteq [0,1]$, by (Maurer, 2009, Prop. 1), $A^{\mathrm{Rid}}(\mathbf{z})$ is 1-bounded – and in particular, $\hat{\mathcal{R}}\big(D^{1/2}\mathbf{z}, A^{\mathrm{Rid}}(D^{1/2}\mathbf{z})\big) \leq 1$ for any $D \in \mathcal{L}_+(\mathcal{X})$ and any dataset $\mathbf{z}$ – and has kernel stability $L_K = 2$ with respect to the square loss. We can then apply Thm. 10 to the function

$$f(\mathbf{z}) = \hat{\mathcal{R}}\big(\mathbf{z}, A_D(\mathbf{z})\big) = \hat{\mathcal{R}}\big(D^{1/2}\mathbf{z}, A^{\mathrm{Rid}}(D^{1/2}\mathbf{z})\big).$$

■

C PROOF OF Thm. 6

In this section, we report the proof of Thm. 6. We do not make any claim of originality in this theorem which is merely a collection of results contained in (Maurer, 2009); we report the proof for completeness.

Theorem 6 (Batch LTL Bound)**.** *Let $\mathcal{X} \subseteq \mathcal{B}_1$, $\mathcal{Y} \subseteq [0,1]$ and let ℓ be the square loss. Let tasks $\mu_1, \dots, \mu_T$ be independently sampled from ρ and Z_t sampled from μ_t^n for $t \in \{1, \dots, T\}$. Let $\hat{D}_T$ be a minimizer of the multi-task empirical risk in Eq. (14) over the set $\mathfrak{D}_\lambda$. Then, for any $\delta \in (0,1]$*

$$\mathcal{E}(\hat{D}_T) - \mathcal{E}(D_*) \leq \frac{4\sqrt{2\pi}\|C_\rho\|_\infty^{1/2}}{\sqrt{n}} \frac{1+\sqrt{\lambda}}{\lambda} + \frac{2\sqrt{2\pi}}{\lambda\sqrt{T}} + \sqrt{\frac{2\log(2/\delta)}{T}}$$

with probability at least $1-\delta$ with respect to the independent sampling of the tasks $\mu_t \sim \rho$ and training sets $Z_t \sim \mu_t^n$ for any $t \in \{1, \dots, T\}$.

Proof. Similarly to the online case, the proof of Thm. 6 relies on the following decomposition.

$$\mathcal{E}(\hat{D}_T) - \mathcal{E}(D_*) = \underbrace{\mathcal{E}(\hat{D}_T) - \hat{\mathcal{E}}_{\mathbf{Z}}(\hat{D}_T)}_{A} + \underbrace{\hat{\mathcal{E}}_{\mathbf{Z}}(\hat{D}_T) - \hat{\mathcal{E}}_{\mathbf{Z}}(D_*)}_{B} + \underbrace{\hat{\mathcal{E}}_{\mathbf{Z}}(D_*) - \mathcal{E}(D_*)}_{C}.$$

We now describe how to deal with each term. We decompose the term A as

$$\mathcal{E}(\hat{D}_T) - \hat{\mathcal{E}}_{\mathbf{Z}}(\hat{D}_T) = \underbrace{\mathcal{E}(\hat{D}_T) - \hat{\mathcal{E}}(\hat{D}_T)}_{A1} + \underbrace{\hat{\mathcal{E}}(\hat{D}_T) - \hat{\mathcal{E}}_{\mathbf{Z}}(\hat{D}_T)}_{A2}$$

and we bound the term $A1$ by Prop. 3 and the term $A2$ by Prop. 12 with confidence parameter $\delta/2$. The term B, thanks to the definition of $\hat{D}_T$, is negative. Lastly, as regards the term C, we split it in

$$\hat{\mathcal{E}}_{\mathbf{Z}}(D_*) - \mathcal{E}(D_*) = \underbrace{\hat{\mathcal{E}}_{\mathbf{Z}}(D_*) - \hat{\mathcal{E}}(D_*)}_{C1} + \underbrace{\hat{\mathcal{E}}(D_*) - \mathcal{E}(D_*)}_{C2},$$

where we bound $C2$ by Prop. 3, while, in order to bound the first term $C1$, we apply Hoeffding's inequality (see Lemma 13 below) with parameters $a_t = 0$ and $b_t = 1$ for any t (thanks to Prop. 1-(5)) and confidence parameter $\delta/2$, i.e. for any $\delta \in (0, 1]$, we have that

$$\hat{\mathcal{E}}_{\mathbf{Z}}(D_*) - \hat{\mathcal{E}}(D_*) \leq \sqrt{\frac{\log(2/\delta)}{2T}}$$

with probability at least $1 - \delta/2$ in $\mathbf{Z}$. Joining all the previous parts, the statement follows. ■

Lemma 13 (Hoeffding's inequality (Boucheron et al., 2004))**.** *Let m be a positive integer and let $X_1, \ldots, X_m$ be independent random variables such that $X_i \in [a_i, b_i]$ with probability 1, for $i = 1, \ldots, m$. Define $\bar{X}_m = \frac{1}{m}\sum_{i=1}^{m} X_i$. Then, for any $\epsilon > 0$, we have that*

$$\mathbb{P}\big[\bar{X}_m - \mathbb{E}\big[\bar{X}_m\big] \geq \epsilon\big] \leq \exp\Big(-\frac{2m^2\epsilon^2}{\sum_{i=1}^{m}(b_i - a_i)^2}\Big),$$

or equivalently, for any $\delta \in (0, 1]$, we have that

$$\bar{X}_m - \mathbb{E}\big[\bar{X}_m\big] \leq \sqrt{\frac{1}{2m^2}\Big(\sum_{i=1}^{m}(b_i - a_i)^2\Big)\log\Big(\frac{1}{\delta}\Big)}$$

with probability at least $1 - \delta$. Moreover, thanks to symmetric arguments, the previous inequalities hold also for $\mathbb{E}\big[\bar{X}_m\big] - \bar{X}_m$.

D NON-ASYMPTOTIC RATES FOR PROJECTED STOCHASTIC SUBGRADIENT ALGORITHM

In this section, we briefly describe how to derive non-asymptotic convergence rates in probability for Projected Stochastic Subgradient Algorithm (PSSA), exploiting the regret bounds for Projected Online Subgradient Algorithm (POSA). In the first part we give a regret bound for POSA and we specialize it to Alg. 1 for the case of the square loss (Lemma 4). In the second part we first show, in general, how a bound on the regret implies a rate in probability for the convergence in the statistical setting and then we specialize this result to obtain the bound on the excess empirical future risk of the output of Alg. 1 for the case of the square loss (Prop. 5). The results contained in this section are standard, we will cite during the presentation some references where the interested reader can find more details. Throughout this section, no differentiability assumptions on the functions will be made, we only require them to be convex and Lipschitz. We also require the boundedness of the diameter of the set over which we optimize. The general analysis will be conducted in a Hilbert space with scalar product $\langle \cdot, \cdot \rangle$ and induced norm $\|\cdot\|$.

D.1 PROJECTED ONLINE SUBGRADIENT ALGORITHM, POSA

The Online Convex Optimization (OCO) framework (Hazan, 2016) over a convex and closed set H of a Hilbert space can be seen as a repeated game: at iteration t, the online player, i.e. the online algorithm, chooses $h^{(t)} \in H$, after this, a cost function $f_t : H \to \mathbb{R}$ is revealed by the adversary and the cost incurred by the online player is $f_t(h^{(t)})$. The cost functions f_t are usually assumed to be bounded convex functions over H, belonging to some bounded family of functions and they could be even adversely chosen. The performance of an online algorithm over a total number of game iterations T is measured by its $regret$, defined as the difference between the total averaged cost the algorithm incurred over T matches and that of the best fixed decision in hindsight:

$$R_T = \frac{1}{T}\sum_{t=1}^{T} f_t(h^{(t)}) - \min_{h \in H} \frac{1}{T}\sum_{t=1}^{T} f_t(h).$$

In the sequel, we will always assume the convexity of the functions f_t and the existence of a minimizer of the batch problem $\hat{h} \in \arg\min_{h \in H} \sum_{t=1}^{T} f_t(h)$. In our case, we will focus on the classical Projected Online Subgradient Algorithm described in Alg. 2 and we will give an upper bound on its regret. When needed, the following assumptions will be made.

Algorithm 2 POSA

Input: $T \in \mathbb{N}$ number of iterations, $\{\gamma_t\}_t$ step sizes
Initialization: $h^{(1)} \in H$
For $t = 1$ to T
Receive f_t, pay $f_t(h^{(t)})$
Choose $u_t \in \partial f_t(h^{(t)})$
Update $h^{(t+1)} = \mathrm{proj}_H(h^{(t)} - \gamma_t u_t)$
Return $h^{(T)}$

Assumption 1. *Assume that for any t the functions f_t are G-Lipschitz on H, i.e. there exists a positive constant such that $\|u\| \leq G$ for any $u \in \partial f_t(h)$ and for any $h \in H$.*

Assumption 2. *Assume that the diameter of the set H is bounded by some constant $\mathcal{D} > 0$, i.e. $\sup_{h,h' \in H} \|h - h'\| \leq \mathcal{D}$.*

The following theorem is a classical result and a slightly different version can be found in (Hazan, 2016, Thm. 3.1), we report here the proof because of clarity and completeness.

Theorem 14 (Regret Bound for Alg. 2)**.** *Under Asm. 1 and Asm. 2, the regret of Alg. 2, with $\gamma_t = c/\sqrt{t}$ for some $c > 0$, is bounded by*

$$R_T \leq \frac{1}{2}\Big(\frac{\mathcal{D}^2}{c} + 2cG^2\Big)\frac{1}{\sqrt{T}}.$$

Moreover, the optimal value for the previous bound, attained at $c = \dfrac{\mathcal{D}}{\sqrt{2}G}$, is $R_T \leq \dfrac{\sqrt{2}\mathcal{D}G}{\sqrt{T}}$.

Proof. Since $u_t \in \partial f_t(h^{(t)})$, by convexity of f_t and definition of subgradient, we have that:

$$f_t(h^{(t)}) - f_t(\hat{h}) \leq \langle u_t, h^{(t)} - \hat{h} \rangle. \tag{20}$$

Using the update rule of Alg. 2, Pythagorean Theorem (i.e. the non-expansiveness property of the projection operator) and Asm. 1, the following relations hold:

$$\begin{aligned}
\|h^{(t+1)} - \hat{h}\|^2 &= \|\mathrm{proj}_H(h^{(t)} - \gamma_t u_t) - \hat{h}\|^2 \\
&\leq \|h^{(t)} - \gamma_t u_t - \hat{h}\|^2 \\
&= \|h^{(t)} - \hat{h}\|^2 - 2\gamma_t \langle u_t, h^{(t)} - \hat{h} \rangle + \gamma_t^2 \|u_t\|^2 \\
&\leq \|h^{(t)} - \hat{h}\|^2 - 2\gamma_t \langle u_t, h^{(t)} - \hat{h} \rangle + \gamma_t^2 G^2,
\end{aligned}$$

which imply that

$$\langle u_t, h^{(t)} - \hat{h} \rangle \leq \frac{\|h^{(t)} - \hat{h}\|^2 - \|h^{(t+1)} - \hat{h}\|^2}{2\gamma_t} + \frac{\gamma_t G^2}{2}. \tag{21}$$

Combining Eq. (21) with Eq. (20), we obtain:

$$f_t(h^{(t)}) - f_t(\hat{h}) \leq \frac{\|h^{(t)} - \hat{h}\|^2}{2\gamma_t} - \frac{\|h^{(t+1)} - \hat{h}\|^2}{2\gamma_t} + \frac{\gamma_t G^2}{2}. \tag{22}$$

Now, summing Eq. (22) from $t=1$ to $t=T$, using the convention $1/\gamma_0 = 0$, and setting $\gamma_t = c/\sqrt{t}$ we can write:

$$\begin{aligned}
\sum_{t=1}^{T}\Big(f_t(h^{(t)}) - f_t(\hat{h})\Big) &\leq \frac{1}{2}\sum_{t=1}^{T}\frac{\|h^{(t)}-\hat{h}\|^2}{\gamma_t} - \frac{1}{2}\sum_{t=1}^{T}\frac{\|h^{(t+1)}-\hat{h}\|^2}{\gamma_t} + \frac{G^2}{2}\sum_{t=1}^{T}\gamma_t \\
&= \frac{1}{2}\sum_{t=1}^{T}\frac{\|h^{(t)}-\hat{h}\|^2}{\gamma_t} - \frac{1}{2}\sum_{t=1}^{T}\frac{\|h^{(t)}-\hat{h}\|^2}{\gamma_{t-1}} - \frac{1}{2}\frac{\|h^{(T+1)}-\hat{h}\|^2}{\gamma_T} + \frac{G^2}{2}\sum_{t=1}^{T}\gamma_t \\
&\leq \frac{1}{2}\sum_{t=1}^{T}\Big(\frac{1}{\gamma_t} - \frac{1}{\gamma_{t-1}}\Big)\|h^{(t)}-\hat{h}\|^2 + \frac{G^2}{2}\sum_{t=1}^{T}\gamma_t \\
&\leq \frac{1}{2}\Big(\frac{\mathcal{D}^2}{\gamma_T} + G^2\sum_{t=1}^{T}\gamma_t\Big) \leq \frac{1}{2}\Big(\frac{\mathcal{D}^2}{c} + 2cG^2\Big)\sqrt{T},
\end{aligned}$$

where we have exploited Asm. 2, more precisely $\|h^{(t)} - \hat{h}\| \leq \mathcal{D}$, the fact that $\sum_{t=1}^{T}\Big(\frac{1}{\gamma_t} - \frac{1}{\gamma_{t-1}}\Big) = \frac{1}{\gamma_T}$ and the inequality $\sum_{t=1}^{T}\frac{1}{\sqrt{t}} \leq 2\sqrt{T} - 1 \leq 2\sqrt{T}$. Dividing by T and optimizing with respect to c, the result follows. ■

We now specialize the regret bound obtained for the generic Alg. 2 to our Alg. 1 described in the paper for the square loss.

Lemma 4 (Regret Bound for Alg. 1)**.** *Let $\mathcal{X} \subseteq \mathcal{B}_1$, $\mathcal{Y} \subseteq [0,1]$ and ℓ be the square loss. Then the regret of Alg. 1 with step-sizes $\gamma_t = (\lambda\sqrt{2t})^{-1}$ is such that*

$$R_T \leq \frac{4\sqrt{2}}{\lambda\sqrt{T}}.$$

Proof. The thesis follows from applying Thm. 14 to the context of Alg. 1 with the square loss. In this case the iteration $h^{(t)}$ coincide with $D^{(t)}$, the cost functions are identified with $f_t = \mathcal{L}_{Z_t}$, hence they are 2-Lipschitz thanks to Prop. 1-(3) and, consequently, we can take $G = 2$ in Thm. 14. Moreover, the diameter $\mathcal{D}$ of the set over which we project $\mathfrak{D}_\lambda$ (in the previous notation H) is $2/\lambda$. Indeed, for any $D \in \mathfrak{D}_\lambda$ we have that $\|D\|_2 \leq \|D\|_1 = \mathrm{tr}(D) \leq 1/\lambda$, hence $\mathcal{D} = \sup_{D,D' \in \mathfrak{D}_\lambda}\|D - D'\|_2 \leq 2/\lambda$. ■

D.2 ONLINE-TO-BATCH CONVERSION

Consider a collection of data points $\{Z_t\}_t$ belonging to some space and let η be a probability distribution over it. In the sequel of the discussion we will ignore all measurability issues. Let H be a set as above and for every $h \in H$ define $F(h) = \mathbb{E}_{Z\sim\eta}\big[\mathcal{L}_Z(h)\big]$, where, for any Z, $\mathcal{L}_Z$ is a convex function. In the following we will consider the optimization problem

$$\min_{h\in H} F(h) \tag{23}$$

and we will assume the existence of a minimizer $h_* \in \arg\min_{h\in H} F(h)$. In order to solve the stochastic problem in Eq. (23), we will analyze the general incremental procedure described in Alg. 3, where the next point is updated by some rule depending on the past history of the process, for instance, if we choose the update $h^{(t+1)} = \mathrm{proj}_H(h^{(t)} - \gamma_t u_t)$, for some $\gamma_t > 0$ and $u_t \in \partial f_t(h^{(t)})$, then Alg. 3 coincides with POSA (Alg. 2) applied to the functions $f_t = \mathcal{L}_{Z_t}$.

In the online setting no further assumptions about the data are made, however, in the statistical setting we typically assume that the data are i.i.d. from the distribution η; since this last setting is more restrictive, one would expect that if Alg. 3 solves the problem in the online framework, i.e. if its regret R_T is such that $R_T \to 0$ as $T \to \infty$, then it will also solve the corresponding problem (23) in the statistical setting. This statement is formally confirmed by the following theorem (Hazan, 2016, Thm. 9.3), which relies on results taken from (Cesa-Bianchi et al., 2004).

Theorem 15 (Online-to-batch)**.** *Let $f_t = \mathcal{L}_{Z_t}$ be convex functions with values in $[0,1]$ for any Z_t, $t \in \{1,\dots,T\}$ and let the points $\{Z_t\}_{t=1}^T$ processed by Alg. 3 be i.i.d. sampled from η. Then, denoting by R_T the regret bound of Alg. 3, for any $\delta \in (0,1]$*

$$F(\bar{h}_T) - F(h_*) \leq R_T + \sqrt{\frac{8\log(2/\delta)}{T}}$$

Algorithm 3 Generic Incremental Procedure in the Online and Statistical Settings

ONLINE SETTING

Input: $T \in \mathbb{N}$ number of iterations, $\{\gamma_t\}_t$ step sizes
Initialization: $h^{(1)} \in H$
For $t = 1$ to T:
Receive $Z_t \longrightarrow$ **no further assumptions**
Define $f_t = \mathcal{L}_{Z_t}$, pay $f_t(h^{(t)})$
Update $h^{(t+1)}$
Return $h^{(T)}$

STATISTICAL SETTING

Input: $T \in \mathbb{N}$ number of iterations, $\{\gamma_t\}_t$ step sizes
Initialization: $h^{(1)} \in H$
For $t = 1$ to T:
Receive $Z_t \longrightarrow$ **sampled i.i.d. from η**
Define $f_t = \mathcal{L}_{Z_t}$, pay $f_t(h^{(t)})$
Update $h^{(t+1)}$
Return $\bar{h}_T = \frac{1}{T}\sum_{t=1}^{T} h^{(t)}$

with probability at least $1-\delta$ *with respect to the independent sampling of the data* Z_t *for any* $t \in \{1, \dots, T\}$.

The previous theorem relies on the theory of Martingales (Grimmett & Stirzaker, 2001) and the analysis of the first term $\frac{1}{T}\sum_{t=1}^{T} f_t(h^{(t)})$ of the regret, see e.g. (Cesa-Bianchi et al., 2004), in fact this term is a data-dependent statistics evaluating the average cumulative error of the prediction $h^{(t)}$ of the algorithm on the next point Z_t, therefore it is reasonable to expect that it contains information about the generalization ability of the algorithm.

Adapting the previous discussion to the setting of Alg. 1 for the square loss, we obtain the following rate for the excess empirical future risk of online estimator returned by the algorithm.

Proposition 5 (Excess Future Empirical Risk Bound for Alg. 1)**.** *Let* $\mathcal{X} \subseteq \mathcal{B}_1$, $\mathcal{Y} \subseteq [0,1]$ *and let* ℓ *be the square loss. Let* $\mu_1, \dots, \mu_T$ *be independently sampled from* ρ *and* Z_t *sampled from* μ_t^n *for* $t \in \{1, \dots, T\}$. *Let* $\bar{D}_T$ *be the output of Alg. 1 with step sizes* $\gamma_t = (\lambda\sqrt{2t})^{-1}$. *Then, for any* $\delta \in (0, 1]$

$$\hat{\mathcal{E}}(\bar{D}_T) - \hat{\mathcal{E}}(\hat{D}_*) \leq \frac{4\sqrt{2}}{\lambda\sqrt{T}} + \sqrt{\frac{8\log(2/\delta)}{T}}$$

with probability at least $1-\delta$ *with respect to the independent sampling of the tasks* $\mu_t \sim \rho$ *and training sets* $Z_t \sim \mu_t^n$ *for any* $t \in \{1, \dots, T\}$.

Proof. The statement directly follows by combining Thm. 15 with the regret bound in Lemma 4 to the context of Alg. 1 for the square loss: we identify the set H with the set $\mathfrak{D}_\lambda$, the output $\bar{h}_T$ with the online estimator $\bar{D}_T$, the expectation $\mathbb{E}_\eta$ with $\mathbb{E}_{\mu\sim\rho}\mathbb{E}_{Z\sim\mu^n}$ and the function F with the future empirical risk $\hat{\mathcal{E}}$, the remaining identifications are obvious. We remark that, thanks to Prop. 1-(5), the boundedness condition on the functions $\mathcal{L}_{Z_t}$ needed in order to apply Thm. 15, is satisfied in our setting. ■

E PROJECTION ON THE SET $\mathfrak{D}_\lambda$

In the following lemma we describe how to perform the projection over the set $\mathfrak{D}_\lambda$ in a finite number of steps. Without loss of generality we consider the case that $\lambda = 1$; the case regarding a general value of λ immediately follows by a rescaling argument.

Lemma 16. *Let* Q *be a* $d \times d$ *symmetric matrix and let* $U\Delta U^\top$ *be the eigen-decomposition of* Q, *where* $\Delta = \mathrm{Diag}(\delta_1, \dots, \delta_d)$, *and* $\delta_d \geq \delta_{d-1} \geq \cdots \geq \delta_1$. *Then the solution of the problem*

$$\hat{D} = \mathrm{proj}_{\mathfrak{D}_\lambda}(Q) = \mathrm{argmin}\Big\{\|D - Q\|^2 : D \succeq 0, \mathrm{tr}(D) \leq 1\Big\}$$

is given by $\hat{D} = Q$ *if* Q *satisfies the constraints and* $\hat{D} = U\Theta U^\top$ *otherwise, where* $\Theta = \mathrm{Diag}(\hat{\theta}_1, \dots, \hat{\theta}_d)$, *with, for every* $i = 1, \dots, d$,

$$\hat{\theta}_i = \begin{cases} 0 \quad \textit{if} \quad \delta_i \leq 0 \\ \delta_i \quad \textit{if} \quad \delta_i > 0 \quad \textit{and} \quad \sum_{j:\delta_j>0} \delta_j \leq 1 \\ \max(0, \delta_i - a) \quad \textit{if} \quad \delta_i > 0 \quad \textit{and} \quad \sum_{j:\delta_j>0} \delta_j > 1 \end{cases} \tag{24}$$

and a is the non-negative solution of the equation $\sum_{j:\delta_j>0}\max\{0,\delta_j-a\}=1$.

Remark 2. *We observe that, the function in the equation $\sum_{j:\delta_j>0}\max\{0,\delta_j-a\}=1$ is piece-wise linear in a and its critical points (i.e. the points at which the slope of the function changes) are the points $\{a_j=\delta_j\}_{j:\delta_j>0}$. Consequently, in order to compute the non-negative solution of this equation, we adopt the procedure described in (McDonald et al., 2016, Thms. 11 and 13). This approach provides us the solution in at most $O(d\log(d))$ time, hence, the computational cost of the projection is dominated by the computational cost $O(d^3)$ of performing the eigen-decomposition of Q.*

The proof of Lemma 16 follows a standard path of reducing the matrix problem to a vector problem, after which an argument based on the Karush–Kuhn–Tucker (KKT) conditions is employed.

Proof. We analyze the case in which the matrix Q does not satisfy the constraints. Thanks to (Bauschke et al., Cor. 24.65), considering the eigen-decomposition of the matrix to be projected $Q=U\Delta U^\top$, where $\Delta=\mathrm{Diag}(\delta)$ with $\delta=(\delta_1,\dots,\delta_d)\in\mathbb{R}^d$ and $\delta_d\geq\delta_{d-1}\geq\cdots\geq\delta_1$, we have that

$$\mathrm{proj}_{\mathfrak{D}_\lambda}(Q)=U\mathrm{Diag}\big(\mathrm{proj}_{\mathfrak{C}_\lambda}(\delta)\big)U^\top,$$

where, using the notation $\mathbb{R}^d_+=\big\{\theta\in\mathbb{R}^d:\theta_i\geq 0, i=1,\dots,d\big\}$, we have introduced the vector-set $\mathfrak{C}_\lambda=\big\{\theta\in\mathbb{R}^d_+:\sum_{i=1}^d\theta_i\leq 1\big\}$. Consequently, it is sufficient to compute $\hat\theta=\mathrm{proj}_{\mathfrak{C}_\lambda}(\delta)$, i.e. we have to solve the constrained vector-problem:

$$\hat\theta=\arg\min\Big\{||\theta-\delta||^2:\theta\in\mathfrak{C}_\lambda\Big\}=\arg\min\Big\{||\theta-\delta||^2:\theta\in\mathbb{R}^d_+,\sum_{i=1}^d\theta_i\leq 1\Big\}. \tag{25}$$

Now, since the problem in Eq. (25) is convex, the KKT conditions are not only necessary, but also sufficient. More precisely, we know that there exist $a\in\mathbb{R}$ and $b=(b_1,\dots,b_d)\in\mathbb{R}^d$ such that

$$\begin{cases}\hat\theta\geq 0\\ \sum_{i=1}^d\hat\theta_i\leq 1\\ a\geq 0\\ b\geq 0\\ a\Big(\sum_{i=1}^d\hat\theta_i-1\Big)=0\quad(*)\\ b\odot\hat\theta=0\\ \hat\theta=\delta-a+b\end{cases}$$

where the symbol $\odot$ denotes the Hadamard product between two vectors in $\mathbb{R}^d$, i.e. the component-wise product. Splitting the two possible cases in $(*)$, we can rewrite KKT conditions as the union of the following two systems:

$$A:\begin{cases}\hat\theta\geq 0\\ \sum_{i=1}^d\hat\theta_i\leq 1\\ b\geq 0\\ a=0\\ b\odot\hat\theta=0\\ \hat\theta=\delta-a+b\end{cases}\quad\text{or}\quad B:\begin{cases}\hat\theta\geq 0\\ a\geq 0\\ b\geq 0\\ \sum_{i=1}^d\hat\theta_i=1\\ b\odot\hat\theta=0\\ \hat\theta=\delta-a+b.\end{cases}$$

Combining all the constraints, one finds that the system A admits solutions iff $\sum_{j:\delta_j>0}\delta_j\leq 1$ and in a such case the solutions of the system are given by:

$$\begin{cases}a=0\\ b_j=\max\{0,-\delta_j\}\quad j=1,\dots,d\\ \hat\theta_j=\max\{0,\delta_j\}\quad j=1,\dots,d.\end{cases}$$

In a similar way, one finds that the system B admits solutions iff $\sum_{j:\delta_j>0} \delta_j > 1$ and in a such case the solutions of the system are given by:

$$\begin{cases} a \geq 0 : \sum_{j:\delta_j>0} \max\{0, \delta_j - a\} = 1 \\ b_j = \max\{0, \delta_j - a\} - \big(\delta_j - a\big) \quad j = 1, \ldots, d \\ \hat{\theta}_j = \max\{0, \delta_j - a\} \quad j = 1, \ldots, d. \end{cases}$$

Finally, combining the two previous cases, we have that

$$\begin{cases} \begin{cases} a = 0 \quad \text{if} \quad \sum_{j:\delta_j>0} \delta_j \leq 1 \\ a \geq 0 : \sum_{j:\delta_j>0} \max\{0, \delta_j - a\} = 1 \quad \text{if} \quad \sum_{j:\delta_j>0} \delta_j > 1 \end{cases} \\ b_j = \max\{0, \delta_j - a\} - \big(\delta_j - a\big) \quad j = 1, \ldots, d \\ \hat{\theta}_j = \max\{0, \delta_j - a\} \quad j = 1, \ldots, d. \end{cases}$$

The expression in Eq. (24) derives by combining all the cases. ■