# Intrinsic Geometries in Learning

Richard Nock[1] and Frank Nielsen[2,3]

[1] Ceregmia — Université Antilles-Guyane, Schoelcher, France
rnock@martinique.univ-ag.fr
[2] LIX — Ecole Polytechnique, Palaiseau, France
nielsen@lix.polytechnique.fr
[3] Sony Computer Science Laboratories Inc., Tokyo, Japan
nielsen@csl.sony.co.jp

**Abstract.** In a seminal paper, Amari (1998) proved that learning can be made more efficient when one uses the intrinsic Riemannian structure of the algorithms' spaces of parameters to point the gradient towards better solutions. In this paper, we show that many learning algorithms, including various boosting algorithms for linear separators, the most popular top-down decision-tree induction algorithms, and some on-line learning algorithms, are spawns of a generalization of Amari's natural gradient to some particular non-Riemannian spaces. These algorithms exploit an intrinsic dual geometric structure of the space of parameters in relationship with particular integral losses that are to be minimized. We unite some of them, such as AdaBoost, additive regression with the square loss, the logistic loss, the top-down induction performed in CART and C4.5, as a single algorithm on which we show general convergence to the optimum and explicit convergence rates under very weak assumptions. As a consequence, many of the classification calibrated surrogates of Bartlett *et al.* (2006) admit efficient minimization algorithms.

## 1 Introduction

This paper is an attempt to unite some supervised learning algorithms that have led the last decade on iterative learning algorithms, and bring some novel performance- or structural-related results. Among the algorithms concerned, there are AdaBoost and related boosting algorithms, top-down decision tree induction algorithms (including those of CART, C4.5), and some on-line learning algorithms.

Our starting point is a result of [1], which states that gradient-based learning leads to better results if one takes into account in the gradient the Riemannian structure of the space of parameters. During the last decade, most of the successes in supervised learning algorithms have been obtained when minimizing functions that serve as primers for the minimization of the empirical risk — functions called surrogates. This is the case for AdaBoost, additive logistic regression, decision tree induction, Support Vector Machines, on-line learning algorithms [14,23,25,36,38], and many others. These surrogates work on spaces of parameters, on which they define singular geometries — generally, they are

F. Nielsen (Ed.): ETVC 2008, LNCS 5416, pp. 175–215, 2009.

not symmetric and do not obey the triangular inequality. A significant amount of work has recently been devoted to set in order the huge set of candidate surrogates. For example, statistical consistency properties have been shown for a wide set containing most of the surrogates relevant to learning, *classification calibrated surrogates* [5]; other important properties, like the algorithmic questions about minimization, have been explicitly left as important problems to settle [5]. A relevant contribution on this side came earlier from [29], who proved mild convergence results on an algorithm inducing linear separators and working on a large class of convex surrogates, not necessarily classification calibrated; [29] also left as an important problem the necessity to fully solve this algorithmic question, such as by providing convergence rates.

In this paper, we show that our algorithms of interest can be seen as particular geodesic walks on the space of parameters, and these geodesic walk take benefit of the non Riemannian geometric structure of this space to progress towards better or optimal solutions, thus generalizing the Riemannian approach of [1]. Informally, the update of parameters is located on iso-Bregman divergence surfaces, progressing towards the minimization of various functions — edges in boosting, Bregman divergences in on-line learning. This progression scheme is very efficient: we show that a very large subset of classification calibrated losses may be minimized by a single boosting algorithm, with guaranteed rates of convergence under weak assumptions. We also show that this algorithm unifies various boosting algorithms, ranging from the top-down induction for decision trees performed in CART [9], C4.5 [34] to AdaBoost [36] and other boosting algorithms for linear separators. Thus, this geometric approach and its performances do not pertain to a specific kind of formalism for classifiers.

Our contribution is also structural: we show that a particular subset of classification calibrated surrogates has analytical, statistical and classification rationales, and strong ties with the maximum likelihood estimation for a subset of the exponential families of distributions. Finally, we provide experimental results on various surrogates, using a single surrogate to learn, or making attempts to tune the surrogate at hand for a more efficient optimization.

In Section 2, we present the learning settings; Section 3 presents the losses and surrogates, and their properties. Section 4 presents the related geometric problems. Section 5 presents our applications on linear separators, and Section 6 does the same for other classifiers. Section 7 presents our experimental results.

## 2 Learning Settings

### 2.1 General Considerations on Supervised Learning

Bold faced variables such as $\boldsymbol{w}$ and $\boldsymbol{x}$, represent vectors whose dimension shall be clear from context. Unless otherwise stated, sets are represented by calligraphic upper-case alphabets, *e.g.* $\mathcal{X}$, and enumerated following their lower-case, such as $\{\boldsymbol{x}_i : i = 1, 2, ...\}$ for vector sets, and $\{x_i : i = 1, 2, ...\}$ for other sets. Blackboard faces such as $\mathbb{S}$ denote subsets of (powers of) $\mathbb{R}$, the set of real numbers.

Machine learning refers in general to the possibility, for a computer, to improve its performances based on its experience. This definition may be formalized as the automatized construction of models to minimize losses based on training data. At the risk of oversimplifying, a major trend focuses on *supervised learning* (that we sometimes call *classification* for short) with *batch* or *on-line* algorithms.

The key input of supervised learning is the notion of *example*. An example is an ordered pair $(\boldsymbol{o}, c)$. The *observation* $\boldsymbol{o}$ belongs to a domain $\mathcal{O}$ of dimension $n$ (such as $\{0,1\}^n$, $\mathbb{R}^n$, the set of all patient descriptions, etc.), and the *class* $c$ belongs to a set

$$\mathcal{C} = \{c^+, c^-\} \tag{1}$$

of *two* classes or labels (such as "ill/not ill" if the observations describe patients). We adopt the common convention that $c^+$ is called the positive class, and $c^-$ the negative class.

The common problem to batch and on-line learning is the *efficient* and *accurate* automated construction of *classifiers*. Without entering into unnecessary details, in both settings, "efficiency" essentially means for the learning algorithms to be polynomial in relevant parameters. A classifier is a function $H : \mathcal{O} \to \mathcal{C}$, which belongs to a set defined by a particular formalism, whose choice is generally made by the user. This choice defines an absolute *bias*: a bias since it influences learning, and absolute since it is not questioned once it is done [28,32]. The freedom in the choice of the classifier is of primary importance, as users sometimes feel uncomfortable with some formalisms, in particular when it comes to interpreting the classifiers themselves [26,32]. Learning algorithms for any formalism should thus be appreciated in the light of their portability, their scalability to other formalisms.

There are many formalisms available, two of which are of primary importance, as they are the most frequently used in supervised learning: linear separators (LS) and decision trees (DT).

A LS is a weighted linear vote:

$$H(\boldsymbol{o}) \doteq \sum_t \alpha_t h_t(\boldsymbol{o}) \ , \tag{2}$$

with real leveraging coefficients, $\alpha_t$, and votes, sometimes called features, that can be themselves classifiers $h_t(.) : \mathcal{O} \to \mathbb{O} \subseteq \mathbb{R}$. The output of these classifiers is not necessarily $\mathbb{R}$: the simplest case is $\mathbb{O} = \{-1, 1\}$; sometimes, one also uses features that *abstain*, with $\mathbb{O} = \{-1, 0, 1\}$, and so on until $\mathbb{R}$ itself [32,36]. In general, $\mathbb{O}$ is centered around the origin (hereafter, this property shall be assumed when using notation $\mathbb{O}$). Regarding such real votes, the notation of the classes in (1) is not convenient anymore. It is more convenient to carry out a first abstraction of the classes by a bijective mapping:

$$c \in \{c^-, c^+\} \rightleftharpoons y^* \in \{-1, +1\} \ .$$

The convention is $c^+ \rightleftharpoons +1$ and $c^- \rightleftharpoons -1$. We thus have two distinct notations for an example: $(\boldsymbol{o}, c)$, $(\boldsymbol{o}, y^*)$, that shall be used without ambiguity. Let us define

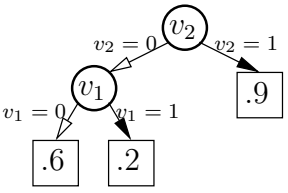


**Fig. 1.** A DT with 3 leaves and 2 internal nodes

threshold function $\sigma : \mathbb{R} \to \{-1, +1\}$, $+1$ iff $x \geq 0$ and $-1$ otherwise. Then the class assigned by a LS $H$ is $\sigma \circ H$.

An ordinary, 2-ary DT, is a rooted directed acyclic tree whose internal nodes have outdegree two (see Figure 1). Each internal node is labeled by a description variable, and each of its outgoing arcs is labeled by a Boolean test over this variable, in such a way that the two outgoing arcs of an internal nodes have complementary tests over the variable. Assume for example that $\mathcal{O} = \{0, 1\}^n$, that is, observations are represented with Boolean variables. In Figure 1, arcs with a black arrow symbolize the test that the related Boolean variable is 1 (**true**), while arcs with a white arrow symbolize the test that the related Boolean variable is 0 (**false**). Starting from the root, an observation $\boldsymbol{o} \in \mathcal{O}$ follows the path whose tests it satisfies, until it reaches a leaf used to predict its class. The fundamental difference between DT and LS is that DT makes a partition of $\mathcal{O}$ in cells (convex for ordinary decision trees) of constant values, and this partition is made in a recursive fashion, which is not the case for LS[1]. It is thus very convenient to put, at each leaf, a constant value representing the class assigned to all observations that reach the leaf. In addition to the two first already exposed, a third convention is used, which consists in putting a value in $[0, 1]$ being an "estimator" for the positive class membership probability for leaf $k$:

$$H(\boldsymbol{o}) = \hat{\mathbf{Pr}}[c = c^+ | \boldsymbol{o} \text{ reaches leaf } k] \ . \tag{3}$$

A second abstraction of the classes in (1) becomes convenient in this case:

$$c \in \{c^-, c^+\} \rightleftharpoons y \in \{0, 1\} \ .$$

The convention is $c^+ \rightleftharpoons 1$ and $c^- \rightleftharpoons 0$. We have one more notation for an example, that shall be used without ambiguity with the two others: $(\boldsymbol{o}, y)$. We define a second threshold function $\tau : [0, 1] \to \{0, 1\}$, 1 iff $x \geq 1/2$ and 0 otherwise. Then, the class assigned by a DT $H$ is $\tau \circ H$.

The quality of a classifier $H$ on example $(\boldsymbol{o}, c)$ is obtained by comparing the prediction $H(\boldsymbol{o})$ to the true class $c$ of the example, via a *loss* function $\ell$. Intuitively, $\ell$ is an increasing function of the discrepancy between $c$ and the output of $H$. The simplest and most natural loss, which historically served as

[1] Modulo some technical assumptions on the votes $h_t$ that virtually systematically hold, any LS also defines a partition of $\mathcal{O}$ in regions of constant values, but this partition is not recursive.

the basis for supervised learning models [37], is the *0/1 loss*, $\ell^{0/1}(c, H)$, which may be defined in two equivalent ways depending on $\mathrm{im}(H)$:

$$\ell^{0/1}_{\mathbb{R}}(y^*, H) \doteq 1_{y^* \neq \sigma \circ H} \text{ if } \mathrm{im}(H) = \mathbb{O} \ , \tag{4}$$

$$\ell^{0/1}_{[0,1]}(y, H) \doteq 1_{y \neq \tau \circ H} \text{ if } \mathrm{im}(H) = [0,1] \ . \tag{5}$$

Here, $1_\pi$ is the indicator variable that takes value 1 iff predicate $\pi$ is **true**, and 0 otherwise. In the general loss case, wherever needed for a clear distinction of the output of $H$, we put in index to $\ell$ an indication of its image ($\mathbb{R}$, meaning it is actually some $\mathbb{O} \subseteq \mathbb{R}$, or $[0,1]$). Sometimes, we also put in exponent an indication of the loss name, as we have done in (4) and (5) for the 0/1 loss. Both losses $\ell_{\mathbb{R}}$ and $\ell_{[0,1]}$ are defined simultaneously via popular *transfer* functions, such as the *logit* transform [14]:

$$\mathrm{logit}(p) \doteq \log \frac{p}{1-p} \ , \forall p \in [0,1]. \tag{6}$$

The following examples on the 0/1 loss are easy to check:

$$\ell^{0/1}_{[0,1]}(y, H) = \ell^{0/1}_{\mathbb{R}}(y^*, \mathrm{logit}(H)) \ ,$$

$$\ell^{0/1}_{\mathbb{R}}(y^*, H) = \ell^{0/1}_{[0,1]}(y, \mathrm{logit}^{-1}(H)) \ .$$

We have implicitly closed the domain of the logit, adding two symbols $\pm\infty$ to ensure that the eventual infinite values for $H$ can be scaled back to $[0,1]$. Hereafter, functions on which the closure of both the domain and the image is assumed are called *admissible*. The loss is then used in a general *risk* which aggregates the losses over a particular set of examples. The way this risk is computed is different in batch and on-line learning.

### 2.2 Our Batch Setting

The most important model of batch learning is the so-called *PAC* (for Probably Approximately Correct) learning model of Valiant [37]. In this model, learning has two requirements, one which is computational, and the other, which is statistical. The input is a sample $\mathcal{S} = \{(\boldsymbol{o}_1, c_1), (\boldsymbol{o}_2, c_2), ..., (\boldsymbol{o}_m, c_m)\}$ of training examples, supposed sampled i.i.d. from a subset of $\mathcal{O} \times \mathcal{C}$. The sampling distribution is unknown, but fixed. Thus, we cannot require that the classifier built on $\mathcal{S}$ be a perfect match for the class of any example of the whole domain, as for example it may be the case that we sample the same example $m$ times. Rather, the statistical constraint is a slightly weaker form of generalization constraint, as we want that $H$ represents a good approximation, with high probability, of the true labeling of the examples. It turns out that modulo some structural conditions on the formalism of $H$, a sufficient condition to learn is to build classifiers with a small overall loss over $\mathcal{S}$, that is, a small *empirical risk* (we follow a convention of [10]):

$$\varepsilon^{0/1}(\mathcal{S}, H) \doteq \sum_i \ell^{0/1}(c_i, H(\boldsymbol{o}_i)) \ . \tag{7}$$

The 0/1 loss has a significant drawback: is not differentiable and not continuous (see Figure 4). Thus, while it is the criterion used to evaluate classifiers, it precludes important candidate machineries for its minimization, and calls in fact for other risks to optimize. For this objective, we define out of a general loss $\ell$ a general *risk* $\varepsilon$:

$$\varepsilon(\mathcal{S}, H) \doteq \sum_i \ell(c_i, H(\boldsymbol{o}_i)) \ . \tag{8}$$

To finish up with batch learning, we focus on a particular batch process for learning which is iterative by nature. In *boosting* [13,22,24,33], we suppose that $H$ is built in a iterative, greedy fashion: at each main step $t$ of the algorithm, we request for a *weak classifier* $h_t$ to a *weak learner*, and $H$ is built by repeatedly folding in all the weak classifiers obtained so far. This growing process for $H$ must ensure that the current classifier keeps the desired formalism: when $H$ is a LS, weak classifiers are simply summed up in the overall vote. When $H$ is a DT, each weak classifier is a decision tree with a single internal node (a *stump*) which replaces a leaf in the current DT $H$. With respect to on-line learning (see below), boosting is dual in the sense that the stream of examples of on-line learning becomes a "stream" of weak classifiers in boosting.

### 2.3 Our On-Line Setting

The setting of on-line learning is inherently iterative but quite different from batch learning [15,25]. In particular, we are not given set $\mathcal{S}$. Rather, we receive examples one by one, out of an infinite stream of examples. The endless nature of the stream is important: should the stream be supposed to end at some moment, the difference with batch learning would be superficial; in particular, efficient learning would essentially boil down to waiting for the stream to end, and then batch learn on the examples seen so far.

In this endless supply of examples, a convenient thing to do is to repeatedly update the current classifier $H$ each time an example is received, to stay close to some *reference* classifier. We start with an initial "guess" classifier $H_0$ which can be a constant prediction, and we repeat infinitely many times (i) the receipt of example $(\boldsymbol{o}_t, c_t)$, (ii) the update of classifier $H_{t-1}$ to classifier $H_t$, for $t = 1, 2, ....$ At iteration $t$, if we denote $\mathcal{S}_t$ the set of current examples received so far, a natural objective for learning is to ensure that the set of classifiers that have been built so far has achieved over the stream of examples a 0/1 loss which stays close to that of a *reference* classifier taken from the same set of classifiers as ours. More formally, this objective may be formalized as requiring, for any $T > 0$ and any reference classifier $H_r$:

$$\sum_{t=1}^{T} \ell^{0/1}(c_t, H_{t-1}(\boldsymbol{o}_t)) \leq \sum_{t=1}^{T} \ell^{0/1}(c_t, H_r(\boldsymbol{o}_t)) + \text{ penalty } , \tag{9}$$

or, from a more general standpoint,

$$\sum_{t=1}^{T} \ell(c_t, H_{t-1}(\boldsymbol{o}_t)) \leq \sum_{t=1}^{T} \ell(c_t, H_r(\boldsymbol{o}_t)) + \text{ penalty } ; \tag{10}$$

this objective becomes perhaps more meaningful from the goodness-of-fit standpoint, if we think of $H_r$ as achieving a convenient minimization of the right-hand side of (9). This, however, may require for $H_r$ a bit more than simply minimizing the risk of the right-hand side, as the penalty can depend upon $H_r$ as well. The role of complexity is naturally pregnant in on-line learning, as the stream may provide us with examples at extremely fast pace. To finish up with on-line learning, we extend the general risk definition in (8) to its relevant twin in on-line learning:

$$\varepsilon\left(\mathcal{S}_T, \{H_t\}_{t=0}^{T-1}\right) \doteq \sum_{t=1}^{T} \ell(c_t, H_{t-1}(\boldsymbol{o}_t)) \ , \forall T > 0 \ , \tag{11}$$

so that (10) may be written: $\varepsilon\left(\mathcal{S}_T, \{H_t\}_{t=0}^{T-1}\right) \leq \varepsilon(\mathcal{S}_T, H_r) +$ penalty.

## 3 Bregman Loss Functions

We now present alternative losses that may be used in both batch and on-line learning. Most interestingly, the same alternatives emerge almost independently out of analytical, classification or statistical rationales.

### 3.1 Integral Losses

Linear separators, decision trees, and many other classifiers, are naturally "more powerful" than what is required since they encode a much greater number of values than the set of two classes. These values typically belong to two kinds of sets. The first contains signed intervals $\mathbb{O}$ typically centered on 0. In the case of LS, it would be $\mathbb{R}$ itself. The second can be reduced to the interval $[0, 1]$ or positive intervals centered on $1/2$.

Suppose that there exists an admissible *transfer function* [25] between these two kinds of sets: a strictly increasing (thus invertible) continuous function $\varrho : \mathbb{O} \to [0, 1]$. Figure 2 presents an example of a transfer function. Monotonicity provides a convenient mapping of the classes between the two kinds of possible outputs of a classifier, as the more "probable" the membership probability to the positive class and the larger (and positive) the real value, and reciprocally. For this reason, we use the transfer function to compute a discrepancy between two predictions, as the area depicted in Figure 2. This discrepancy is called an *integral loss*.

**Definition 1.** *The integral loss with parameter $\varrho$ between two predictions $y$ and $y'$ in $[0, 1]$ is:*

$$\ell_{\mathbb{R}}^{\varrho}(H, H') \doteq \int_{H}^{H'} (\varrho(x) - y)\mathrm{d}x \ , \tag{12}$$

*with $H \doteq \varrho^{-1}(y)$ and $H' \doteq \varrho^{-1}(y')$.*

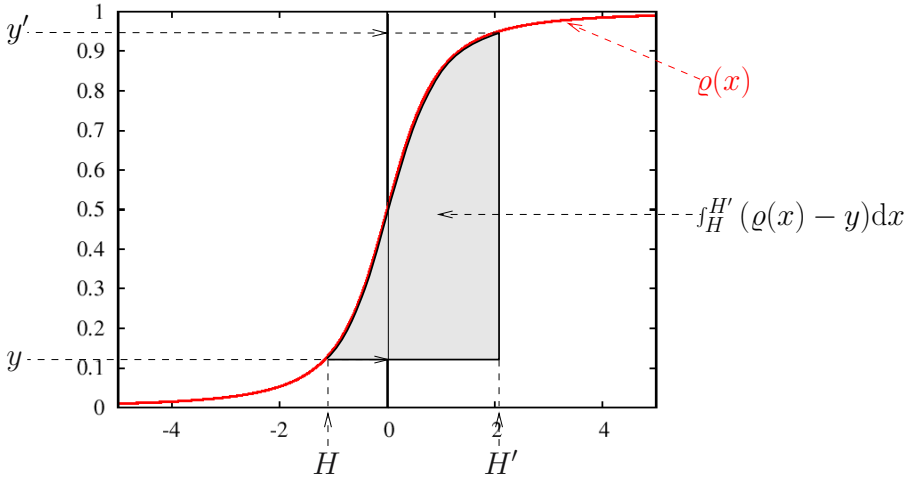


**Fig. 2.** The loss as computed in (12) for transfer function $\varrho$

$\mathrm{logit}^{-1}$ is an example of a transfer function (6). The rightmost column of Table 2 presents other examples of transfer functions. Figure 3 shows that the 0/1 loss is not an integral loss, but a limit case, considering that the integrals can be computed from limit histograms and the 0/1 loss is the simplest non-trivial case of histogram. Figure 2 displays an invariance of the integral loss which is particularly convenient for classification. Indeed, because $\varrho$ is invertible, we have:

$$\ell_{\mathbb{R}}^{\varrho}(H, H') = \int_{y'}^{y} (\varrho^{-1}(x) - H')\mathrm{d}x = \ell_{\mathbb{R}}^{(\varrho^{-1})}(y', y) \ . \tag{13}$$

Both integrals in eq. (12) and (13) can be rephrased as follows, if we denote $\psi \doteq \int \varrho$ and $\psi^\star \doteq \int \varrho^{-1}$:

$$\begin{aligned} \ell_{\mathbb{R}}^{\varrho}(H, H') &= \psi(H') - \psi(H) - (H' - H)\varrho(H) \\ &= \psi^\star(y) - \psi^\star(y') - (y - y')\varrho^{-1}(y') \\ &= \ell_{\mathbb{R}}^{(\varrho^{-1})}(y', y) \ . \end{aligned}$$

Integral losses coincide with a particular kind of distortion measure: Bregman Loss Functions [3].

**Definition 2.** *For any strictly convex function $\psi : \mathbb{X} \to \mathbb{R}$ defined over a closed interval $\mathbb{X}$ of $\mathbb{R}$, differentiable over the opened interval, the Bregman Loss Function (BLF, [3]) $D_\psi$ with generator $\psi$ is:*

$$D_\psi(x||x') \doteq \psi(x) - \psi(x) - (x - x')\nabla_\psi(x') \ , \tag{14}$$

*where $\nabla_\psi$ denotes the first derivative of $\psi$.*

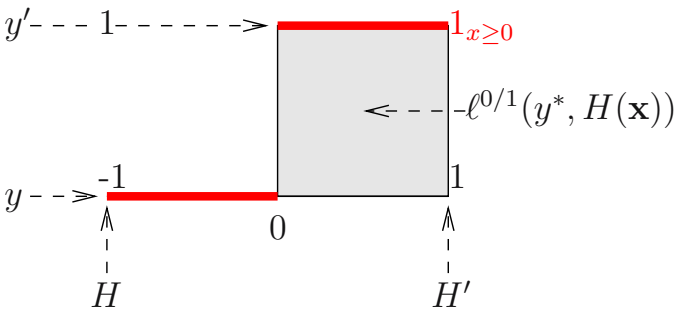


**Fig. 3.** The 0/1 loss is a limit case of integral loss for function $1_{x\geq 0}$, and real-valued classifiers that can only take on values in $\{-1, 1\}$

The parameter $\varrho$ in the integral loss is thus the derivative of the BLF generator. We also make use of a vector-based notation for BLFs, and it shall mean a component-wise sum of BLFs, such as:

$$D_\psi(\boldsymbol{a}||\boldsymbol{b}) = \sum_i D_\psi(a_i||b_i) \ . \tag{15}$$

BLFs define a subset of a set of distortion measures with interesting geometric features: Bregman divergences [8].

**Definition 3.** *For any strictly convex function $\psi : \mathbb{X} \to \mathbb{R}$ defined over a closed convex set $\mathbb{X}$ of $\mathbb{R}^m$, and continuously differentiable over its relative interior, the Bregman divergence $D_\psi$ with generator $\psi$ is:*

$$D_\psi(\boldsymbol{x}||\boldsymbol{x'}) \doteq \psi(\boldsymbol{x}) - \psi(\boldsymbol{x}) - (\boldsymbol{x} - \boldsymbol{x'})^\top \boldsymbol{\nabla}_\psi(\boldsymbol{x'}) \ , \tag{16}$$

*where $\boldsymbol{\nabla}_\psi$ denotes the gradient of $\psi$.*

Table 1 gives some examples of Bregman divergences. The first three are *separable* as they satisfy (15) [12]. Kullback-Leibler is convex in its both parameters, while Itakura-Saito is not; the Squared Euclidean distance is symmetric — in fact, Mahalanobis distortion is the only symmetric Bregman divergence [31].

In our context, it shall be useful to think the gradients as being admissible, thus making it possible to extend the corrresponding Bregman divergences to $\mathbb{X}^2$, and not simply $\mathbb{X}$ times its relative interior. The invariance described in (13) is captured with the following important notion.

**Definition 4.** *For any strictly convex function $\psi : \mathbb{X} \to \mathbb{R}$ defined over a closed convex set $\mathbb{X}$ of $\mathbb{R}^m$, the Legendre transform $\psi^\star$ of $\psi$ is:*

$$\psi^\star(\boldsymbol{x}) \doteq \sup_{\boldsymbol{x'}} \{\boldsymbol{x}^\top \boldsymbol{x'} - \psi(\boldsymbol{x'})\} \ , \tag{17}$$

*where $\boldsymbol{x}$ belongs to the relative interior of $\mathbb{X}$.*

**Table 1.** Examples of Bregman divergences (spd = symmetric positive definite)

| $\psi(\boldsymbol{x})$ | $D_\psi(\boldsymbol{x}\|\|\boldsymbol{x}')$ | Name |
|---|---|---|
| $\sum_i x_i \log x_i - x_i$ | $\sum_i \{x_i \log(x_i/x_i') - x_i + x_i'\}$ | Kullback-Leibler |
| $\sum_i -\log x_i$ | $\sum_i \{(x_i/x_i') - \log(x_i/x_i') - 1\}$ | Itakura-Saito |
| $\sum_i x_i^2$ | $\sum_i (x_i - x_i')^2$ | Squared Eucl. dist. |
| $\boldsymbol{x}^\top \Sigma \boldsymbol{x}$ | $(\boldsymbol{x} - \boldsymbol{x}')^\top \Sigma (\boldsymbol{x} - \boldsymbol{x}')$ | Mahalanobis ($\Sigma$ spd) |
| $(1/2)\|\|\boldsymbol{x}\|\|_q^2$ $= \frac{1}{2}\left[\left(\sum_i \|x_i\|^q\right)^{1/q}\right]^2$ | $(1/2)\|\|\boldsymbol{x}\|\|_q^2 - (1/2)\|\|\boldsymbol{x}'\|\|_q^2 - (\boldsymbol{x} - \boldsymbol{x}')^\top \boldsymbol{\nabla}_\psi(\boldsymbol{x}')$ $\boldsymbol{\nabla}_\psi(\boldsymbol{x})_i = \sigma(x_i)\|x_i\|^{q-1}/\|\|\boldsymbol{x}\|\|_q^{q-2}$ | $q$-norm ($q > 1$) |

Because of the strict convexity of $\psi$, the analytic expression of the Legendre transform becomes:

$$\psi^\star(\boldsymbol{x}) \doteq \boldsymbol{x}^\top \nabla_\psi^{-1}(\boldsymbol{x}) - \psi(\nabla_\psi^{-1}(\boldsymbol{x})) \ . \tag{18}$$

$\psi^\star$ is also strictly convex and differentiable. There are two important results to note, that easily follow from the identity $\nabla_\psi = \nabla_{\psi^\star}^{-1}$:

$$\begin{aligned} \psi^{\star\star} &= \psi \ , \\ D_\psi(\boldsymbol{x}\|\|\boldsymbol{x}') &= D_{\psi^\star}(\nabla_\psi(\boldsymbol{x}')\|\|\nabla_\psi(\boldsymbol{x})) \ , \end{aligned}$$

and that latter equality is just a generalization of (13). To summarize, BLFs have a strong analytical rationale to compute the losses of classifiers with dense outputs like LS and DT. The 0/1 loss is a limit case of BLF.

### 3.2 Surrogate Losses and Classification Calibrated Losses

We now introduce a popular formalization for losses over real-valued classifiers. A serious alternative to directly minimizing (7) is to rather focus on the minimization of a *surrogate risk* [5] (surrogate, for short). This is a function $\varepsilon(\mathcal{S}, H)$ as in (8) whose *surrogate loss* $\ell(c, H(\boldsymbol{o}))$ satisfies

$$\ell^{0/1}(c, H(\boldsymbol{o})) \leq \ell(c, H(\boldsymbol{o})) \ .$$

Four surrogate losses are particularly important in supervised learning:

$$\ell_{\mathbb{R}}^{\exp}(y^*, H) \doteq \exp(-y^* H) \ , \tag{19}$$

$$\ell_{\mathbb{R}}^{\log}(y^*, H) \doteq \log(1 + \exp(-y^* H)) \ , \tag{20}$$

$$\ell_{\mathbb{R}}^{\mathrm{sqr}}(y^*, H) \doteq (1 - y^* H)^2 \ , \tag{21}$$

$$\ell_{\mathbb{R}}^{\mathrm{hinge}}(y^*, H) \doteq \max\{0, 1 - y^* H\} \ . \tag{22}$$

(19) is the exponential loss, (20) is the logistic loss, (21) is the squared loss and (22) is hinge loss. The first three are examples of *strictly convex losses*; all are plotted in Figure 4.

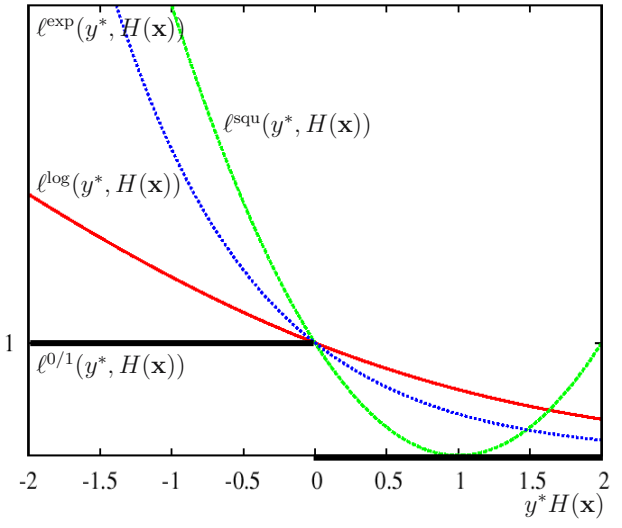


**Fig. 4.** The surrogate losses in (19), (20), (21) versus the 0/1 loss

**Definition 5.** *A Strictly Convex Loss (*SCL*) is a strictly convex function* $\psi : \mathbb{X} \to \mathbb{R}_+$ *differentiable on the opened interval and such that* $\nabla_\psi(0) < 0$*, with* $\mathbb{X}$ *centered on zero.*

Once again, it shall be useful to think of the derivative as being admissible. There is an immediate link between the 0/1 loss and any SCL, as dividing the SCL by $\psi(0) > 0$ immediately yields an upperbound of the 0/1 loss. Thus, any efficient machinery to minimize the surrogate yields an indirect minimization of the empirical risk.

We now explain the rationale of SCL, and in particular its importance for classification. Following [5], we first define classification calibrated losses (CCL). Suppose that all examples in $\mathcal{S}$ have the same observation, $\boldsymbol{o}$, and the two classes are in proportion $\eta$ for the positive class, and $1 - \eta$ for the negative class. Any general surrogate $\varepsilon_{\mathbb{R}}$ with surrogate loss $\ell_{\mathbb{R}}$ simplifies over this sample and can be written:

$$\varepsilon_{\mathbb{R}}(\eta, H) \doteq \eta \ell_{\mathbb{R}}(H) + (1 - \eta)\ell_{\mathbb{R}}(-H) \ , \tag{23}$$

where $H$ is a real constant prediction. Classification calibration requires that, for any $\eta \neq 1/2$, the minimal risk is smaller than the minimal risk in which we require $H$ to be of a different sign than $2\eta - 1$. More precisely, $\ell_{\mathbb{R}}$ belongs to classification calibrated losses (CCL) iff:

$$\varepsilon^+_{\mathbb{R}}(\eta) < \varepsilon^-_{\mathbb{R}}(\eta) \ , \forall \eta \neq 1/2 \ , \tag{24}$$

$$\varepsilon^+_{\mathbb{R}}(\eta) \doteq \inf_H \varepsilon_{\mathbb{R}}(\eta, H) \ ,$$

$$\varepsilon^-_{\mathbb{R}}(\eta) \doteq \inf_{H : H(2\eta - 1) \leq 0} \varepsilon_{\mathbb{R}}(\eta, H) \ . \tag{25}$$

**Table 2.** Correspondence between permissible functions, the corresponding BCLs and the transfer functions

| $\phi(x)$ | $a_\phi$ | $\mathrm{im}(\nabla_\phi) \supseteq \mathrm{im}(H)$ | $F_\phi(y^*H) = (\phi^\star(-y^*H) - a_\phi)/b_\phi$ | $\hat{\mathbf{Pr}}_\phi[c = c^+ \vert H; \boldsymbol{o}] = \nabla_\phi^{-1}(H)$ |
|---|---|---|---|---|
| (27) | $\mu$ | $\mathbb{R}$ | $\frac{1}{1-\mu}\left(-y^*H + \sqrt{(1-\mu)^2 + (y^*H)^2}\right)$ | $\frac{1}{2}\left(1 + H/\sqrt{(1-\mu)^2 + H^2}\right)$ |
| (28) | 0 | $\mathbb{R}$ | $-y^*H + \sqrt{1 + (y^*H)^2}$ | $\frac{1}{2}\left(1 + H/\sqrt{1 + H^2}\right)$ |
| (29) | 0 | $\mathbb{R}$ | $\log(1 + \exp(-y^*H))$ | $\exp(H)/(1 + \exp(H))$ |
| (26) | 0 | $[-1, 1]$ | $(1 - y^*H)^2$ | $\frac{1}{2}(1 + H)$ |

In our setting, quantity $2\eta - 1 \in [-1, 1]$ is the inverse of the transfer function for the following convex function:

$$\psi = \phi_{\mathrm{B}}(x) \doteq -x(1-x) \ . \tag{26}$$

We could have replaced this transfer function by many other examples, like for example logit in (6). We use this particular case because (25) is the original definition of classification calibrated losses of [5]. Furthermore, (26) is to play an important role later.

To summarize, condition $H(2\eta - 1) \leq 0$ in (25) imposes $H$ to be an overall wrong real prediction on $\mathcal{S}$. Thus, condition (24) states that from the efficient minimization of the surrogate necessarily follows the most accurate prediction of the classes, for every observation. Failing to meet this weak condition would make the surrogate meaningless for classification purposes. It follows from [5], Theorem 4, that SCL$\subset$CCL, spanning all strictly convex differentiable and classification calibrated losses, such as (19), (20), (21).

So far, we have defined two main classes of losses, integral losses with some analytical rationale, and strictly convex losses with some classification rationale. There seems to be a visual difference between these two classes of losses, as the former distinguish class $y$ and prediction $H$ as different parameters, while the latter aggregate $y^*$ and $H$ in a single parameter, $y^*H$ which can be called an *edge* (see *e.g.* (19) — (22)). The following section shows that this difference is essentially superficial.

### 3.3 Balanced Convex Losses

Let us adopt a principled approach on what should be a "good" loss function for classification. Ne need to import the main three assumptions that underlie classification losses in a majority of works in supervised learning. These assumptions, stated for $\mathrm{im}(H) \subseteq [0, 1]$ without loss of generality, are:

(**A1**) *The loss is lower-bounded* by 0. We have:

$$\ell_{[0,1]}(.,.) \geq 0 \ .$$

(**A2**) *The loss is a proper scoring rule.* Consider a singleton domain $\mathcal{O} = \{\boldsymbol{o}\}$. Then, the best (constant) prediction is:

$$\arg \min_{H \in [0,1]} \varepsilon_{[0,1]}(\mathcal{S}, H) = \eta \doteq \hat{\mathbf{P}}\mathbf{r}[c = c^+ | \boldsymbol{o}] \in [0,1] \ ,$$

where $p$ is the proportion of positive examples with observation $\boldsymbol{o}$.

(**A3**) *The loss is symmetric* in the following sense:

$$\ell_{[0,1]}(y, H) = \ell_{[0,1]}(1 - y, 1 - H), \forall y \in \{0,1\}, \forall H \in [0,1] \ .$$

Lower-boundedness in **A1** is standard. For **A2**, we can equivalently write an analogue of (23) for $[0,1]$ classifiers:

$$\varepsilon_{[0,1]}(\mathcal{S}, H) = \varepsilon_{[0,1]}(\eta, H) = \eta \ell_{[0,1]}(1, H) + (1 - \eta)\ell_{[0,1]}(0, H) \ ,$$

which is just the expected loss of zero-sum games used in [18] (eq. (8)) with Nature states reduced to the class labels. The fact that the minimum is achieved at $H = \eta$ makes the loss a proper scoring rule. $\eta$ also defines *Bayes classifier*, *i.e.* the one which minimizes the 0/1 loss [2]. **A3** scales to $H \in [0,1]$ a well-known symmetry in the *cost matrix* that holds for domains without class dependent misclassification costs. This $2 \times 2$ matrix, L, gives

$$l_{ij} \doteq \ell(i - 1, j - 1)$$

for any values $(i, j) \in \{1, 2\}^2$. Usually, it is admitted that $\ell(1,1) = \ell(0,0)$, *i.e.* right classification incurs the same loss regardless of the class. Generally, this loss is zero. Problems *without* class-dependent misclassification costs, on which focus the vast majority of theoretical studies, also make the assumption that $\ell(1,0) = \ell(0,1)$. Assumption **A3** scales theses two properties to $H \in [0,1]$. We now extend a terminology due to [23]

**Definition 6.** *A function $\phi : [0,1] \to \mathbb{R}_+$ is permissible iff $-\phi$ is differentiable on $(0,1)$, strictly concave, symmetric about $x = 1/2$, and with $-\phi(0) = -\phi(1) \doteq a_\phi \geq 0$.*

Hereafter, $\phi$ refers to a permissible function. Once again, it shall be useful to think of the derivative of $\phi$ as being admissible (assuming the logit closure we make in Subsection 2.1, all permissible functions are admissible). Definition 6 relies on $-\phi$ rather than $\phi$ because $-\phi$ is the function generally used, as it spans *e.g.* a subset of the generalized entropies [18], or it represents the function actually used for the induction of some classifiers including DT (see Section 6) [19,23]. For all popular permissible $\phi$, we have $a_\phi = 0$ [23]. We let $b_\phi \doteq -\phi(1/2) - a_\phi > 0$. In addition to (26), below are more examples of permissible functions $\phi$, that have been arranged from the bottom-most to the topmost function (when scaled so that $\phi(1/2) = -1$).

$$\phi_\mu(x) \doteq -(\mu + (1 - \mu)\sqrt{x(1 - x)}) \ , \forall \mu \in (0,1) \ . \tag{27}$$

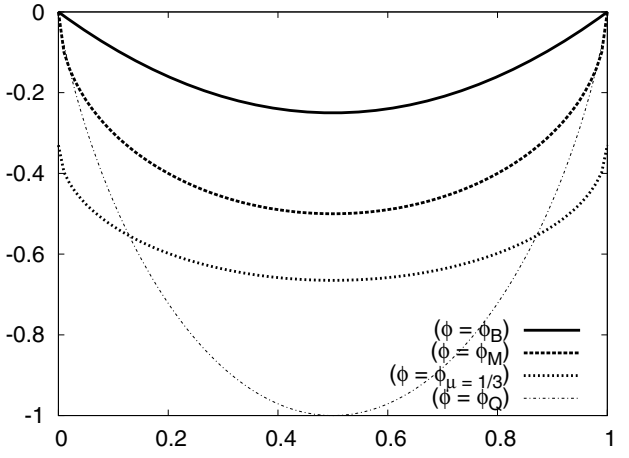


**Fig. 5.** The permissible functions in (26), (27), (28), (29)

$$\phi_{\mathrm{M}}(x) \doteq -\sqrt{x(1-x)} \ , \tag{28}$$

$$\phi_{\mathrm{Q}}(x) \doteq x \log x + (1-x)\log(1-x) \ , \ . \tag{29}$$

When scaled so that $\phi(1/2) = -1$, most confound with the opposite of popular choices: Gini index for (26) [9], Bit-entropy for (29) [34], and Matsushita's error for (28) [23,27]. Figure 5 plots these permissible functions. Finally, we say that loss $\ell_{[0,1]}$ is *properly defined* iff $\mathrm{dom}(\ell_{[0,1]}) = [0,1]^2$ and it is twice differentiable on $(0,1)^2$. This last definition is only a technical convenience: even the 0/1 loss coincides on $\{0,1\}$ with properly defined losses. In addition, the differentiability condition would be satisfied by many popular surrogates. Hinge loss (22) is a notable exception, yet it plays a key role in the properties of *balanced convex surrogates*, for which the following Lemma is central.

**Lemma 1.** *Any loss $\ell_{[0,1]}(.,.)$ is properly defined and satisfies assumptions **A1**, **A2** and **A3** if and only if*

$$\ell_{[0,1]}(y, H) = D_\phi(y||H) \ ,$$

*for some permissible function $\phi$.*

**Proof:** ($\Leftarrow$) Assumption **A3** follows from the strict concavity and symmetry of $-\phi$. Assumptions **A1** and **A2** follow from usual properties of BLFs [3]. ($\Rightarrow$) Without assumption **A3**, $\ell_{[0,1]}(y, H)$ is a BLF [3], $D_\phi(y||H)$ for some strictly convex function $\phi$, differentiable on $(0,1)$. Modulo rearrangements in assumption **A3**, we obtain

$$\nabla_{\tilde{\phi}}(H) = (\tilde{\phi}(H) - \tilde{\phi}(y))/(H-y), \forall y, H \in [0,1] \ ,$$

with $\tilde{\phi}(x) = -\phi(1-x) + \phi(x)$. It comes that $\tilde{\phi}(x) = ax + b$ for some $a, b \in \mathbb{R}$. Since $\tilde{\phi}(1-x) = -\tilde{\phi}(x)$, we easily obtain $a = b = 0$, *i.e.* $\phi(x) = \phi(1-x)$.

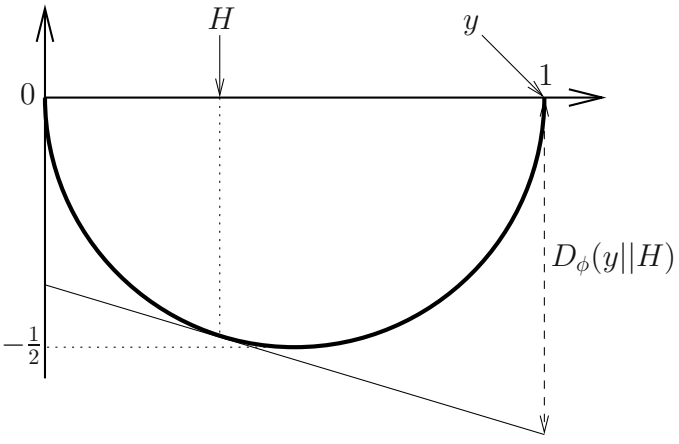


**Fig. 6.** Plot of $\ell(y, H) = D_\phi(y||H)$ (Lemma 1) for $\phi = \phi_{\mathrm{M}}(x)$ in (28) and $y, H \in [0, 1]$

Ultimately, since a BLF $D_\phi(y||H)$ does not change by adding a constant term to $\phi$, we can suppose without loss of generality that $\phi(0) = \phi(1) = -a_\phi \leq 0$, which makes that $\phi$ is permissible. □

$\phi$ is thus the "signature" of the loss. We insist on the fact that we could have replaced **A1** by a simple lower-boundedness condition without reference to zero, in which case from Lemma 1 the loss would be a BLF plus a constant factor, without impact on the structural or minimization properties that are to come. Using Lemma 1, Figure 6 depicts an example of $\ell(y, H)$ for $\phi$ as in (28). Permissible functions are useful to define the following subclass of SCL, of particular interest

**Definition 7.** *Let $\phi$ permissible. The Balanced Convex Loss (*BCL*) with signature $\phi$, $F_\phi$, is:*

$$F_\phi(x) \doteq (\phi^\star(-x) - a_\phi)/b_\phi \ . \tag{30}$$

Balanced Convex Surrogates (BCS) are defined accordingly as sums of BCL:

$$\varepsilon^\phi_{\mathbb{R}}(\mathcal{S}, H) \doteq \sum_i F_\phi(y_i^* H(\boldsymbol{o}_i)) \ . \tag{31}$$

All BCL share a common shape. Indeed, $\phi^\star(x)$ satisfies the following relationships:

$$\phi^\star(x) = \phi^\star(-x) + x \ , \tag{32}$$

$$\lim_{x \to \mathrm{infim}(\nabla_\phi)} \phi^\star(x) = a_\phi \ . \tag{33}$$

Noting that $F_\phi(0) = 1$ and $\nabla_{F_\phi}(0) = -(1/b_\phi)\nabla_\phi^{-1}(0) < 0$, it follows that BCS $\subset$ SCS, where the strict inequality comes from the fact that (19) is a SCL but not a BCL. It also follows

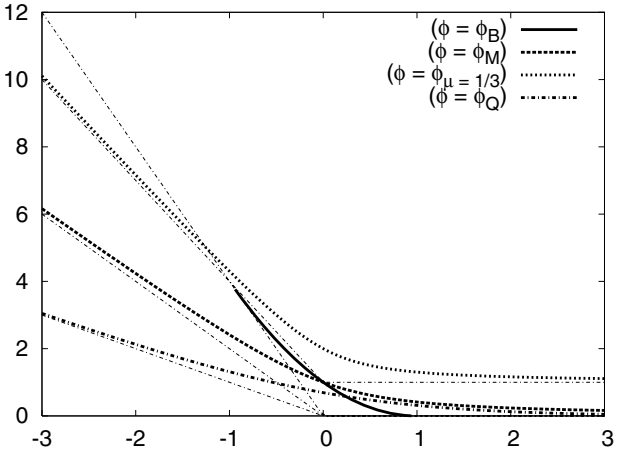


**Fig. 7.** Bold curves depict plots of $\phi^\star(-x)$ for the $\phi$ in (26), (27), (28), (29); thin dotted half-lines display parts of its asymptotes

$$\lim_{x \to \mathrm{supim}(\nabla_\phi)} F_\phi(x) = 0 \text{ from (33)} \ ,$$
$$\lim_{x \to \mathrm{infim}(\nabla_\phi)} F_\phi(x) = -x/b_\phi \text{ from (32)} \ .$$

We get that the asymptotes of any BCL can be summarized as

$$\underline{\ell}(x) \doteq x(\sigma(x) - 1)/(2b_\phi) \ . \tag{34}$$

When $b_\phi = 1$, this is the linear hinge loss [16], a generalization of (22) for which $x \doteq y^* H - 1$. Thus, while hinge loss is not a BCL, it is the limit behavior of any BCL. Figure 7 presents examples of BCL. Figure 8 presents the inverse of the transfer functions for the same $\phi$ as in Figure 7. The additional sigmoid curve indexed by variable $\zeta$, is for a permissible $\phi_\zeta$ as follows ($\forall \zeta \in \mathbb{R}_{-,*}$):

$$\phi_\zeta(x) \doteq -\frac{2}{\zeta} \log \cosh \left( \frac{\zeta}{2} \left( x - \frac{1}{2} \right) \right) \ . \tag{35}$$

When properly scaled, this permissible function is located in between $2\phi_{\mathrm{B}}$ in (26) and $-\min\{x, 1-x\}$ (and strictly in between for $x \neq 0, 1/2, 1$) — in this last case, the corresponding transfer function converges to the 0/1 loss. Tuning $\zeta \in \mathbb{R}_{-,*}$ makes the function span all the available area. It was chosen to show that there can be different concave/convex regimes for $\nabla_\phi$. Since $\mathrm{dom}(F_\phi) = \mathrm{im}(\nabla_\phi)$, there are also much different domains for the BCLs.

The following Lemma states some relationships that are easy to check using $\psi^{\star\star} = \psi$. They are particularly interesting when $\mathrm{im}(H) = \mathbb{O} \subseteq \mathbb{R}$.

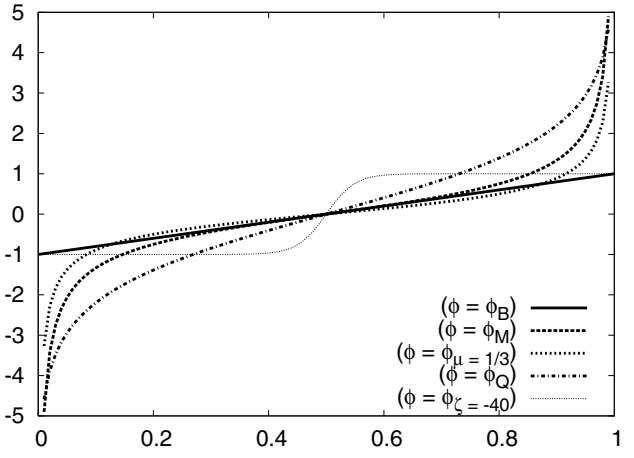


**Fig. 8.** Plots of $\nabla_\phi$ for the same $\phi$ as in Figure 7, plus an additional one that displays a particular regime (see text for details)

**Lemma 2.** *For any* SCL $\psi$,

$$\psi(y^*H) = D_{\psi^\star}(0||\nabla_{\psi^\star}^{-1}(y^*H)) - \psi^\star(0) \ . \tag{36}$$

*Furthermore, for any* BCL $F_\phi$,

$$D_\phi(y||\nabla_\phi^{-1}(H)) = b_\phi F_\phi(y^*H) \ , \tag{37}$$

$$= D_\phi(1||\nabla_\phi^{-1}(y^*H)) \ . \tag{38}$$

*Finally, for any* SCL $\psi$, *there exists a functions* $\varphi$ *such that*

$$D_\varphi(y||\nabla_\varphi^{-1}(H)) = \psi(y^*H) \tag{39}$$

*iff the restriction of* $\psi^\star(-x)$ *to the interval* $\nabla_\psi^{-1}([-1,0])$ *is permissible.*

**Proof:** The equalities are straightforward to check, so we concentrate on the last property, that we only have to check for the implication $\Rightarrow$. We note that:

$$D_\varphi(y||\nabla_\varphi^{-1}(H)) = \varphi(y) + \varphi^\star(H) - yH \ . \tag{40}$$

Making the difference of (39) for $y = 0$ and $y = 1$ yields $\psi(H) = \psi(-H) - H + (\varphi(1) - \varphi(0))$, *i.e.* $\varphi(0) = \varphi(1)$ (obtained with $H = 0$). Differentiating (39) = (40) in $H$ for $y = 0$ yields $\nabla_{\varphi^\star}(x) = -\nabla_\psi(-x)$, *i.e.* $\varphi(x) = \psi^\star(-x) + K$. We easily obtain $\nabla_\varphi(1/2) = 0$ and $\nabla_\varphi(x) = -\nabla_\varphi(1-x)$, *i.e.* $\varphi$ is permissible. □

To summarize, up to additive constants that play no role in their minimization, integral losses and strictly convex losses coincide; they are just different writings of the same losses, that can be used as an efficient primer for the minimization

of the 0/1 loss. Both match a wide subset of classification calibrated losses, and contain a set called balanced convex losses. In addition to the analytical and classification rationales of its supersets, BCL coincides with losses that match **A1** — **A3**. The transfer function defines the signature of the loss, a crucial part of the loss. The following Subsection shows that BCL has another rationale which ties its minimization to maximum likelihood estimation for popular families of densities, whose members are precisely parameterized by the signature of the loss.

### 3.4 Balanced Convex Losses and Exponential Families of Distributions

(37) tells us that the transfer function as used in Figure 2 is in fact the inverse of the permissible function's gradient ($\nabla_\phi^{-1}$) in a BCL. This provides us with an estimator of the membership probability to the positive class, if $H$ is such that $\mathrm{im}(H) = \mathbb{O}$:

$$\hat{\mathbf{Pr}}_\phi[c = c^+|H; \boldsymbol{o}] \doteq \nabla_\phi^{-1}(H(\boldsymbol{o})) \ . \tag{41}$$

We can shed some statistical light in the estimation given in (41), illustrated in Table 2 (rightmost column). This exploits famous distributions known as the exponential families of distributions [30]. Prominent members include Bernoulli (multinomial), normal, Poisson, Laplacian, negative binomial, Rayleigh, Wishart, Dirichlet, and Gamma distributions, but we shall only need a subset which turns out to be in bijection with the set of all BCLs. More precisely, using the general bijection between BLFs and the exponential families of distributions of [4], there exists through eq. (36) a bijection between the set of BCL and a subset of these exponential families whose members' pdfs may be written:

$$\mathbf{Pr}_\phi[y|\theta] = \exp(-D_\phi(y||\nabla_\phi^{-1}(\theta)) + \phi(y) - \nu(y)) \ ,$$

where $\theta \in \mathbb{R}$ denotes the natural parameter of the pdf, and $\nu(.)$ is used for normalization. Plugging $\theta = H(\boldsymbol{o})$, using (36) and (41), we obtain that any BCS (31) can be rewritten as:

$$\varepsilon_{\mathbb{R}}^{\phi}(\mathcal{S}, H) = u + \sum_i - \log \hat{\mathbf{Pr}}_\phi[y_i|H(\boldsymbol{o}_i)] \ ,$$

where $u$ does not play a role in the minimization of the BCS with $H$. We obtain the following Lemma, in which we suppose again $\mathrm{im}(H) = \mathbb{O}$.

**Lemma 3.** *Minimizing any* BCS *with classifier $H$ yields a maximum likelihood estimation, for each observation $\boldsymbol{o}$, of the natural parameter $\theta = H(\boldsymbol{o})$ of an exponential family defined by signature $\phi$.*

Real-valued hypotheses like linear separators may thus be viewed as estimating the natural parameters; by duality, classifiers that are able to fit $[0, 1]$ values, like decision trees, would rather be considered estimating the expectation parameters

of the corresponding exponential families, *i.e.* obtained via the transfer function, $\nabla_\phi^{-1}(\theta)$ (Subsection 3.1).

To end up, only one exponential family is in fact concerned in our setting. Assuming $y \in \{0, 1\}$, the pdf simplifies and we end up with

$$\mathbf{Pr}_\phi[y|\theta] = \frac{1}{1 + \exp(-\theta)} \ ,$$

the logistic prediction for a Bernoulli prior. To summarize, minimizing any surrogate whose loss meets **A1**, **A2** and **A3** (*i.e.* any BCS) amounts to the same ultimate goal. The crux of the choice of the BCS mainly relies on algorithmic and data-dependent considerations for its efficient minimization.

### 3.5 Margins

It has been soon remarked in machine learning that the output of classifiers returning real values is useful beyond its thresholding via functions $\sigma$ or $\tau$ (Subsection 2.1). In fact, we can also retrieve a measure of its "confidence" [36]. For example, when $\mathrm{im}(H) = \mathbb{O}$, it can be its absolute value [36]. Intuitively, learning should aim at providing classifiers that decide right classes with large confidences. Integrating both notions of class and confidence in criteria to optimize was done via margins [33,35]. Informally, the (normalized) margin of $H$ on example $(\boldsymbol{o}, y^*)$, $\mu_H((\boldsymbol{o}, y^*))$, takes value in $[-1, 1]$; it is positive only when the class assigned by $H$ is the right one, and its absolute value quantifies the confidence in the classification. Different definitions of margins coexist, each of which tailored to a particular kind of classifier, with a particular kind of outputs: for example, in the case of linear separators, we may have [36,35]:

$$\mu_H((\boldsymbol{o}, y^*)) \doteq \frac{y^* \sum_t \alpha_t h_t(\boldsymbol{o})}{\sum_t \alpha_t} \ .$$

Lemma 2 suggests a general and simple margin definition that we state for $\mathrm{im}(H) = \mathbb{O}$ and any permissible $\phi$. Fix:

$$\mu_H((\boldsymbol{o}, y^*)) \doteq 2\nabla_\phi^{-1}(y^* H(\boldsymbol{o})) - 1 \ . \tag{42}$$

Eq. (42) is just a scaling of the transfer function to interval $[-1, 1]$. When $\phi$ is chosen as in (29), (42) simplifies to the margin adopted in [33] for linear separators. The link between the maximization of margins as in (42) and loss minimization comes from Lemma 2. Indeed, (38) states that the minimization of any loss that meet **A1**, **A2**, **A3** is equivalent to margin maximization. Finally, since $\phi$ is permissible, (41) yields:

$$\mu_H((\boldsymbol{o}, y^*)) = y^*(2\hat{\mathbf{P}}\mathbf{r}_\phi[c = c^+|H; \boldsymbol{o}] - 1) \ \in [-1, 1] \ . \tag{43}$$

(43) shows that the margin simplifies to a quantity homogeneous to an edge as defined in Subsection 3.2, for any permissible $\phi$. This is convenient for experiments, as we can make fair comparisons between margins for different $\phi$.

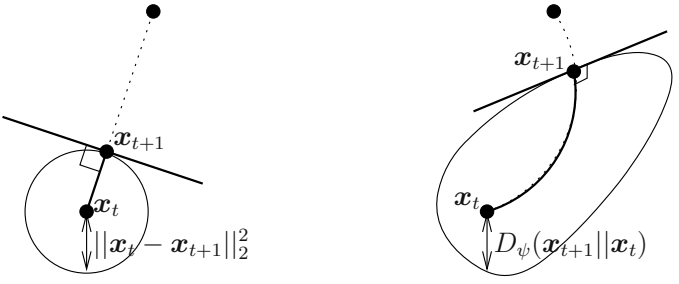


**Fig. 9.** The ordinary gradient (left), Amari's natural gradient and our geodesic walk (right) make that $\boldsymbol{x}_t$ is moved along a curve orthogonal to the Bregman ball defined by the constraint in (44) for the corresponding Bregman divergence (see text)

## 4 Bregman Geometric Problems

In this Section, our iterative settings for learning shifts to a geometric problem in which we move along a particular path in the closed convex set $\mathbb{X}$ of Definitions 2 and 3.

### 4.1 A Geodesic Walk

We are going to build in an iterative fashion elements $\boldsymbol{x}_t$, $t = 1, 2, ...T$, and seek them so as to progressively minimize a function $\vartheta(\boldsymbol{x})$, under the constraint that the step length $\kappa_t > 0$ is of fixed size, and this size is measured with a Bregman divergence $D_\psi$. More formally, we want:

$$\boldsymbol{x}_{t+1} = \arg\min_{\boldsymbol{x}\in\mathbb{X}} \vartheta(\boldsymbol{x}) \text{ s.t. } D_\psi(\boldsymbol{x}||\boldsymbol{x}_t) = \kappa_t \ , \forall t = 1, 2, .... \tag{44}$$

**Lemma 4.** *The solution to (44) is*

$$\boldsymbol{x}_{t+1} = \boldsymbol{\nabla}_\psi^{-1}\left(\boldsymbol{\nabla}_\psi(\boldsymbol{x}_t) - \frac{1}{\lambda(\kappa_t)}\boldsymbol{\nabla}_\vartheta(\boldsymbol{x}_t)\right) \ , \tag{45}$$

*for some* $\lambda(\kappa_t) \in \mathbb{R}_*$.

**Proof:** Write $\boldsymbol{x} \doteq \boldsymbol{x}_t + \epsilon\boldsymbol{a}$ for some sufficiently small constant $\epsilon$. Then we search for the $\boldsymbol{a}$ which minimizes:

$$\vartheta(\boldsymbol{x}) = \vartheta(\boldsymbol{x}_t) + \epsilon\boldsymbol{\nabla}_\vartheta^\top(\boldsymbol{x}_t)\boldsymbol{a} \ , \tag{46}$$

and the constraint becomes:

$$D_\psi(\boldsymbol{x}_t + \epsilon\boldsymbol{a}||\boldsymbol{x}_t) = \psi(\boldsymbol{x}_t + \epsilon\boldsymbol{a}) - \psi(\boldsymbol{x}_t) - \epsilon\boldsymbol{\nabla}_\psi^\top(\boldsymbol{x}_t)\boldsymbol{a} = \kappa_t \ . \tag{47}$$

The stationarity conditions of the Lagrangian give us after differentiating by $\boldsymbol{a}$ $\boldsymbol{\nabla}_\psi(\boldsymbol{x}_t+\epsilon\boldsymbol{a}) = \boldsymbol{\nabla}_\psi(\boldsymbol{x}_t)-(1/\lambda)\boldsymbol{\nabla}_\vartheta(\boldsymbol{x}_t)$ ($\lambda$ is the constraint's Lagrange multiplier), *i.e.* the statement of the Lemma. □

Lemma 4 provides us with a geodesic walk to the minimization of function $\vartheta(\boldsymbol{x})$ [31,30], namely:

$$\boldsymbol{x}_{t+1} = \boldsymbol{\nabla}_{\psi}^{-1} \left( \boldsymbol{\nabla}_{\psi}(\boldsymbol{x}_t) - \eta_t \boldsymbol{\nabla}_{\vartheta}(\boldsymbol{x}_t) \right) \ , \tag{48}$$

where $\eta_t$ is a learning rate [1,25]. This geodesic walk works only if the domains of $\vartheta$ and $\psi$ are compatible, which shall be the case in what follows. Lemma 4 brings a generalization of Amari's *natural gradient* [1]. To see this, take $D_{\psi}$ as Mahalanobis distortion (Table 1). Then (48) simplifies and we obtain that the steepest descent direction $\boldsymbol{a}$ satisfies :

$$\boldsymbol{a} = \Sigma^{-1} \boldsymbol{\nabla}_{\vartheta}(\boldsymbol{x}_t) \ .$$

This is a general form of Amari's natural gradient defined on Riemannian spaces. Lemma 4 tells that Amari's seminal notion may be generalized to a non-metric space embedded with a general Bregman divergence. The solution to (44) moves along a curve orthogonal to the ball defined by the constraint, where the orthogonality uses the Bregman-Pythagoras Theorem of Subsection 4.2 [30,31] (see Figure 9). For the sake of simplicity, we write

$$\boldsymbol{u} \diamond \boldsymbol{p} \doteq \boldsymbol{\nabla}_{\psi}^{-1} \left( \boldsymbol{\nabla}_{\psi}(\boldsymbol{p}) + \boldsymbol{u} \right) \ ,$$

where $\psi$ becomes implicit and clear from context; also, we call $\boldsymbol{u} \diamond \boldsymbol{p}$ the *Legendre dual* of the ordered pair $(\boldsymbol{u}, \boldsymbol{p})$. The Legendre dual satisfies:

$$\boldsymbol{\nabla}_{\psi}(\boldsymbol{u} \diamond \boldsymbol{p}) = \boldsymbol{\nabla}_{\psi}(\boldsymbol{p}) + \boldsymbol{u} \ , \tag{49}$$

$$\boldsymbol{u} \diamond (\boldsymbol{u}' \diamond \boldsymbol{p}) = (\boldsymbol{u} + \boldsymbol{u}') \diamond \boldsymbol{p} \ , \forall \boldsymbol{u}, \boldsymbol{u}' \in \boldsymbol{\nabla}_{\psi}(\mathbb{X}), \forall \boldsymbol{p} \in \mathbb{X} \ . \tag{50}$$

The construction of the Legendre dual can be explained in a simple way, as depicted in Figure 10 when $\psi = \phi$ is a permissible function (Definition 6).

Here is how we use the geodesic walk of Lemma 4. $\boldsymbol{x}_t$ refers to an object that we update to learn. This object, which has historically been called $\boldsymbol{w}$ rather than $\boldsymbol{x}$ since it refers to "weights", belongs to a set that we should denote $\mathbb{W}$ rather than $\mathbb{X}$. This object is different in boosting and on-line learning: in on-line learning, it refers to the classifiers $H_t$ $(t = 0, 1, ...)$, while in boosting, it refers to the learning sample $\mathcal{S}$, and more precisely, weights $\boldsymbol{w}_t$ that are put on the examples. For the boosting part, we define $m \times T$ matrix M with

$$m_{it} \doteq -y_i^* h_t(\boldsymbol{o}_i) \ . \tag{51}$$

Given leveraging coefficients vector $\boldsymbol{\alpha} \in \mathbb{R}^T$ for classifier $H$, we thus get:

$$-y_i^* H(\boldsymbol{o}_i) = (\mathrm{M}\boldsymbol{\alpha})_i \ . \tag{52}$$

In this case, $\vartheta(\boldsymbol{w}_t)$ is the *edge* of the classifier (a generalization of the notion previously defined in Subection 3.2):

$$\vartheta(\boldsymbol{w}_t) \doteq -\boldsymbol{w}_t^{\top} \mathrm{M}\boldsymbol{\alpha} \ , \tag{53}$$

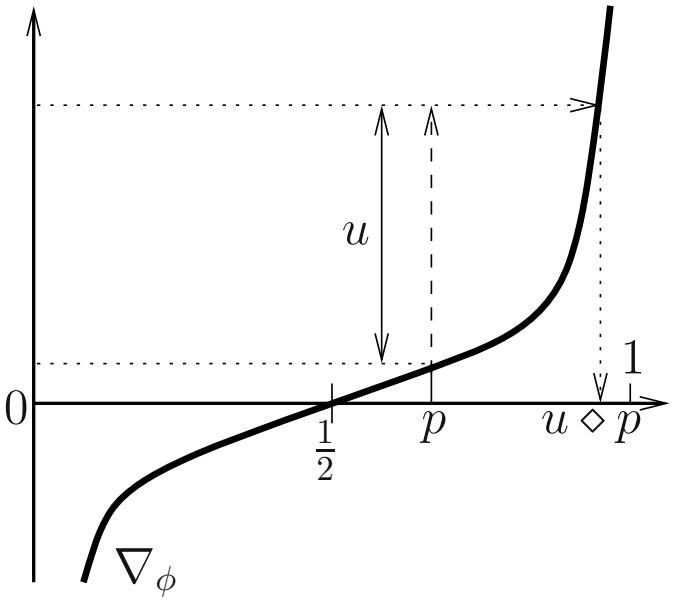


**Fig. 10.** Construction of the Legendre dual $u \diamond p$ for some permissible $\phi$. $\nabla_\phi$ is symmetric around point $(1/2, 0)$, for any permissible $\phi$.

and we get

$$\boldsymbol{\nabla}_\vartheta(\boldsymbol{w}_t) = \boldsymbol{\nabla}_\vartheta(\boldsymbol{\alpha}) = -\mathrm{M}\boldsymbol{\alpha} \ . \tag{54}$$

In on-line learning, we shall consider for the sake of simplicity only linear separators as classifiers. Thus, we let $H_t(\boldsymbol{o}) \doteq \boldsymbol{w}_t^\top \boldsymbol{o}, t = 0, 1, ...$, and consider a general BCL as in (37); in our on-line learning context (10), $\vartheta$ refers to the BCL over the current example, and we get:

$$\vartheta(\boldsymbol{w}_{t-1}) \doteq D_\phi(y_t || \nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t)) \ . \tag{55}$$

We obtain:

$$\boldsymbol{\nabla}_\vartheta(\boldsymbol{w}_{t-1}) = \boldsymbol{\nabla}_\vartheta(\boldsymbol{w}_{t-1}, \boldsymbol{o}_t, y_t) = (\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t) - y_t)\boldsymbol{o}_t \ . \tag{56}$$

### 4.2 A Bregman-Pythagoras Theorem

The following Lemma states a fundamental property on Bregman divergences.

**Lemma 5.** *For any elements* $\boldsymbol{u}, \boldsymbol{x}, \boldsymbol{x}'$ *such that* $(\boldsymbol{x}' - \boldsymbol{u})^\top (\boldsymbol{\nabla}_\psi(\boldsymbol{x}) - \boldsymbol{\nabla}_\psi(\boldsymbol{x}')) = 0$, *we have:*

$$D_\psi(\boldsymbol{u} || \boldsymbol{x}) = D_\psi(\boldsymbol{u} || \boldsymbol{x}') + D_\psi(\boldsymbol{x}' || \boldsymbol{x}) \ . \tag{57}$$

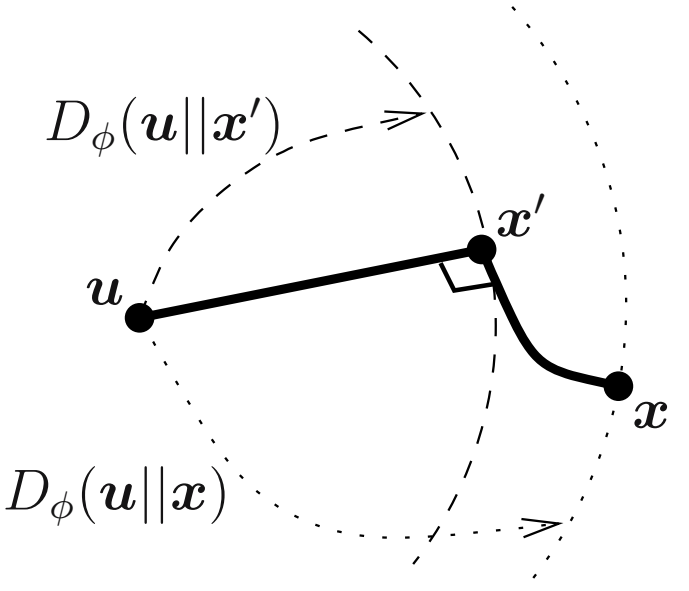


**Fig. 11.** An illustration of Bregman-Pythagoras theorem

The proof of this Lemma is immediate once we remark the three-points property:

$$D_\psi(\boldsymbol{u}||\boldsymbol{x}) = D_\psi(\boldsymbol{u}||\boldsymbol{x}') + D_\psi(\boldsymbol{x}'||\boldsymbol{x}) + (\boldsymbol{x}' - \boldsymbol{u})^\top(\boldsymbol{\nabla}_\psi(\boldsymbol{x}) - \boldsymbol{\nabla}_\psi(\boldsymbol{x}')) \ . \quad (58)$$

Lemma 5 gives us a generalization of Pythagoras theorem, that we retrieve with the squared Euclidean distance (Table 1). In the literature, some more sophisticated generalizations of Pythagoras theorem have been published [20,30]. The one we propose in Lemma 5 is sufficient for our purpose. Figure 11 gives a schematic view of Bregman-Pythagoras theorem.

### 4.3 An Optimisation Problem Associated to the Geodesic Walk

There is an interesting problem related to the geodesic walk, which relies on a point $\boldsymbol{x}_0$ of $\mathbb{X}$ and the subset of $\mathbb{X}$ which may be reached through geodesic walks starting from $\boldsymbol{x}_0$. As this problem is particularly interesting for boosting, we use the boosting formulation for $\boldsymbol{\nabla}_\varepsilon$ in (54). The set is:

$$\mathbb{X}_0 \doteq \{\mathrm{M}\boldsymbol{\alpha} \diamond \boldsymbol{x}_0 : \boldsymbol{\alpha} \in \mathbb{R}^m\} \ . \quad (59)$$

The problem we want to solve is:

$$\boldsymbol{x}_\star = \arg\min_{\boldsymbol{x} \in \overline{\mathbb{X}}_0} D_\psi(\boldsymbol{x}_1||\boldsymbol{x}) \ , \quad (60)$$

for some $\boldsymbol{x}_1 \in \mathbb{X}$. Here, $\overline{\mathbb{X}}_0$ is the closure of $\mathbb{X}_0$. There is a dual and "orthogonal" optimization problem of interest, namely:

$$\boldsymbol{x}_\star = \arg\min_{\boldsymbol{x} \in \mathbb{X}_1} D_\psi(\boldsymbol{x}||\boldsymbol{x}_0) \ , \quad (61)$$

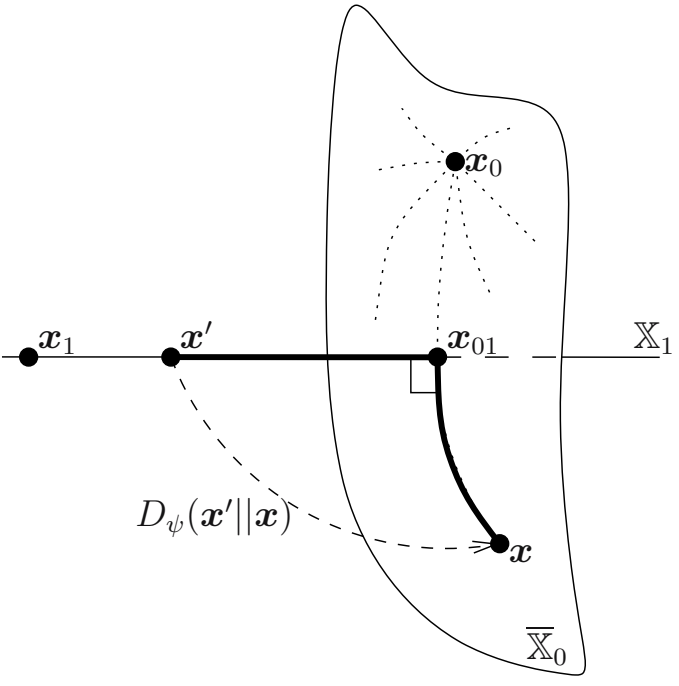


**Fig. 12.** Illustration of Lemma 6: the solutions of (60) and (61) naturally emerge as a consequence as the Bregman-Pythagoras orthogonality of $\overline{\mathbb{X}}_0$ and $\mathbb{X}_1$.

where

$$\mathbb{X}_1 \doteq \{\boldsymbol{x} \in \mathbb{X} : \mathrm{M}\boldsymbol{x} = \mathrm{M}\boldsymbol{x}_1\} \ . \tag{62}$$

What is particularly interesting is when $\overline{\mathbb{X}}_0 \cap \mathbb{X}_1 \neq \emptyset$, a condition which would not necessarily hold without the closure of $\mathbb{X}_0$ (for the technical reasons, we refer to [10]). Provided the intersection is not empty, we let:

$$\boldsymbol{x}_{01} \in \overline{\mathbb{X}}_0 \cap \mathbb{X}_1 \ . \tag{63}$$

We have an interesting four-points property met by any Bregman divergence:

$$\begin{aligned} &D_\psi(\boldsymbol{x}'_a||\boldsymbol{x}_a) - D_\psi(\boldsymbol{x}'_a||\boldsymbol{x}_b) - D_\psi(\boldsymbol{x}'_b||\boldsymbol{x}_a) + D_\psi(\boldsymbol{x}'_b||\boldsymbol{x}_b) \\ &= (\boldsymbol{x}'_a - \boldsymbol{x}'_b)^\top (\boldsymbol{\nabla}_\psi(\boldsymbol{x}_b) - \boldsymbol{\nabla}_\psi(\boldsymbol{x}_a)) \ , \forall \boldsymbol{x}_a, \boldsymbol{x}_b, \boldsymbol{x}'_a, \boldsymbol{x}'_b \in \mathbb{X} \ . \end{aligned} \tag{64}$$

Take $\boldsymbol{x}_a, \boldsymbol{x}_b \in \mathbb{X}_0$. We obtain:

$$\begin{aligned} (\boldsymbol{x}'_a - \boldsymbol{x}'_b)^\top (\boldsymbol{\nabla}_\psi(\boldsymbol{x}_b) - \boldsymbol{\nabla}_\psi(\boldsymbol{x}_a)) &\overset{(59)}{=} (\boldsymbol{x}'_a - \boldsymbol{x}'_b)^\top (\mathrm{M}\boldsymbol{\alpha}_b - \mathrm{M}\boldsymbol{\alpha}_a) \\ &= (\boldsymbol{\alpha}_b - \boldsymbol{\alpha}_a)^\top (\mathrm{M}\boldsymbol{x}'_a - \mathrm{M}\boldsymbol{x}'_b) \ , \end{aligned} \tag{65}$$

and if we make the additional assumption that $\boldsymbol{x}'_a, \boldsymbol{x}'_b \in \mathbb{X}_1$ (62), then (65) is zero. If we take $\boldsymbol{x}_a = \boldsymbol{x}'_a = \boldsymbol{x}_{01}$, then we obtain the following Lemma.

**Lemma 6.** $D_\psi(\boldsymbol{x}'||\boldsymbol{x}) = D_\psi(\boldsymbol{x}'||\boldsymbol{x}_{01}) + D_\psi(\boldsymbol{x}_{01}||\boldsymbol{x}), \forall \boldsymbol{x} \in \mathbb{X}_0, \forall \boldsymbol{x}' \in \mathbb{X}_1$.

This Lemma, illustrated in Figure 12, gives us the solutions of both (60) and (61).

**Lemma 7.** $\boldsymbol{x}_{01} = \boldsymbol{x}_\star$, *the solution of (60) and (61).*

**Proof:** Take any $\boldsymbol{x} \in \overline{\mathbb{X}}_0$ and $\boldsymbol{x}' = \boldsymbol{x}_1$ in Lemma 6: we obtain $D_\psi(\boldsymbol{x}_1||\boldsymbol{x}) = D_\psi(\boldsymbol{x}_{01}||\boldsymbol{x}) + D_\psi(\boldsymbol{x}_1||\boldsymbol{x}_{01}) \geq D_\psi(\boldsymbol{x}_1||\boldsymbol{x}_{01})$, with equality iff $\boldsymbol{x} = \boldsymbol{x}_{01}$. Hence, $\boldsymbol{x}_{01}$ is the solution to (60). The same reasoning with $\boldsymbol{x}' \in \mathbb{X}_1$ and $\boldsymbol{x} = \boldsymbol{x}_0$ in Lemma 6 yields the same result for (61). □

Thus, the optimization problems in (60) ad (61) reduce to finding an element in the intersection of $\overline{\mathbb{X}}_0$ and $\mathbb{X}_1$. Figure 12 provides us with a graphical interpretation of Lemma 7.

## 5 Applications on Linear Separators

### 5.1 Boosting

Here is the way we assemble the geometric pieces of Section 4 to devise a general boosting algorithm for LS. We keep our notations in (51) for matrix M, and show how to minimize any SCS

$$\varepsilon^\psi_{\mathbb{R}}(\mathcal{S}, H) \doteq \sum_i \psi(y_i^* H(\boldsymbol{o}_i)) \ , \tag{66}$$

for any SCL $\psi$ with $\mathrm{dom}(\psi) = \mathbb{R}$. This assumption on the domain of $\psi$ is not restrictive for LS, as otherwise it would make it necessary to truncate the outputs of the LS to remain within the domain, and it is known that such procedures present significant masking problems [14]. For some BCL, we may be forced to give up with the BCL regime to make the fitting, at the expense of the lost of the direct relationships with the maximum likelihood fitting explained in Subsection 3.4. For example, we can use (21), *but* it is necessary to get out of the BCL regime and work in $\mathbb{R}$ instead of $[-1, 1]$.

To simplify notations, we let:

$$\tilde{\psi}(x) \doteq \psi^\star(-x) \ . \tag{67}$$

With this notation, (36) becomes:

$$\psi(y^* H) = D_{\tilde{\psi}}(0||\nabla^{-1}_{\tilde{\psi}}(-y^* H)) - \tilde{\psi}(0) \ . \tag{68}$$

We let $\mathbb{W} \doteq \mathrm{dom}(\nabla_{\tilde{\psi}}) = -\mathrm{im}(\nabla_\psi)$, where this latter equality comes from $\nabla_{\tilde{\psi}}(x) = -\nabla_{\psi^\star}(-x) = -\nabla^{-1}_\psi(-x)$. It also comes $\mathrm{im}(\nabla_{\tilde{\psi}}) = \mathbb{R}$. Algorithm ULS provides us with a general induction scheme for LS, whose properties hold regardless of the SCL at hand. The step which is not explained in the algorithm

**Algorithm 1:** Algorithm ULS(M, $\psi$)

**Input**: $\mathrm{M} \in \mathbb{R}^{m \times T}$, SCL $\psi$ with $\mathrm{dom}(\psi) = \mathbb{R}$;
Let $\boldsymbol{\alpha}_1 \leftarrow \mathbf{0}$;
Let $\boldsymbol{w}_0 \leftarrow \nabla_{\tilde{\psi}}^{-1}(0)\mathbf{1}$;
**for** $j = 1, 2, ...J$ **do**
  //Weight Update: make a geodesic walk on $D_{\tilde{\psi}}$ to minimize $\vartheta =$ edge (53)

$$\boldsymbol{w}_j \leftarrow (\mathrm{M}\boldsymbol{\alpha}_j) \diamond \boldsymbol{w}_0 \ ; \tag{69}$$

  Let $\mathcal{T}_j \subseteq \{1, 2, ..., T\}$;
  Let $\boldsymbol{\delta}_j \leftarrow \mathbf{0}$;
  //Leveraging Coefficients: ensure Bregman-Pythagoras Theorem on weights

$$\forall t \in \mathcal{T}_j, \text{ pick } \delta_{j,t} \text{ such that } : \sum_{i=1}^{m} m_{it}((\mathrm{M}\boldsymbol{\delta}_j) \diamond \boldsymbol{w}_j)_i = 0 \ ; \tag{70}$$

  Let $\boldsymbol{\alpha}_{j+1} \leftarrow \boldsymbol{\alpha}_j + \boldsymbol{\delta}_j$;
**Output**: $H(\boldsymbol{o}) \doteq \sum_{t=1}^{T} \alpha_{J+1,t} h_t(\boldsymbol{o}) \in \mathrm{LS}$

is the choice of the set of indices $\mathcal{T}_j$ on which we compute the leveraging coefficients of the features. In fact, this step does not really belong to ULS: ever since the seminal works on boosting [21], this step has mostly been the retrieval, by a weak learner, of a single weak classifier — called feature here because everything is like if it were mapping each observation to a new real variable which is used to build the final LS. Thus, we can suppose for the moment that this choice is assumed by the weak learner, and we shall discuss it later. Through the more recent works on boosting, the choice of $\mathcal{T}_j$ has come with various flavors: in classical boosting, at step $j$, we would fit a single $\alpha_t$ [10]; in totally corrective boosting, we would rather fit $\{\alpha_t, 1 \leq t \leq j\}$ [39]. Intermediate schemes may be used as well for $\mathcal{T}_j$, provided they ensure that, at each step $j$ of the algorithm and for any feature $h_t$, it may be chosen at some $j' > j$. ULS is displayed in Algorithm 1. In Algorithm 1, $\mathcal{T}_j$ may be chosen according to whichever scheme underlined above.

The following Theorem provides a first general convergence property for ULS.

**Theorem 1.** *The output of ULS(M, $\psi$) converges to a classifier $H_\star$ realizing:*

$$H_\star = \arg\min_{H \in \mathrm{LS}} \varepsilon_{\mathbb{R}}^{\psi}(\mathcal{S}, H) \ . \tag{71}$$

**Proof:** In (69), (50) brings $\boldsymbol{w}_{j+1} = (M\boldsymbol{\alpha}_{j+1}) \diamond \boldsymbol{w}_0 = (M\boldsymbol{\delta}_j) \diamond \boldsymbol{w}_j$. We thus have:

$$\begin{aligned} D_{\tilde{\psi}}(\mathbf{0}||\boldsymbol{w}_{j+1}) - D_{\tilde{\psi}}(\mathbf{0}||\boldsymbol{w}_j) = & -[\tilde{\psi}((M\boldsymbol{\delta}_j) \diamond \boldsymbol{w}_j) - \tilde{\psi}(\boldsymbol{w}_j) + \boldsymbol{w}_j^\top \nabla_{\tilde{\psi}}(\boldsymbol{w}_j)] \\ & + ((M\boldsymbol{\delta}_j) \diamond \boldsymbol{w}_j)^\top \nabla_{\tilde{\psi}}((M\boldsymbol{\delta}_j) \diamond \boldsymbol{w}_j) \ . \end{aligned} \tag{72}$$

Because of (49), (72) is just (for short, $r \doteq ((M\boldsymbol{\delta}_j) \diamond \boldsymbol{w}_j)^\top \nabla_{\tilde{\psi}}(\boldsymbol{w}_j)$):

$$\begin{aligned}((M\boldsymbol{\delta}_j) \diamond \boldsymbol{w}_j)^\top \nabla_{\tilde{\psi}}((M\boldsymbol{\delta}_j) \diamond \boldsymbol{w}_j) &= r + ((M\boldsymbol{\delta}_j) \diamond \boldsymbol{w}_j)^\top M\boldsymbol{\delta}_j \\ &= r - \sum_{i=1}^{m} y_i^* \sum_{t=1}^{T} \delta_{j,t} h_t(\boldsymbol{o}_i)((M\boldsymbol{\delta}_j) \diamond \boldsymbol{w}_j)_i \\ &= r - \sum_{t=1}^{T} \delta_{j,t} \sum_{i=1}^{m} y_i^* h_t(\boldsymbol{o}_i)((M\boldsymbol{\delta}_j) \diamond \boldsymbol{w}_j)_i \\ &= r + \sum_{t=1}^{T} \delta_{j,t} \underbrace{\sum_{i=1}^{m} m_{it}((M\boldsymbol{\delta}_j) \diamond \boldsymbol{w}_j)_i}_{b_t} \\ &= r \ . \end{aligned} \tag{73}$$

(73) holds because $\delta_{j,t} = 0$, or $b_t = 0$ from the choice of $\delta_{j,t}$ in (70). We obtain:

$$D_{\tilde{\psi}}(\mathbf{0}||\boldsymbol{w}_{j+1}) - D_{\tilde{\psi}}(\mathbf{0}||\boldsymbol{w}_j) = -D_{\tilde{\psi}}(\boldsymbol{w}_{j+1}||\boldsymbol{w}_j) \ . \tag{74}$$

This is Bregman-Pythagoras Theorem on weights (Lemma 5). This relationship is fundamental for the proof. Indeed, it comes from (68) and (69) that:

$$\varepsilon_{\mathbb{R}}^{\psi}(\mathcal{S}, H_j) = D_{\tilde{\psi}}(\mathbf{0}||\boldsymbol{w}_j) - m\tilde{\psi}(0) = D_{\tilde{\psi}}(\mathbf{0}||\boldsymbol{w}_j) + \text{ const } . \tag{75}$$

Thus, (74) is the difference between two successive SCS, a measure of the progress to the limit classifier. Since any SCS is lowerbounded and the right-hand side of (74) cannot be strictly positive, ULS must converge, and so there remains to characterize the classifier obtained after convergence, *i.e.* when $\boldsymbol{w}_j = \boldsymbol{w}_{j+1}$. Take

$$\boldsymbol{x}_0 \doteq \nabla_{\tilde{\psi}}^{-1}(0)\mathbf{1} \text{ in (61)} \ , \tag{76}$$

$$\mathbb{X}_0 \doteq \mathbb{W} \text{ in (59)} \ . \tag{77}$$

$$\boldsymbol{x}_1 \doteq \mathbf{0} \text{ in (60)} \ , \tag{78}$$

$$\mathbb{X}_1 \doteq \{\boldsymbol{\alpha} \in \mathbb{R}^m : \mathrm{M}^\top \boldsymbol{\alpha} = \mathrm{M}^\top \boldsymbol{x}_1 = \mathbf{0}\} = \mathrm{Ker}\mathrm{M}^\top \text{ in (62)} \ , \tag{79}$$

Problem (60) can thus be rewritten as $\boldsymbol{w}_\star = \arg\min_{\boldsymbol{w} \in \overline{\mathbb{W}}} D_{\tilde{\psi}}(\mathbf{0}||\boldsymbol{w})$, or equivalently, with (75), as:

$$H_\star = \arg \min_{H \in \mathrm{LS}} \varepsilon_{\mathbb{R}}^{\psi}(\mathcal{S}, H) \ . \tag{80}$$

The geodesic walk in (69) and (70) ensures:

$$\mathbf{1}_{\mathcal{T}_j}^\top \mathrm{M}^\top \boldsymbol{w}_{j+1} = 0 \ , \tag{81}$$

where $\mathbf{1}_{\mathcal{T}_j}$ is the Boolean vector with ones on the indexes of $\mathcal{T}_j$. After convergence, $\mathrm{M}\boldsymbol{\delta}_j = \mathbf{0}$ for any $\mathcal{T}_j$, and hence the corresponding vector $\boldsymbol{w}_{j+1}$ is such that (81)

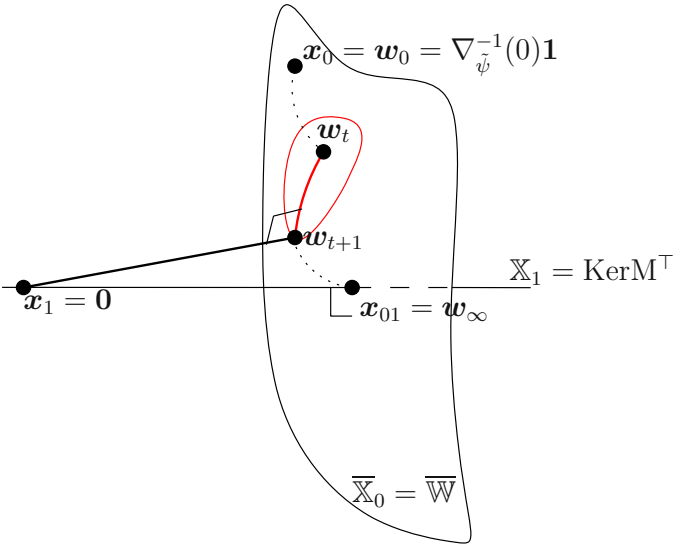


**Fig. 13.** A schematic view of how ULS behaves: iterating geodesic walks make the weight vector converge to the null space of $\mathrm{M}^\top$, and the classifier $H$ converges to the optimum of the surrogate risk $\varepsilon_{\mathbb{R}}^{\psi}(\mathcal{S}, H)$ (see text)

holds regardless of the choice of $\mathbf{1}_{\mathcal{T}_j}$. Taking (81) for each possible singleton index in $\mathcal{T}_j$, we obtain that, after convergence:

$$\mathrm{M}^\top \boldsymbol{w}_{j+1} = \mathbf{0} \;\; \Rightarrow \boldsymbol{w}_{j+1} \in \mathrm{KerM}^\top \;\; . \tag{82}$$

Hence, $\boldsymbol{w}_{j+1} = \boldsymbol{w}_\star$, the optimal solution to (60), and the corresponding classifier we end up with is $H_\star$, the solution to (80). □

We emphasize the fact that Theorem 1 proves the convergence towards the global optimum of $\varepsilon_{\mathbb{R}}^{\psi}$, regardless of $\psi$. The optimum is defined by the LS with features in M that realizes the smallest $\varepsilon_{\mathbb{R}}^{\psi}$. Figure 13 displays the way ULS behaves. The Bregman ball around $\boldsymbol{w}_j$ shows the geodesic walk made in step (69) to compute $\boldsymbol{w}_{j+1}$ by nullifying the current edge of the features selected in $\mathcal{T}_j$. Notice that in practice, it may be a tedious task to satisfy exactly (74), in particular for totally corrective boosting [39]. ULS has the flavor of boosting algorithms repeatedly modifying a set of weights $\boldsymbol{w}$ over the examples, the most popular algorithm being AdaBoost [13].

In fact, this similarity is more than syntactical, as ULS satisfies two first popular algorithmic boosting properties, the first of which being (70) which implies (81): after the computation of the leveraging coefficients, the next weights are somehow decorrelated with the classes, if we refer to the zero edge. One may wonder under which conditions (70) admits a solution. The following Lemma shows that it always admit a finite solution when no "trivial" solution exist for the minimization of the SCS at hand.

**Lemma 8.** *Suppose that there does not exist some $h_t$ with all $m_{it}$ of the same sign, $\forall i = 1, 2, ..., m$. Then, for any choice of $\mathcal{T}_j$ in ULS, (70) has always a finite solution.*

**Proof:** Let:

$$Z \doteq D_{\tilde{\psi}}(\mathbf{0}||(M\boldsymbol{\alpha}_{j+1}) \diamond \boldsymbol{w}_0) \ . \tag{83}$$

We have

$$Z = m\tilde{\psi}(0) + \sum_{i=1}^{m} \tilde{\psi}((M(\boldsymbol{\delta}_j + \boldsymbol{\alpha}_j))_i)$$

from (68), a function convex in all leveraging coefficients. Define $|\mathcal{T}_j| \times |\mathcal{T}_j|$ matrix $E$ with:

$$e_{uv} \doteq \frac{\partial^2 Z}{\partial \delta_{j,u} \partial \delta_{j,v}}$$

(for the sake of simplicity, $\mathcal{T}_j = \{1, 2, ..., |\mathcal{T}_j|\}$, where $|.|$ denotes the cardinal). We have:

$$e_{uv} = \sum_{i=1}^{m} \frac{m_{iu} m_{iv}}{\varphi(((M\boldsymbol{\delta}_j) \diamond \boldsymbol{w}_j)_i)} \ ,$$

with:

$$\varphi(x) \doteq \frac{\mathrm{d}^2 \tilde{\psi}(x)}{\mathrm{d}x^2} \ , \tag{84}$$

a function strictly positive in the relative interior of $\mathbb{W}$ since $\tilde{\psi}$ is strictly convex. Let $q_{i,j} \doteq 1/\varphi(((M\boldsymbol{\delta}_j) \diamond \boldsymbol{w}_j)_i) > 0$. It is easy to show that:

$$\boldsymbol{x}^\top E \boldsymbol{x} = \sum_{i=1}^{m} q_{i,j} (\boldsymbol{x}^\top \tilde{\boldsymbol{m}}_i)^2 \geq 0 \ , \forall \boldsymbol{x} \in \mathbb{R}^{|\mathcal{T}_j|} \ , \tag{85}$$

with $\tilde{\boldsymbol{m}}_i \in \mathbb{R}^{|\mathcal{T}_j|}$ the vector containing the entries $m_{it}$ with $t \in \mathcal{T}_j$. Thus, $E$ is positive semidefinite; as such, (70), which is the same as solving $\partial Z / \partial \delta_{j,u} = 0$, $\forall u \in \mathcal{T}_j$ (*i.e.* minimizing $Z$) has always a solution. □

The condition for the Lemma to work is absolutely not restrictive, as if such an $h_t$ were to exist, we would not need to run ULS: indeed, we would have either $\varepsilon^{0/1}(\mathcal{S}, h_t) = 0$, or $\varepsilon^{0/1}(\mathcal{S}, -h_t) = 0$.

We give three examples of specializations of ULS:

- take for example $\psi(x) = \exp(-x)$ (19). In this case, $\mathbb{W} = \mathbb{R}_+$, $\boldsymbol{w}_0 = \mathbf{1}$ and it is not hard to see that ULS matches real AdaBoost with unnormalized weights [36]. The difference is syntactical: the LS output by ULS and real AdaBoost are the same;

- now, take any BCL. In this case, $\tilde{\psi} = \phi$, $\mathbb{W} = [0,1]$ (recall that we close $\mathbb{W}$ in the same way as we did for the logit in Section 2.1), and $\boldsymbol{w}_0 = 1/2\mathbf{1}$. In all these cases, where $\mathbb{W} \subseteq \mathbb{R}_+$, $\boldsymbol{w}_j$ is always a distribution up to a normalization factor, and this would also be the case for any strictly decreasing SCS $\psi$. The BCL case brings an interesting display of how the weights behave through the geodesic walk.
  Figure 10 displays a typical Legendre dual for a BCL. Consider example $(\boldsymbol{o}_i, y_i)$, and its weight update, $w_{j,i} \leftarrow (M\boldsymbol{\alpha}_j)_i \diamond w_{0,i} = (-y_i^* H(\boldsymbol{o}_i)) \diamond w_{0,i}$ for the current classifier $H$. Fix $p = w_{0,i}$ and $u = -y_i^* H(\boldsymbol{o}_i)$ in Figure 10. We see that the new weight of the example gets larger iff $u > 0$, *i.e.* iff the example is given the *wrong* class by $H$, which is the second boosting property met by ULS;
- as a last example, take $\psi(x) = (1-x)^2$ in (21). In this case, as argued at the beginning of Subsection 5.1, we leave the BCL regime to work with the function defined over $\mathbb{R}$ instead of $[-1,1]$. Since $\nabla_{\tilde{\psi}}^{-1}(x) = 2(1-x)$, weights actually span $\mathbb{R}$ itself, and the negative regime appears for $x \geq 1$. This is not surprising as the SCL is increasing when $x \geq 1$, and so minimizing it in this region "reverses" the polarity of search with respect to the BCL regime.

ULS turns out to meet a third boosting property, and the most important as it contributes to root the algorithm in the seminal boosting theory of the early nineties: we have guarantees on its convergence rate under a generalization of the well-known "Weak Learning Assumption" (WLA) [36]. To state the WLA, we plug the iteration in the index of the distribution normalization coefficient in (83), and define $Z_j \doteq ||\boldsymbol{w}_j||_1$ ($||.||_k$ is the $L_k$ norm). The WLA is:

$$(\mathbf{WLA}) \forall j, \exists \gamma_j > 0 : \left| \frac{1}{|\mathcal{T}_j|} \sum_{t \in \mathcal{T}_j} \frac{1}{Z_j} \sum_{i=1}^{m} m_{it} w_{j,i} \right| \geq \gamma_j \ . \tag{86}$$

This is indeed a generalization of the usual WLA for boosting algorithms, that we obtain taking $|\mathcal{T}_j| = 1$, $h_t \in \{-1, +1\}$ [33]. Few algorithms are known that formally boost WLA in the sense that requiring only WLA implies guaranteed rates for the minimization of $\varepsilon_{\mathbb{R}}^{\psi}$. We show that ULS meets this property $\forall \psi \in$ SCL. To state this, we need few more definitions. Let $\boldsymbol{m}_t$ denote the $t^{th}$ column vector of $M$, $a_{\boldsymbol{m}} \doteq \max_t ||\boldsymbol{m}_t||_2$ and $a_Z \doteq \min_j Z_j$. Let $a_\gamma$ denote the average of $\gamma_j$ $(\forall j)$, and $a_\varphi \doteq \min_{x \in \mathrm{int}(\mathbb{W})} \varphi(x)$ ($\varphi$ defined in (84)).

**Theorem 2.** *Under the WLA, ULS reaches the minimum of the surrogate risk* $\varepsilon_{\mathbb{R}}^{\psi}(\mathcal{S}, H)$ *in*

$$J = \mathcal{O}\left( \frac{m a_{\boldsymbol{m}}^2}{a_\varphi a_Z^2 a_\gamma^2} \right) \tag{87}$$

*iterations.*

**Proof:** We use Taylor expansions with Lagrange remainder for $\tilde{\psi}$, and then the mean-value theorem, and obtain that $\forall w, w + \Delta \in \mathbb{W}, \exists w^\star \in [\min\{w + \Delta, w\}, \max\{w + \Delta, w\}]$ such that:

$$D_{\tilde{\psi}}(w+\Delta||w) = \Delta^2\varphi(w^\star)/2 \geq (\Delta^2/2)a_\varphi \geq 0 \ . \tag{88}$$

We use $m$ times this inequality with $w = w_{j,i}$ and $\Delta = (w_{j+1,i} - w_{j,i})$, sum the inequalities, combine with Cauchy - Schwartz and Jensen's inequalities, and obtain:

$$D_{\tilde{\psi}}(\boldsymbol{w}_{j+1}||\boldsymbol{w}_j) \geq a_\varphi \left(\frac{a_Z\gamma_j}{2a_{\boldsymbol{m}}}\right)^2 \ . \tag{89}$$

Using (74), we obtain that $D_{\tilde{\psi}}(\mathbf{0}||\boldsymbol{w}_{J+1}) - m\tilde{\psi}(0)$ equals:

$$\begin{aligned} &-m\tilde{\psi}(0) + D_{\tilde{\psi}}(\mathbf{0}||\boldsymbol{w}_1) + \sum_{j=1}^{J}\left(D_{\tilde{\psi}}(\mathbf{0}||\boldsymbol{w}_{j+1}) - D_{\tilde{\psi}}(\mathbf{0}||\boldsymbol{w}_j)\right) \\ &= m\psi(0) - \sum_{j=1}^{J} D_{\tilde{\psi}}(\boldsymbol{w}_{j+1}||\boldsymbol{w}_j) \ . \end{aligned} \tag{90}$$

But, (68) together with the definition of $\boldsymbol{w}_j$ in (69) yields

$$D_{\tilde{\psi}}(0||w_{J+1,i}) = \tilde{\psi}(0) + \psi(y_i^* H(\boldsymbol{o}_i)), \forall i = 1, 2, ..., m \ , \tag{91}$$

which ties up the SCS to (90); the guaranteed decrease in the right-hand side of (90) by (89) makes that there remains to check when the right-hand side becomes negative to conclude that ULS has reached the optimum. This gives the bound of the Theorem. □

The bound in Theorem 2 is mainly useful to prove that the WLA guarantees a convergence rate of order $\mathcal{O}(m/a_\gamma^2)$ for ULS, but not the best possible as it is in some cases far from being optimal.

To finish up with ULS, if we update a single leveraging coefficient for $h_t$ at step $j$, then the WLA can be simplified as (using the same notation as in (81)):

$$\left|\mathbf{1}_{\{t\}}^\top \mathrm{M}^\top \boldsymbol{w}_j\right| \geq \gamma_j Z_j \ . \tag{92}$$

Without loss of generality, we can suppose that the edge of $h_t$ on $\boldsymbol{w}_t$ is always strictly positive, since otherwise we can consider $-h_t$. Suppose that $h_t$ is retrieved by a weak learner, distinct from ULS: the role of the weak learner is to obtain a bottomline weak classifier $h_t$ with strictly positive edge, while ULS systematically reduces the edge of this weak classifier to zero by a geodesic walk in (69), yielding $\mathbf{1}_{\{t\}}^\top \mathrm{M}^\top \boldsymbol{w}_{j+1} = 0$ in (81), and forcing the weak learner to find a different weak classifier for the next step. Most of ULS thus reduces to an edge game parametrized by M, the weak learner picking $\mathbf{1}_{\{t\}}^\top$ while ULS picks $\boldsymbol{w}_{j+1}$. The game ends when the weak learner has no more possibility to ensure the WLA, in which case ULS has converged to the optimal classifier $H_\star$.

### 5.2 On-Line Learning

To keep this Section short, we refer the reader to *e.g.* [17,25] for more details. The principle of the learning algorithm is simpler than for ULS, as we only make the geodesic walk each time an example is received from the stream. The geodesic walk is parametrized by (55) and (56), from which we obtain Algorithm 2 [25].

We recall that function $\vartheta$ is a general BCL $D_\phi(y_t||\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t))$, and that our objective is to obtain an upperbound of the kind $\varepsilon\left(\mathcal{S}_T, \{H_t\}_{t=0}^{T-1}\right) \leq \varepsilon(\mathcal{S}_T, H_r) +$ penalty (10), where $H_r$ is a reference LS characterized by weight (leveraging) vector $\boldsymbol{r}$.

There is a particularly interesting Bregman divergence to compute a convenient penalty, the $q$-norm divergence $D_{\psi_q}$ [15,25] (Table 1). There is an interesting result regarding $D_{\psi_q}$ for OLS, namely that if we perform the geodesic walk on $D_{\psi_q}$, it is necessarily bounded. Its proof follows [25].

**Lemma 9.** *Assume* $1 < q \leq 2 \leq p < \infty$, *and* $1/p + 1/q = 1$. *Then:*

$$D_{\psi_q}(\boldsymbol{w}_{t-1}||\boldsymbol{w}_t) \leq \frac{\eta^2(p-1)}{2}(\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t) - y_t)^2 ||\boldsymbol{o}_t||_p^2 \ . \tag{94}$$

Furthermore, it can be noticed that we have:

$$D_\phi(y_t||\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t)) = \frac{\varphi(x_t)}{2}(\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t) - y_t)^2 \ , \tag{95}$$

for some $x_t$ in the interior of the interval defined by $\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t)$ and $y_t$, where $\varphi$ is as defined in (84) with $\tilde{\psi} = \phi$. There are two reasonable assumptions to make about Algorithm 2. The first states that the $p$-norm of any observation of the stream is upperbounded:

$$\sup_t ||\boldsymbol{o}_t||_p \leq \lambda_p \ . \tag{96}$$

The second states that $\varphi(x_t)$ is lowerbounded in (95), which is also reasonable given that the main quantity which could be responsible of extreme deviations in the geodesic walk is itself bounded (96):

$$\inf_t \varphi(x_t) \geq \lambda \ . \tag{97}$$

**Algorithm 2:** Algorithm OLS($\mathcal{S}$)

**Input**: Stream $\mathcal{S}$ of examples $(\boldsymbol{o}_t, c_t), t = 1, 2, ...$, with $\boldsymbol{o}_t \in \mathbb{R}^m$;
Let $\boldsymbol{w}_0 \leftarrow \boldsymbol{0}$;
**for** $t = 1, 2, ...$ **do**
  //Classifier Update: make a geodesic walk on $D_{\psi_q}$ to minimize $\vartheta =$BCL (55)

$$\boldsymbol{w}_t \leftarrow (-\eta(\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t) - y_t)\boldsymbol{o}_t) \diamond \boldsymbol{w}_{t-1} \ ; \tag{93}$$

**Output**: $H_t(\boldsymbol{o}) \doteq \boldsymbol{w}_t^\top \boldsymbol{o} \in$ LS

The following Theorem, whose proof follows [25], gives us a desired bound on the deviation of $H_t$ with respect to the reference LS $H_r$. To interpret this bound, one can take as reference the optimal vector $\boldsymbol{r}_\star$ of the optimal LS $H_\star$ in term of BCL, and suppose that $\boldsymbol{r}_\star$ has few non-zero entries. In this case, Algorithm 2 manages to stay quite close to the optimum.

**Theorem 3.** *Assume* $1 < q \leq 2 \leq p < \infty$, $1/p + 1/q = 1$, *and*

$$\eta \doteq \frac{\lambda}{(p-1)\lambda_p^2} \tag{98}$$

*in Algorithm 2 (parameters defined in (96) and (97)). Then the following holds for any* $T > 0$*:*

$$\sum_{t=1}^{T} D_\phi(\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t)||\nabla_\phi^{-1}(\boldsymbol{r}^\top \boldsymbol{o}_t)) \leq \sum_{t=1}^{T} D_\phi(y_t||\nabla_\phi^{-1}(\boldsymbol{r}^\top \boldsymbol{o}_t)) + \frac{(p-1)\lambda_p^2||\boldsymbol{r}||_q^2}{2\lambda} \ .$$

**Proof:** In the same way as we do for (74), we can measure a progress to reference $d_t$ as:

$$\begin{aligned} d_t &\doteq D_{\psi_q}(\boldsymbol{r}||\boldsymbol{w}_{t-1}) - D_{\psi_q}(\boldsymbol{r}||\boldsymbol{w}_t) \ . \qquad (99)\\ &\overset{(58)}{=} (\boldsymbol{w}_{t-1} - \boldsymbol{r})^\top (\boldsymbol{\nabla}_{\psi_q}(\boldsymbol{w}_{t-1}) - \boldsymbol{\nabla}_{\psi_q}(\boldsymbol{w}_t)) - D_{\psi_q}(\boldsymbol{w}_{t-1}||\boldsymbol{w}_t)\\ &\overset{(93)}{=} \eta(\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t) - y_t)(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t - \boldsymbol{r}^\top \boldsymbol{o}_t) - D_{\psi_q}(\boldsymbol{w}_{t-1}||\boldsymbol{w}_t) \ . \qquad (100) \end{aligned}$$

From the three-points property in (58), we obtain for any BLF $D_\phi$:

$$\begin{aligned} D_\phi(y_t||\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t)) &+ D_\phi(\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t)||\nabla_\phi^{-1}(\boldsymbol{r}^\top \boldsymbol{o}_t))\\ &- D_\phi(y_t||\nabla_\phi^{-1}(\boldsymbol{r}^\top \boldsymbol{o}_t))\\ &= (\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t) - y_t)(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t - \boldsymbol{r}^\top \boldsymbol{o}_t) \ . \qquad (101) \end{aligned}$$

Combining (100) and (101), we get:

$$\begin{aligned} d_t \quad &= \quad \eta \left\{ D_\phi(\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t)||\nabla_\phi^{-1}(\boldsymbol{r}^\top \boldsymbol{o}_t)) - D_\phi(y_t||\nabla_\phi^{-1}(\boldsymbol{r}^\top \boldsymbol{o}_t)) \right\}\\ &\quad +\eta D_\phi(y_t||\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t)) - D_{\psi_q}(\boldsymbol{w}_{t-1}||\boldsymbol{w}_t)\\ &\overset{(94),(96),(97)}{\geq} \eta \left\{ D_\phi(\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t)||\nabla_\phi^{-1}(\boldsymbol{r}^\top \boldsymbol{o}_t)) - D_\phi(y_t||\nabla_\phi^{-1}(\boldsymbol{r}^\top \boldsymbol{o}_t)) \right\}\\ &\quad +\frac{\eta}{2}(\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t) - y_t)^2 \left(\lambda - \eta(p-1)\lambda_p^2\right)\\ &\overset{(98)}{=} \eta \left\{ D_\phi(\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t)||\nabla_\phi^{-1}(\boldsymbol{r}^\top \boldsymbol{o}_t)) - D_\phi(y_t||\nabla_\phi^{-1}(\boldsymbol{r}^\top \boldsymbol{o}_t)) \right\} \ . \end{aligned}$$

We sum this inequality for $t = 1, 2, ..., T$, rearrange, and get:

$$\begin{aligned} &\sum_{t=1}^{T} D_\phi(\nabla_\phi^{-1}(\boldsymbol{w}_{t-1}^\top \boldsymbol{o}_t)||\nabla_\phi^{-1}(\boldsymbol{r}^\top \boldsymbol{o}_t))\\ &\leq \frac{(p-1)\lambda_p^2}{\lambda} \sum_{t=1}^{T} d_t + \sum_{t=1}^{T} D_\phi(y_t||\nabla_\phi^{-1}(\boldsymbol{r}^\top \boldsymbol{o}_t)) \ . \qquad (102) \end{aligned}$$

Now, we sum (99) for $t = 1, 2, ..., T$, and get:

$$\begin{aligned}\sum_{t=1}^{T} d_t &= D_{\psi_q}(\boldsymbol{r}||\boldsymbol{w}_0) - D_{\psi_q}(\boldsymbol{r}||\boldsymbol{w}_T) \\ &= \frac{1}{2}||\boldsymbol{r}||_q^2 - D_{\psi_q}(\boldsymbol{r}||\boldsymbol{w}_T) \\ &\leq \frac{1}{2}||\boldsymbol{r}||_q^2 \ . \end{aligned} \tag{103}$$

We combine (103) and (102), and get the statement of the Theorem. □

## 6 Applications to More Classifiers

In Subsection 2.1, we have presented the importance of the choice of the classifier for the user. In the preceding Section, all algorithms rely on linear separators. While the choice seems natural for on-line learning — it is easy to update a LS, while it may be much harder to cope with on-line modifications of DT —, the boosting setting calls for more applications to other formalisms. Apart from LS, the main formalisms on which boosting algorithms have been developed are decision trees [23,21]. We now show that ULS is scalable to DT as well. More precisely, one can naturally induce a DT with ULS, and it turns out that this algorithm, which immediately captures the theoretical properties of ULS, is a generalization of the most popular DT induction algorithms [9,23,34].

We refer to Subsection 2.1 for a presentation of DT. Recall that the structure of a DT makes it possible to label the leaves with $[0, 1]$ or arbitrary real values. In Figure 1, the tree uses the former convention. In Figure 14 (left), we provide a tree which is equivalent from the empirical risk standpoint, but uses signed values at the leaves. We make use some new notations that we now present.

A DT $H$ induces a partition of $\mathcal{S}$ according to subsets $\mathcal{S}_k$, where $k \in \mathcal{L}(H) \subset \mathbb{N}_*$, and $\mathcal{L}(H)$ is a subset of natural integers in bijection with the set of leaves of

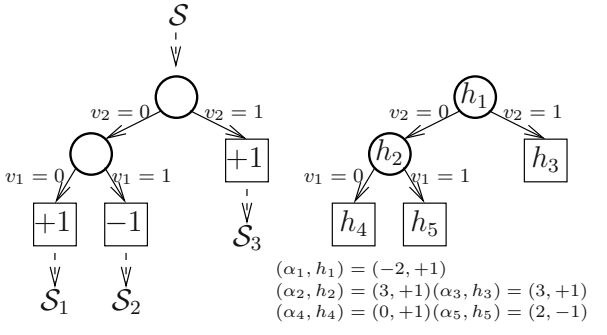


**Fig. 14.** Left: a decision tree with 3 leaves (squares) and 2 internal nodes (circles), with real-valued leaves, equivalent from the empirical risk standpoint to the DT in Figure 1; right: an equivalent linearized decision tree, for the proof of Theorem 4

**Algorithm 3:** Algorithm UDT($\mathcal{S}, \phi$)

**Input**: Learning sample $\mathcal{S}$, permissible function $\phi$;
Let $H \leftarrow$ single leaf;
**for** $j = 1, 2, ...J$ **do**
Pick some leaf $k' \in \mathcal{L}(H)$, and some observation variable $v$ such that:

$$A_\phi(H, H_{|k' \rightarrow v}) \doteq \sum_{k \in \mathcal{L}(H_{|k' \rightarrow v})} \frac{|\mathcal{S}_k|}{b_\phi} \left( -\phi \left( \frac{|S_k^+|}{|S_k|} \right) \right) - \sum_{k \in \mathcal{L}(H)} \frac{|\mathcal{S}_k|}{b_\phi} \left( -\phi \left( \frac{|S_k^+|}{|S_k|} \right) \right) < 0 \ ; \quad (104)$$

Let $H \leftarrow H_{|k' \rightarrow v}$;
**Output**: $H \in$ DT

the DT (see Figure 14). We let $S_k^+ \doteq \{(\boldsymbol{o}, c^+) \in \mathcal{S}_k\}$ denote the subset of positive examples that fall on leaf $k$. To decide a class, we can label leaves using real values to make predictions, following the convention of linear separators (used in Figure 14), or use $[0, 1]$ values. In fact, using assumption **A2** on balanced convex losses, the estimator for the class membership probabilities in (3) naturally becomes for leaf $k$:

$$\hat{\mathbf{Pr}}[c = c^+ | H; \boldsymbol{o} \text{ reaches leaf } k] = \frac{|S_k^+|}{|S_k|} \in [0, 1] \ .$$

The most popular DT induction algorithms integrate a stage in which a large DT is induced in a top-down fashion, the so-called TDIDT scheme (Top-Down Induction of DT). This scheme consists, after having initialized the DT to a single leaf, in repeatedly replacing a leaf by a sub-tree with two leaves (a stump) [9,23,34]. For this reason, it is convenient to define, for any $k \in \mathcal{L}(H)$ and any Boolean description variable $v$, $H_{|k \rightarrow v}$ to be the DT built from $H$ after having replaced leaf $k$ by the subtree of two leaves rooted at $v$. The TDIDT scheme can be conveniently abstracted as displayed in Algorithm 3. In UDT, $\phi$ is the free parameter which is instantiated with different choices to yield all popular schemes: (26) is chosen in [9], (29) is chosen in [34] and (28) is chosen in [23]. In fact, it is the *opposite* of the permissible function which is used (we would have $\phi = \tilde{\psi}$ in ULS), but we keep $\phi$ in order not to laden our notations. All popular TDIDT schemes would also also normalize $A_\phi$ (division by $m$), but this does not change the choices made for $k'$ and $v$, as after having picked $k'$, they all pick the best stump, *i.e.* the one which minimizes (104).

Because all $\phi$ considered in existing algorithms are permissible, we also restrict ourselves to balanced convex surrogates, and so we seek the minimization of the BCS in (31). In the following Theorem, we not only show that UDT achieves the minimization of any BCS with signature $\phi$: while bitterly different from each

other on paper, we show that UDT and ULS are offshoots of the same algorithm, thereby generalizing an observation of [19] to the whole family of losses that meet assumptions **A1**, **A2** and **A3**.

**Theorem 4.** *The output of UDT($\mathcal{S}$, $\phi$) converges to a classifier $H_\star$ realizing:*

$$H_\star = \arg\min_{H \in \mathrm{DT}} \varepsilon_{\mathbb{R}}^{\phi}(\mathcal{S}, H) \ . \tag{105}$$

**Proof:** The proof makes use of linearized decision trees (LDT) of [19]. A LDT has the same graph shape as a DT, but real values are put on every node (not just on leaves). The classification of some observation sums these real values over the whole path that it follows, from the root to a leaf. To each path from the root to a leaf can thus be associated a constant LS, that sums these real values. The right part of Figure 14 presents how to generate the equivalent LDT from the DT given on the left. We can indeed check that $\alpha_1 h_1 + \alpha_2 h_2 + \alpha_4 h_4 = +1$, and so on for the other leaves.

Thus, we can use ULS to build each of these LS: each feature $h_t$ is constant and put on some tree node, ULS is run on the subset of $\mathcal{S}$ that reaches the node, in order to compute the leveraging coefficient $\alpha_t$. The splits are computed after a further minimization of the given BCS.

Suppose that the current LDT $H$ has $T$ nodes, and we wish to compute $\alpha_k$ for some $h_k$ located at leaf node index $k$. To do so, we number the internal nodes using natural integers, excluding from the choices the integers chosen for the leaves. Let $\wp(k)$ be the set of indices corresponding to the path from the root to leaf $k$. The solution of (70) can be computed exactly, and yields:

$$\alpha_k = \frac{1}{h_k} \left( \nabla_\phi \left( \frac{|S_k^+|}{|S_k|} \right) - \sum_{t \in \wp(k) \setminus \{k\}} \alpha_t h_t \right) \ .$$

Thus, for any observation $\boldsymbol{o}$ that reaches leaf $k$, we get:

$$H(\boldsymbol{o}) = \nabla_\phi \left( \frac{|S_k^+|}{|S_k|} \right) \ , \tag{106}$$

naturally the inverse of (105). Finally, the BCS of $H$ simplifies as:

$$\begin{aligned}
\varepsilon_{\mathbb{R}}^{\phi}(\mathcal{S}, H) &\doteq \sum_i F_\phi(y_i^* H(\boldsymbol{o}_i)) = \frac{1}{b_\phi} \sum_i D_\phi(y_i || \nabla_\phi^{-1}(H(\boldsymbol{o}_i))) \\
&= \frac{1}{b_\phi} \sum_{k \in \mathcal{L}(H)} \sum_{(\boldsymbol{o}, y) \in \mathcal{S}_k} D_\phi \left( y \middle\| \frac{|S_k^+|}{|S_k|} \right) \\
&= \frac{1}{b_\phi} \sum_{k \in \mathcal{L}(H)} |\mathcal{S}_k| \times \left\{ \frac{|S_k^+|}{|S_k|} D_\phi \left( 1 \middle\| \frac{|S_k^+|}{|S_k|} \right) + \left( 1 - \frac{|S_k^+|}{|S_k|} \right) D_\phi \left( 0 \middle\| \frac{|S_k^+|}{|S_k|} \right) \right\} \\
&= -\frac{m a_\phi}{b_\phi} + \sum_{k \in \mathcal{L}(H)} \frac{|\mathcal{S}_k|}{b_\phi} \left( -\phi \left( \frac{|S_k^+|}{|S_k|} \right) \right) \ .
\end{aligned}$$

It is straightforward to check from this last equality and Lemma 2 that the progress to optimum in (74) ULS becomes exactly (104) in UDT. The LDT obtained is equivalent [19] (see also Figure 14) to a twin DT in which we put at leaf $k$ either:

$$\frac{|S_k^+|}{|S_k|} \in [0,1] \ ,$$

or:

$$\nabla_\phi \left( \frac{|S_k^+|}{|S_k|} \right) \in \mathrm{im}(\nabla_\phi) \subseteq \mathbb{R} \ ,$$

eventually closing once again the domain of the gradient of $\phi$ to ensure proper scalability in $[0,1]$. We finally end up with UDT. □

From (106), it comes that the $[0,1]$ value put at leaf $k$ satisfies:

$$\frac{|S_k^+|}{|S_k|} = \nabla_\phi^{-1}(H(\boldsymbol{o})) \ ,$$

with $\boldsymbol{o}$ any observation that reaches leaf $k$. From Subsection 3.4 and Lemma 3, it comes that the leaf value is also $\nabla_\phi^{-1}(\theta)$, and so fitting a DT to the minimization of a BCS yields local (leaves-based) maximum likelihood estimators of the expectation parameter of the exponential family defined by signature $\phi$.

## 7 Experiments

This section is an attempt to summarize some interesting experimental properties that seem to emerge out of the numerous surrogates and classifiers considered. To remain concise, we have chosen to focus only on ULS. We have compared against each other 11 flavors of ULS, including AdaBoost [36], on a benchmark of 52 domains (49 from the UCI repository [7]), with $32 \leq m \leq 14500$. True risks are estimated via stratified 10-fold cross validation; ULS is ran for $r$ (fixed) features $h_t$, each of which is a boolean rule: **If** Monomial **then** Class$= \pm 1$ **else** Class $= \mp 1$, with at most $l$ (fixed) literals, induced following the greedy minimization of the SCS at hand. Leveraging coefficients (70) are approximated up to $10^{-10}$ precision. Figure 15 summarizes the results.

Out of the 11 flavors, only one picks $\psi$ in SCS$\setminus$BCS (AdaBoost); the ten others exclusively rely on BCS. Out of these ten, the first four flavors pick $\phi$ in (26), (27), (28) and (29). The fifth uses another generalization of (28):

$$\phi_\upsilon(x) \doteq (x(1-x))^\upsilon \ , \forall \upsilon \in (0,1) \ . \tag{107}$$

Recent works have demonstrated the interest in fitting a metric (Mahalanobis) to the domain at hand, *prior* to using an instanced-based (*non inductive*) classification algorithm [11]. The wide range of SCS available for ULS inspired us to

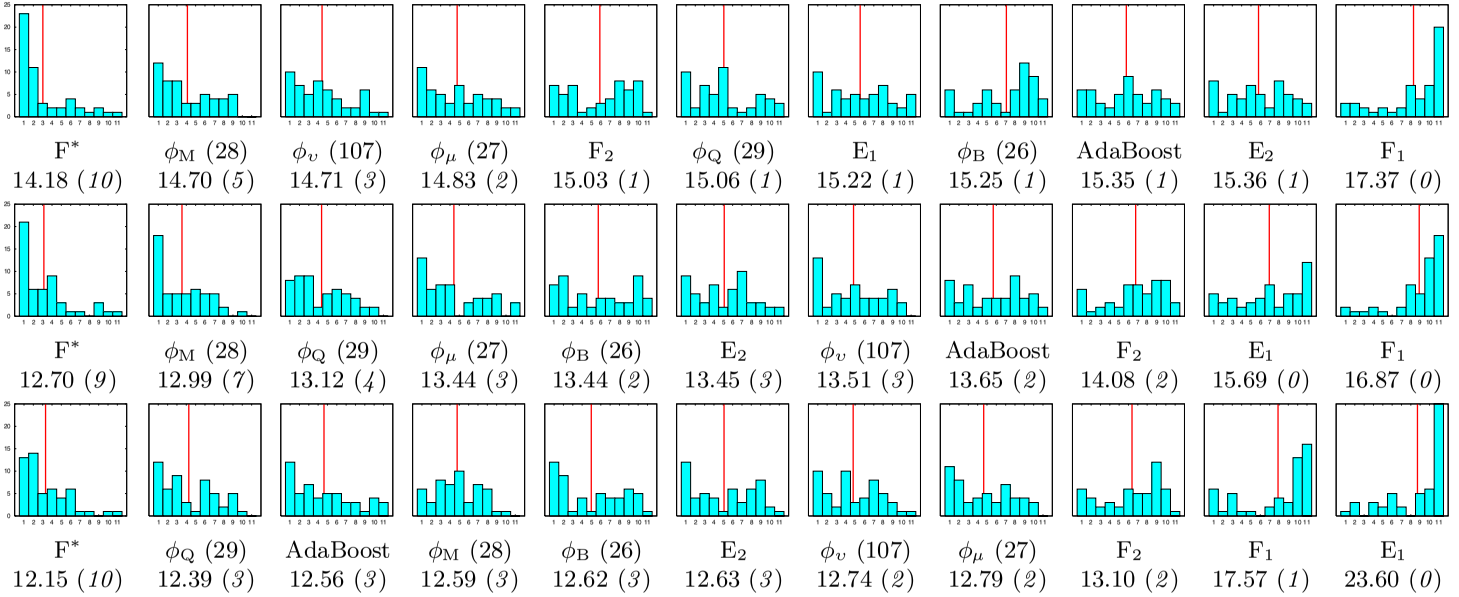


**Fig. 15.** Summary of our results over the 52 domains for the 11 algorithms. Rows are respectively for $l = 2, r = 10$ (top), $l = 10, r = 20$ (middle), $l = 3, r = 100$ (bottom). In each row, ranking histograms (wrt true risks) are given for each algorithm; vertical (red) bars show the average rank over all domains. Histograms are ordered from left to right in increasing average true risk over all domains (shown below histograms). The *italic* numbers give, for each algorithm, the number of algorithms it *beats* according to a Student paired t-test over all domains with .1 threshold probability.

kill two birds in one shot: mix the *adaptive tuning* of a SCS to the domain at hand with the *inductive learning* of a LS with ULS on this domain. We relate experiments on the adaptive tuning of a BCS out-of-a-bag of BCS. This gives the five last flavors of ULS with BCS. The first four fit the BCS at *each stage* of the inner loop (**for** $j$ ...) of ULS. Two (noted "$F_.$") pick the BCS which minimizes the empirical risk in the bag; two others (noted "$E_.$") pick the BCS which maximizes the current edge. There are two different bags corresponding to four permissible functions each: the first (index "1") contains (26), (27), (28) and (29); the second (index "2") contains (27), (28), (29) and (107). We wanted to evaluate (26) because it forces to renormalize the leveraging coefficients in $H$ each time it is selected, to ensure that the output of $H$ lies in $[-1, 1]$. The last adaptive flavor, $F^*$, "externalizes" the choice of the BCS: it selects for each fold the BCS which yields the smallest empirical risk in a bag corresponding to five $\phi$: (26), (27), (28), (29) and (107). It was suggested by the fact that, if ULS resists overfitting as AdaBoost does, we might hope for good performances at least for small classifiers. We selected small bags not only for time considerations: if there were to be some particular interest in a fine selection of the BCS, it would ideally already happen for small bags.

All results in Figure 15 advocate for the superiority of $F^*$ against all other approaches. For example, when $l = 2, r = 10$, $F^*$ tops all algorithms for almost half the domains. Even when we replace the .1 threshold probability by a .01 threshold probability (see Figure 15), $F^*$ still beats 7 algorithms. An interesting phenomenon happens for small classifiers: permissible functions with stronger concave regimes (e.g. (28)) tend to improve performances. While it was previously remarked for decision tree induction in [23], it is actually predicted up to some extent by Theorem 2, as the bound on $J$ is inversely proportional to the minimum of the second derivative of $\tilde{\psi}$. This phenomenon becomes (predictibly) dampened as classifiers become large, but we ultimately cannot conclude that (29) beats (28) and / or AdaBoost, according to Student paired t-test ($l = 3, r = 100$).

This makes the SCL derived from (28) a very interesting alternative to the logistic loss and AdaBoost, which might be useful in other supervised learning schemes as well. Finally, mixing permissible functions with different gradient images ($E_1$, $F_1$) is clearly a bad choice, but $F^*$ and $E_2$ are advocacies for further works on mixed fittings of SCS and classifiers. This is confirmed by a close look at the domains: for almost each algorithm and each choice of $(l, r)$, there exists a domain on which it ranks first, and one on which it ranks last. This is all the more important as previous works highlight the role of early stopping in consistency for convex surrogates [6].

## Acknowledgments

Support is acknowledged by ANR *Blanc* project ANR-07-BLAN-0328-01 "Computational Information Geometry and Applications". The second author acknowledge support from DIGITEO GAS: Geometric Algorithms for Statistics.

## References


1. Amari, S.-I.: Natural Gradient works efficiently in Learning. Neural Computation 10, 251–276 (1998)
2. Azran, A., Meir, R.: Data dependent risk bounds for hierarchical mixture of experts classifiers. In: Shawe-Taylor, J., Singer, Y. (eds.) COLT 2004. LNCS, vol. 3120, pp. 427–441. Springer, Heidelberg (2004)
3. Banerjee, A., Guo, X., Wang, H.: On the optimality of conditional expectation as a bregman predictor. IEEE Trans. on Information Theory 51, 2664–2669 (2005)
4. Banerjee, A., Merugu, S., Dhillon, I., Ghosh, J.: Clustering with Bregman divergences. Journal of Machine Learning Research 6, 1705–1749 (2005)
5. Bartlett, P., Jordan, M., McAuliffe, J.D.: Convexity, classification, and risk bounds. Journal of the Am. Stat. Assoc. 101, 138–156 (2006)
6. Bartlett, P., Traskin, M.: Adaboost is consistent. In: NIPS*19 (2006)
7. Blake, C.L., Keogh, E., Merz, C.J.: UCI repository of machine learning databases (1998), `http://www.ics.uci.edu/~mlearn/MLRepository.html`
8. Bregman, L.M.: The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. USSR Comp. Math. and Math. Phys. 7, 200–217 (1967)
9. Breiman, L., Freidman, J.H., Olshen, R.A., Stone, C.J.: Classification and regression trees. Wadsworth (1984)
10. Collins, M., Schapire, R., Singer, Y.: Logistic regression, adaboost and Bregman distances. In: COLT 2000, pp. 158–169 (2000)
11. Davis, J., Kulis, B., Jain, P., Sra, S., Dhillon, I.: Information-theoretic metric learning. In: ICML 2007 (2007)
12. Dhillon, I., Sra, S.: Generalized non-negative matrix approximations with Bregman divergences. In: Advances in Neural Information Processing Systems, vol. 18 (2005)
13. Freund, Y., Schapire, R.E.: A Decision-Theoretic generalization of on-line learning and an application to Boosting. Journal of Comp. Syst. Sci. 55, 119–139 (1997)
14. Friedman, J., Hastie, T., Tibshirani, R.: Additive Logistic Regression: a Statistical View of Boosting. Ann. of Stat. 28, 337–374 (2000)
15. Gates, G.W.: The Reduced Nearest Neighbor rule. IEEE Trans. on Information Theory 18, 431–433 (1972)
16. Gentile, C., Warmuth, M.: Linear hinge loss and average margin. In: NIPS*11, pp. 225–231 (1998)
17. Gentile, C., Warmuth, M.: Proving relative loss bounds for on-line learning algorithms using Bregman divergences. In: Tutorials of the 13 $^{th}$ International Conference on Computational Learning Theory (2000)
18. Grünwald, P., Dawid, P.: Game theory, maximum entropy, minimum discrepancy and robust Bayesian decision theory. Ann. of Statistics 32, 1367–1433 (2004)
19. Henry, C., Nock, R., Nielsen, F.: $\mathbb{I}$Real boosting *a la Carte* with an application to boosting Oblique Decision Trees. In: Proc. of the 21 $^{st}$ International Joint Conference on Artificial Intelligence, pp. 842–847 (2007)
20. Herbster, M., Warmuth, M.: Tracking the best regressor. In: COLT 1998, pp. 24–31 (1998)
21. Kearns, M.J., Vazirani, U.V.: An Introduction to Computational Learning Theory. MIT Press, Cambridge (1994)
22. Kearns, M.J.: Thoughts on hypothesis boosting, ML class project (1988)
23. Kearns, M.J., Mansour, Y.: On the boosting ability of top-down decision tree learning algorithms. Journal of Comp. Syst. Sci. 58, 109–128 (1999)

24. Kearns, M.J., Valiant, L.: Cryptographic limitations on learning boolean formulae and finite automata. In: Proc. of the 21 $^{th}$ ACM Symposium on the Theory of Computing, pp. 433–444 (1989)
25. Kivinen, J., Warmuth, M., Hassibi, B.: The $p$-norm generalization of the LMS algorithm for adaptive filtering. IEEE Trans. on Signal Processing 54, 1782–1793 (2006)
26. Kohavi, R.: The power of Decision Tables. In: Proc. of the 10 $^{th}$ European Conference on Machine Learning, pp. 174–189 (1995)
27. Matsushita, K.: Decision rule, based on distance, for the classification problem. Ann. of the Inst. for Stat. Math. 8, 67–77 (1956)
28. Mitchell, T.M.: The need for biases in learning generalization. Technical Report CBM-TR-117, Rutgers University (1980)
29. Murata, N., Takenouchi, T., Kanamori, T., Eguchi, S.: Information geometry of $\mathcal{U}$-Boost and Bregman divergence. Neural Computation, 1437–1481 (2004)
30. Nielsen, F., Boissonnat, J.-D., Nock, R.: On Bregman Voronoi diagrams. In: Proc. of the 19 $^{th}$ ACM-SIAM Symposium on Discrete Algorithms, pp. 746–755 (2007)
31. Nielsen, F., Boissonnat, J.-D., Nock, R.: Bregman Voronoi Diagrams: properties, algorithms and applications, 45 p. (submission, 2008)
32. Nock, R.: Inducing interpretable Voting classifiers without trading accuracy for simplicity: theoretical results, approximation algorithms, and experiments. Journal of Artificial Intelligence Research 17, 137–170 (2002)
33. Nock, R., Nielsen, F.: A $\mathbb{R}$eal Generalization of discrete AdaBoost. Artif. Intell. 171, 25–41 (2007)
34. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann, San Francisco (1993)
35. Schapire, R.E., Freund, Y., Bartlett, P., Lee, W.S.: Boosting the margin: a new explanation for the effectiveness of voting methods. Annals of statistics 26, 1651–1686 (1998)
36. Schapire, R.E., Singer, Y.: Improved boosting algorithms using confidence-rated predictions. Machine Learning Journal 37, 297–336 (1999)
37. Valiant, L.G.: A theory of the learnable. Communications of the ACM 27, 1134–1142 (1984)
38. Vapnik, V.: Statistical Learning Theory. John Wiley, Chichester (1998)
39. Warmuth, M., Liao, J., Rätsch, G.: Totally corrective boosting algorithms that maximize the margin. In: ICML 2006, pp. 1001–1008 (2006)