# Adaptive and Self-Confident On-Line Learning Algorithms

**Peter Auer**
University of Technology, Graz
Institute for Theoretical Computer Science
Klosterwiesgasse 32/2
A-8010 Graz, AUSTRIA.
pauer@igi.tu-graz.ac.at

**Claudio Gentile**
DSI, Università di Milano,
Via Comelico 39,
20135 Milano, Italy.
gentile@dsi.unimi.it

## Abstract

Most of the performance bounds for on-line learning algorithms are proven assuming a constant learning rate. To optimize these bounds, the learning rate must be tuned based on quantities that are generally unknown, as they depend on the whole sequence of examples. In this paper we show that essentially the same optimized bounds can be obtained when the algorithms adaptively tune their learning rates as the examples in the sequence are progressively revealed. Our adaptive learning rates apply to a wide class of on-line algorithms, including $p$-norm algorithms for generalized linear regression and Weighted Majority for linear regression with absolute loss.

We emphasize that our adaptive tunings are radically different from previous techniques, such as the so-called doubling trick. Whereas the doubling trick restarts the on-line algorithm several times using a constant learning rate for each run, our methods save information by changing the value of the learning rate very smoothly. In fact, for Weighted Majority over a finite set of experts our analysis provides a better leading constant than the doubling trick.

## 1 Introduction

In this paper we analyze on-line learning algorithms that tune their learning rate in an on-line fashion. In this introduction we will use the (randomized) Weighted Majority algorithm [Lit89, LW94, Vov90, Vov98, CBFH$^+$97] as a motivating example to describe the problem we are interested in. In the main part of the paper we will apply our techniques to the much more general class of *quasi-additive algorithms* [GLS97, KW98b].

### 1.1 An illustrating example

The Weighted Majority algorithm processes the examples one at a time in trials. In each trial $t$ the algorithm receives a binary vector $\boldsymbol{x}_t \in \{-1,+1\}^n$ and is asked to predict the value of an unknown binary label $y_t \in \{-1,+1\}$ associated with $\boldsymbol{x}_t$. The algorithm keeps a weight vector $\boldsymbol{w}_t$ representing its current hypothesis. The vector $\boldsymbol{w}_t$ is an $n$-dimensional probability vector. The algorithm's prediction at time $t$ is the linear combination $\hat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t \in [-1,+1]$. After receiving the correct label $y_t$, the algorithm incurs a loss $l_t = \frac{1}{2}|y_t - \hat{y}_t| \in [0,1]$. Finally, the algorithm updates the weights according to the rule

$$w_{t+1,i} = w_{t,i} \exp\{-\eta\,|y_t - x_{t,i}|\}/W_t,$$

where $\eta$ is the *learning rate* and $W_t$ is the normalizing factor making $\boldsymbol{w}_{t+1}$ a probability vector. Now, let $L_{i,T} = |\{1 \le t \le T : x_{t,i} \neq y_t\}|$ be the number of times the $i$th input component disagreed with $y_t$ on a sequence of $T$ trials (i.e., the number of "mistakes" of the $i$th component). The analysis of the Weighted Majority algorithm with fixed $\eta$ shows that, up to lower-order terms, the cumulative loss $L_T = \sum_{t=1}^{T} l_t$ can be upper bounded as

$$L_T \le (1+\eta)\, L_T^* + \tfrac{1+\eta}{\eta}\, c\, \ln n,$$

where $L_T^* = \min_{1\le i\le n} L_{i,T}$ and $c$ is a suitable positive constant. To obtain an optimal bound (up to lower-order terms) the learning rate $\eta$ has to be chosen with respect to $L_T^*$. For instance, the bound above is optimized by $\eta = \sqrt{c(\ln n)/L_T^*}$ which yields

$$L_T \le L_T^* + 2\sqrt{L_T^*\, c\, \ln n} + c\, \ln n.$$

Thus, according to the last bound, the loss of the algorithm is asymptotically the same as the loss of the best fixed component, if we disregard lower-order terms. Obviously this tuning needs *a priori* knowledge about the optimal number of mistakes $L_T^*$, which is usually not available. To solve this tuning problem, we will use an adaptive learning rate $\eta_t$ that varies over time, depending on the information the algorithm gains about $L_T^*$ during the learning process.

### 1.2 Incremental update of the learning rate versus the doubling trick

A standard method to deal with the tuning problem is the so-called "doubling trick": An upper bound $B$ on $L_T^*$ is assumed and the learning rate is tuned with respect to this bound as suggested by the previous section, i.e., $\eta = \sqrt{(c\,\ln n)/B}$. We call "round" a sequence of trials where $B$ is constant. If during the learning process the loss of the best component exceeds bound $B$, then this bound is increased, the learning algorithm is restarted, and a new round

begins. A sophisticated analysis of the doubling trick for the Weighted Majority algorithm can be found in [CBFH+97]. We may say that the doubling trick makes an on-line algorithm coarsely adaptive, as the learning rate is constant within a round and makes big jumps between rounds. However, a major disadvantage is that the on-line algorithm is restarted from scratch at the beginning of each round, hence losing all the information collected during the past rounds.

More disadvantages arise if the learning setting is made more general. For instance, in generalized linear regression [KW98b] checking the termination condition for the current round might be computationally expensive. Furthermore, it is not clear how the doubling trick could be applied when the loss on each trial can be arbitrarily large. In contrast, this paper analyzes on-line learning algorithms that are "incrementally adaptive", as they modify their learning rates possibly in each trial, and typically by a small amount. Via our approach we design on-line algorithms whose performance bounds are in some cases better, and never significantly worse, than those proven for the doubling trick. Moreover, some of our techniques are efficiently applicable to very general learning settings and can even handle unbounded loss functions. Finally, we expect that our algorithms behave better in practice than those based on the doubling trick.

As a remark, we note that algorithms with an incrementally adaptive learning rate were also proposed by Vovk [Vov97] and Azoury and Warmuth [AW99] for the problem of on-line linear regression with square loss.[1] However, their proof techniques are different from ours and do not seem easily extendible to more general regression problems.

### 1.3 The formal learning model

In this section we describe the learning model more precisely and give our basic notation. An *example* is a pair $(\boldsymbol{x}, y)$, where $\boldsymbol{x} \in \mathcal{R}^n$ is called an *instance*, and $y \in \mathcal{R}$ is the *label* associated with $\boldsymbol{x}$. On-line learning proceeds in trials. In the $t$-th trial the on-line algorithm receives an instance $\boldsymbol{x}_t$ and is required to give a *prediction* $\hat{y}_t$ about the unknown label $y_t$ associated with $\boldsymbol{x}_t$. Then $y_t$ is revealed and the algorithm incurs a loss $L(y_t, \hat{y}_t)$, measuring the discrepancy between the prediction $\hat{y}_t$ and the label $y_t$. We call a sequence $S = ((\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \dots)$ of instances and labels processed by the algorithm in a run a *trial sequence*.

We adopt a well-established mathematical model to analyze these algorithms. It is a generalization of a learning model introduced by Littlestone and Warmuth [Lit88, Lit89, LW94] and Angluin [Ang88]. We are given a *comparison class* of predictors and a loss function $L$. Broadly speaking, the goal of $A$ is to learn on the fly the best off-line predictor in the comparison class for the whole sequence $S$. Formally, we measure the performance of $A$ on $S$ by the cumulative loss $L_{A,T}(S)$ the algorithm $A$ suffers on $S$: $L_{A,T}(S) = \sum_{t=1}^{T} L(y_t, \hat{y}_t)$. We compare this loss to the loss of the comparison class, i.e., to the loss of the best predictor in the comparison class for the same trial sequence $S$.

This paper focuses on the linear regression problem with various losses, mainly the square loss and the absolute loss. Such problems have been widely investigated in the last years (see, e.g., Littlestone [Lit89, Lit91], Vovk [Vov90, Vov97, Vov98, Vov99], Littlestone and Warmuth [LW94], Cesa-Bianchi et al. [CBFH+97, CBLW96, CB99], Kivinen and Warmuth [KW97, KW98b], Yamanishi [Yam98], Grove et al. [GLS97], Gentile and Littlestone [GL99], Azoury and Warmuth [AW99], and references therein).

In linear regression the learner's hypothesis at time $t$ is represented by a weight vector $\boldsymbol{w}_t \in \mathcal{R}^n$, and the prediction $\hat{y}_t$ is often a function of $\boldsymbol{w}_t \cdot \boldsymbol{x}_t$. For instance, if $L$ is the square loss $L(y_t, \hat{y}_t) = \frac{1}{2}(y_t - \hat{y}_t)^2$ and the prediction is just $\hat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$, then we compare the cumulative loss $L_{A,T}(S)$ of the algorithm with the least cumulative square loss that could be incurred by predicting with a fixed weight vector $\boldsymbol{u}$ in the comparison class. In particular, setting $L_{\boldsymbol{u},T}(S) = \sum_{t=1}^{T} L(y_t, \boldsymbol{u} \cdot \boldsymbol{x}_t)$, our goal is to bound the loss difference (this is also called the *regret* or *relative loss*) $L_{A,T}(S) - \min_{\boldsymbol{u}} L_{\boldsymbol{u},T}(S)$, for an arbitrary trial sequence $S$.

In the absolute loss setting we have $L(y_t, \hat{y}_t) = \frac{1}{2}|y_t - \hat{y}_t|$. In this paper we only consider the case when the labels $y_t$ have range $[-1, +1]$. The prediction $\hat{y}_t$ is a suitable clipping function of $\boldsymbol{w}_t \cdot \boldsymbol{x}_t$, whose range is again $[-1, +1]$. As a consequence, $L(y_t, \hat{y}_t) \in [0, 1]$. We are still aimed at bounding the cumulative (absolute) loss of $A$, but the way we measure the performance of the best off-line $\boldsymbol{u}$ might be different (see Section 3.1). The case $y_t \in \{-1, +1\}$ is of special interest, since it can be interpreted as a binary classification problem where the algorithm is allowed to make randomized predictions. The absolute loss $\frac{1}{2}|y_t - \hat{y}_t|$ is then the probability of a prediction mistake, i.e., the probability that $y_t \neq \hat{y}_t$, and the cumulative loss is just the expected number of mistakes. For instance, the Weighted Majority algorithm of Section 1.1 can be seen as an algorithm for the following restricted class of regression problems: the loss function is the absolute loss $L(y_t, \hat{y}_t) = \frac{1}{2}|y_t - \hat{y}_t|$, the prediction is $\hat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$, and the comparison class is the set of the $n$ unit vectors.

### 1.4 Incrementally adaptive on-line learning algorithms

In this section we distinguish two ways how we will tune the learning rates of the algorithms. The more obvious way is to set the learning rate $\eta_t$ with respect to the loss of the comparison class observed so far. For the Weighted Majority algorithm of Section 1.1 this would be $\eta_t = \sqrt{(c \ln n)/L_t^*}$, where $L_t^* = \min_{1 \le i \le n} \sum_{\tau=1}^{t} \frac{1}{2}|y_\tau - x_{\tau,i}|$. In Section 2 we analyze this choice of the learning rate for the Weighted Majority algorithm.

An alternative way for tuning the learning rate $\eta_t$ is to use the loss of the learning algorithm to calculate $\eta_t$. This is done under the assumption that the current cumulative loss of the algorithm closely matches the current cumulative loss of the comparison class. We call such algorithms *self-confident*, as they "trust themselves" in tuning $\eta_t$. There are two reasons to use a self-confident tuning instead of a tuning based on the true current loss of the comparison class. First, as we said in Section 1.2, the evaluation of the loss of the comparison class might be computationally hard: storing all past examples and resorting to numerical methods might be necessary for loss functions more difficult than the square loss. Second, the analysis of self-confident algorithms seems to be much simpler in most cases. In fact, we have been able to

[1] The learning rate in those papers is actually a covariance matrix.

analyze the tuning based on the current loss of the comparison class only for the relatively simple Weighted Majority algorithm. This is done in Section 2. Self-confident learning algorithms will be analyzed in Sections 3.1 and 3.2.

## 2 An incrementally adaptive Weighted Majority algorithm

In this section we present an incrementally adaptive version of the Weighted Majority algorithm for prediction with absolute loss. To keep the analysis simple, we consider the randomized classification case. That is, we assume that $x_{t,i}, y_t \in \{-1, +1\}$. However, the main result of this section, Theorem 1, holds with minor modifications also for the regression case, where $x_{t,i}, y_t \in [-1, +1]$. Recall the notation of Section 1.1. The construction of the algorithm, see Figure 1, is quite straightforward from the original Weighted

**For** $t = 1, 2, \ldots, T$ :

1. Calculate the weights as
$$\begin{aligned} w_{t,i} &= \alpha_t^{-L_{i,t-1}}/W_t, \\ \epsilon_t &= \min\left\{\frac{1}{4}, \sqrt{2\frac{\ln n}{L^*_{t-1}}}\right\}, \\ \alpha_t &= \frac{1}{1-\epsilon_t}, \\ W_t &= \sum_{i=1}^n \alpha_t^{-L_{i,t-1}}; \end{aligned}$$
2. Get instance $\boldsymbol{x}_t$ and predict with $\hat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$;
3. Get label $y_t$ and incur loss $\frac{1}{2}|y_t - \hat{y}_t|$.

Figure 1: The incrementally adaptive Weighted Majority algorithm

Majority algorithm [Lit89, LW94, Vov90, CBFH+97]. Essentially, the only modification necessary is the introduction of a varying learning rate. (In this section we found it more convenient to set $\eta_t = \ln \alpha_t$ and focus on the tuning of $\alpha_t$.) There is a subtle point here, though, since the learning rate is changed also retrospectively: In step 1 of the algorithm the weight $w_{t,i}$ is set proportionally to $\alpha_t^{-L_{i,t-1}}$, whereas one might expect that it is set to $w_{t-1,i}\alpha_t^{-|y_t - x_{t,i}|}$. Setting the weight to $\alpha_t^{-L_{i,t-1}}$ actually means that the new learning rate parameter $\alpha_t$ is applied to all past trials. This is quite essential for the analysis. Notice that in Figure 1 we have $L_{i,0} = L_0^* = 0$. Thus $w_{1,i} = 1/n$ and $\epsilon_1 = 1/4$. For this algorithm we have the following result.

**Theorem 1** *Let $S = (\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots$, where $(\boldsymbol{x}_t, y_t) \in \{-1, +1\}^n \times \{-1, +1\}$. Then the adaptive Weighted Majority algorithm in Figure 1, run on a prefix of $S$ of arbitrary length $T$, achieves the following cumulative absolute loss bound:*

$$\begin{aligned} L_T &\le L_T^* + 2\sqrt{2L_T^* \ln n} \\ &\quad + \frac{9}{2}(\ln n)(\ln(1 + L_T^*)) + 6\ln n + 1, \end{aligned}$$

*where*

$$L_T^* = \min_{1 \le i \le n} \sum_{t=1}^T \tfrac{1}{2}|y_t - x_{t,i}|. \ \square$$

This result improves on results obtained by the doubling trick [CBFH+97], where it was shown that $L_T \le L_T^* + (3.3302 + o(1))\sqrt{L_T^* \ln n}$ as $T \to \infty$. Our result gives $L_T \le L_T^* + (2.83 + o(1))\sqrt{L_T^* \ln n}$ as $T \to \infty$.

### 2.1 Analysis of the algorithm

The analysis proceeds in the usual way by bounding the corresponding Bregman divergences $d(\cdot, \cdot)$ (see Section 3). In the scenario of the Weighted Majority algorithm the comparison class consists of the unit vectors $\boldsymbol{u}_i$, $i = 1, \ldots, n$, and the Bregman divergence is just the relative entropy. Thus $d(\boldsymbol{u}_i, \boldsymbol{w}_t) = -\ln w_{t,i}$. For the analysis we will bound

$$\frac{\ln w_{t+1,i^*_{t+1}}}{\epsilon_{t+1}} - \frac{\ln w_{t,i^*_t}}{\epsilon_t}$$

where $i_t^* = \arg\max_i w_{t,i}$ is the component with the minimal number of mistakes so far. Set $W'_t = \sum_{i=1}^n \alpha_t^{-L_{i,t}}$ so that $w'_{t,i} = \alpha_t^{-L_{i,t}}/W'_t$ would be the weights for the next trial if the learning rate were not changed. We get

$$\begin{aligned} &\frac{\ln w_{t+1,i^*_{t+1}}}{\epsilon_{t+1}} - \frac{\ln w_{t,i^*_t}}{\epsilon_t} \\ &= \ln w_{t+1,i^*_{t+1}} \left(\frac{1}{\epsilon_{t+1}} - \frac{1}{\epsilon_t}\right) \\ &\quad + \frac{\ln w_{t+1,i^*_{t+1}} - \ln w'_{t,i^*_{t+1}}}{\epsilon_t} + \frac{\ln w'_{t,i^*_{t+1}} - \ln w_{t,i^*_t}}{\epsilon_t} \\ &\ge -\ln n \left(\frac{1}{\epsilon_{t+1}} - \frac{1}{\epsilon_t}\right) \\ &\quad + \frac{1}{\epsilon_t} \ln \frac{\alpha_{t+1}^{-L_t^*} W'_t}{\alpha_t^{-L_t^*} W_{t+1}} + \frac{1}{\epsilon_t} \ln \frac{\alpha_t^{-L_t^*}}{\alpha_t^{-L^*_{t-1}}} + \frac{1}{\epsilon_t} \ln \frac{W_t}{W'_t}. \end{aligned} \tag{1}$$

We denote the three logarithmic factors in the right-most side of (1) as

$$B_{1,t} = \ln \frac{\alpha_{t+1}^{-L_t^*} W'_t}{\alpha_t^{-L_t^*} W_{t+1}}, \ B_{2,t} = \ln \frac{\alpha_t^{-L_t^*}}{\alpha_t^{-L^*_{t-1}}}, \ B_{3,t} = \ln \frac{W_t}{W'_t}.$$

We proceed by lower bounding $B_{1,t}$, $B_{2,t}$ and $B_{3,t}$. To bound $B_{1,t}$ we use the following technical lemma whose proof is given in the appendix.

**Lemma 2** *For $1 < \alpha \le \beta$ and any $\ell_1, \ldots, \ell_{n-1} \ge 0$ it holds that*

$$\ln\left(\frac{1 + \sum_{i=1}^{n-1} \beta^{-\ell_i}}{1 + \sum_{i=1}^{n-1} \alpha^{-\ell_i}}\right) \ge -(\beta - \alpha)\frac{\ln n}{\ln \alpha}.$$

Thus

$$B_{1,t} \geq -(\alpha_t - \alpha_{t+1}) \frac{\ln n}{\ln \alpha_{t+1}}. \ \square$$

We assume that $\alpha_t \neq \alpha_{t+1}$ and $\epsilon_t < 1/4$ (there is at most one trial with $\alpha_t \neq \alpha_{t+1}$ and $\epsilon_t = 1/4$). Then $L_t^* = L_{t-1}^* + 1$, and since $\alpha_t$ as a function of $L_{t-1}^*$ is convex we get

$$\alpha_t - \alpha_{t+1} \leq \frac{\partial}{\partial L_{t-1}^*} \alpha_t = \alpha_t^2 \sqrt{\frac{\ln n}{2(L_{t-1}^*)^3}} = \alpha_t^2 \frac{\epsilon_t^3}{4 \ln n}.$$

Analogously we get

$$\frac{1}{\ln \alpha_{t+1}} \leq \frac{1}{\epsilon_{t+1}} \leq \frac{1}{\epsilon_t} + \frac{1}{\epsilon_t^2} \sqrt{\frac{\ln n}{2(L_{t-1}^*)^3}} = \frac{1}{\epsilon_t}(1 + \frac{\epsilon_t^2}{4 \ln n}).$$

Thus

$$B_{1,t} \geq -\frac{\alpha_t^2 \epsilon_t^2}{4}(1 + \frac{\epsilon_t^2}{4 \ln n}) \geq -\frac{\epsilon_t^2}{4}(1 + 5\epsilon_t)$$

for trials $t$ with $\alpha_t \neq \alpha_{t+1}$ and $\epsilon_t < 1/4$. For trials with $\alpha_t = \alpha_{t+1}$ we have $B_{1,t} = 0$.

For the next bound we use $\ln(1-\epsilon) \geq -\epsilon - \epsilon^2/2 - \epsilon^3$ for $\epsilon \leq 1/4$:

$$B_{2,t} = \ln \frac{\alpha_t^{-L_t^*}}{\alpha_t^{-L_{t-1}^*}} \geq -\epsilon_t (L_t^* - L_{t-1}^*)(1 + \epsilon_t/2 + \epsilon_t^2).$$

Finally, $B_{3,t}$ is bounded in the usual way,

$$\begin{aligned}
B_{3,t} &= \ln \frac{W_t}{W_t'} \\
&= -\ln \left( \sum_{i=1}^n w_{t,i} \alpha_t^{-|y_t - x_{t,i}|/2} \right) \\
&= -\ln \left( 1 + \sum_{i=1}^n w_{t,i} \left( \alpha_t^{-|y_t - x_{t,i}|/2} - 1 \right) \right) \\
&\geq -\sum_{i=1}^n w_{t,i} \left( \alpha_t^{-|y_t - x_{t,i}|/2} - 1 \right) \\
&= \epsilon_t \sum_{i=1}^n w_{t,i} |y_t - x_{t,i}|/2 \\
&\geq \epsilon_t |y_t - \sum_{i=1}^n w_{t,i} x_{t,i}|/2 \\
&= \epsilon_t (L_t - L_{t-1}).
\end{aligned}$$

Using the bounds on $B_{1,t}$, $B_{2,t}$, $B_{3,t}$ in (1) and summing for $t = 1, \ldots, T$ gets

$$\begin{aligned}
4 \ln n &\geq \frac{\ln w_{T+1,i_{T+1}}}{\epsilon_{T+1}} - \frac{\ln w_{1,1}}{\epsilon_1} \\
&\geq -\ln n \left( \frac{1}{\epsilon_{T+1}} - \frac{1}{\epsilon_1} \right) \\
&\quad -1 - \sum_{t: \epsilon_{t+1} \neq \epsilon_t, \epsilon_t < 1/4} (1 + 5\epsilon_t) \frac{\epsilon_t}{4} \\
&\quad -L_T^* - \sum_{t=1}^T (\epsilon_t/2 + \epsilon_t^2)(L_t^* - L_{t-1}^*) \\
&\quad +L_T \\
&\geq -L_T^* + L_T - \frac{\ln n}{\epsilon_{T+1}} + 4 \ln n - 1 \\
&\quad -\frac{3}{4} \sum_{t: \epsilon_{t+1} \neq \epsilon_t, \epsilon_t < 1/4} (\epsilon_t + 3\epsilon_t^2) \\
&\quad - \sum_{t: L_{t+1}^* \neq L_t^*, \epsilon_t = 1/4} (\epsilon_t/2 + \epsilon_t^2) \\
&\geq -L_T^* + L_T - \sqrt{L_T^* (\ln n)/2} + 4 \ln n - 1 \\
&\quad -\frac{3}{4} \int_{32 \ln n - 1}^{L_T^*} \left( \sqrt{\frac{2 \ln n}{L}} + \frac{6 \ln n}{L} \right) dL \\
&\quad -32 \ln n (\frac{1}{8} + \frac{1}{16}) \\
&\geq -L_T^* + L_T - \sqrt{L_T^* (\ln n)/2} - 2 \ln n - 1 \\
&\quad -\frac{3}{2} \sqrt{2 L_T^* \ln n} - \frac{9}{2} (\ln n)(\ln(1 + L_T^*)) \\
&\geq -L_T^* + L_T - 2\sqrt{2 L_T^* (\ln n)} - 2 \ln n - 1 \\
&\quad -\frac{9}{2} (\ln n)(\ln(1 + L_T^*)).
\end{aligned}$$

This proves Theorem 1.

## 3 Quasi-additive learning algorithms

This section describes a general class of algorithms we will deal with in the remainder of the paper. This class of algorithms was introduced by Grove et al. [GLS97] in the context of binary classification and also, independently, by Kivinen and Warmuth [KW98b] in the context of (generalized) linear regression. These algorithms are called *quasi-additive* in [GLS97] and *general additive* in [KW98b]. This class of algorithms covers a wide variety of learning algorithms. For instance, in the binary classification setting this class includes Perceptron [Ros62, Blo62, Nov62] and algorithms in the Winnow family [Lit88, Lit89], such as Weighted Majority; in the regression setting this class includes the Widrow-Hoff rule [WH60] and algorithms in the EG family [KW97]. The general additive algorithms also include the subfamily of the $p$-norm algorithms [GLS97, GL99], which are treated in Section 3.1.

All these algorithms have the same basic structure. In the generic trial $t$ the algorithm stores the weight vector $\boldsymbol{w}_t$,

lying in a suitable weight space. Combined with the current instance $\boldsymbol{x}_t$, the vector $\boldsymbol{w}_t$ determines the algorithm's prediction $\hat{y}_t$, which is a function of $\boldsymbol{w}_t \cdot \boldsymbol{x}_t$. Then, based on the label $y_t$, the algorithm performs the weight update step $\boldsymbol{w}_t \rightarrow \boldsymbol{w}_{t+1}$. At the core of the weight update lies the rule $\boldsymbol{w}_{t+1} = \mathbf{f}^{-1}(\mathbf{f}(\boldsymbol{w}_t) + \eta_t\, g(y_t, \hat{y}_t)\, \boldsymbol{x}_t)$, where $\mathbf{f}$ is a smooth bijective mapping from the adopted weight space to $\mathcal{R}^n$, $\mathbf{f}^{-1}$ is the inverse of $\mathbf{f}$ and $g$ is a suitable function of $y_t$ and $\hat{y}_t$. For instance, in linear regression with square loss we have $g(y_t, \hat{y}_t) = y_t - \hat{y}_t$. The Widrow-Hoff rule is then obtained when $\mathbf{f}$ is the identity mapping, while the EGU algorithm [KW97] is given by the componentwise logarithm $\mathbf{f}(\boldsymbol{w}_t) = (\ln w_{t,1}, \ldots, \ln w_{t,n})$.

The standard way to analyze these algorithms (e.g., [Lit88, Lit89, LW94, HKW95, AW98, CBLW96, CBFH+97, GLS97, KW97, Byl97, GW98, AW99, GL99]) is to define a measure of progress related to the mapping $\mathbf{f}$. The measure of progress we use here is the so-called *Bregman divergence* [Bre67, CL81] associated with $\mathbf{f}$. We denote the divergence by $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w})$, where $\boldsymbol{u}$ and $\boldsymbol{w}$ are weight vectors. Informally, we can define $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w})$ as follows. Assume that $\mathbf{f}$ is the gradient of some convex function $P_{\mathbf{f}}$ on a convex weight space. Then $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w})$ is the difference between $P_{\mathbf{f}}(\boldsymbol{u})$ and the first-order Taylor expansion of $P_{\mathbf{f}}$ around $\boldsymbol{w}$. (Thus by convexity $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}) \geq 0$.) For instance, if $\mathbf{f}$ is the identity then $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}) = \frac{1}{2}||\boldsymbol{u} - \boldsymbol{w}||_2^2$, whereas if $\mathbf{f}$ is the componentwise logarithm then $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w})$ is the (unnormalized) relative entropy divergence: $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}) = \sum_{i=1}^n (u_i \ln \frac{u_i}{w_i} + w_i - u_i)$. A further example of Bregman divergence is provided in Section 3.1, where we analyze a self-confident version of the $p$-norm algorithms. In Section 3.2 we discuss how to extend this analysis to algorithms in the Winnow-EG family.

### 3.1 Self-confident $p$-norm algorithms

This section defines and analyzes our self-confident $p$-norm algoritms. We deal with two linear regression settings: 1) the square loss setting; 2) the absolute loss setting with binary labels (i.e., the binary classification problem where the algorithm makes randomized predictions). Our results for square loss are easily extended to more general regression frameworks, such as Helmbold, Kivinen and Warmuth's [HKW95, KW98b] generalized linear regression model.

We first need to recall some preliminaries about the dual norms technology we will be using in this section. Given a vector $\boldsymbol{w} = (w_1, \ldots, w_n) \in \mathcal{R}^n$ and $p \geq 1$ we denote by $||\boldsymbol{w}||_p$ the $p$-norm of $\boldsymbol{w}$, i.e., $||\boldsymbol{w}||_p = \left(\sum_{i=1}^n |w_i|^p\right)^{1/p}$ (also, $||\boldsymbol{w}||_\infty = \lim_{p\to\infty} \left(\sum_{i=1}^n |w_i|^p\right)^{1/p} = \max_i |w_i|$). We say that $p$ and $q$ are *dual* if $\frac{1}{p} + \frac{1}{q} = 1$ holds. For example, the 1-norm is dual to the $\infty$-norm and the 2-norm is self-dual. For the rest of this section we assume that $p$ and $q$ are some pair of dual values with $p \geq 2$.

The $p$-norm algorithms [GLS97, GL99] are defined in terms of the following bijective mapping $\mathbf{f}$ (a $p$ indexing for $\mathbf{f}$ is understood): $\mathbf{f} : \mathcal{R}^n \rightarrow \mathcal{R}^n$, $\mathbf{f} = (f_1, \ldots, f_n)$, where

$$f_i(\boldsymbol{w}) = \frac{\mathrm{sign}(w_i)\, |w_i|^{q-1}}{||\boldsymbol{w}||_q^{q-2}}, \quad \boldsymbol{w} = (w_1, \ldots, w_n) \in \mathcal{R}^n. \tag{2}$$

The function $\mathbf{f}$ is just the gradient of $P_{\mathbf{f}}(\boldsymbol{w}) = \frac{1}{2}||\boldsymbol{w}||_q^2$. The inverse $\mathbf{f}^{-1}$ of $\mathbf{f}$ is given by [GL99] $\mathbf{f}^{-1} : \mathcal{R}^n \rightarrow \mathcal{R}^n$, $\mathbf{f}^{-1} = (f_1^{-1}, \ldots, f_n^{-1})$, where

$$f_i^{-1}(\boldsymbol{\theta}) = \frac{\mathrm{sign}(\theta_i)\, |\theta_i|^{p-1}}{||\boldsymbol{\theta}||_p^{p-2}}, \quad \boldsymbol{\theta} = (\theta_1, \ldots, \theta_n) \in \mathcal{R}^n,$$

i.e., $\mathbf{f}^{-1}$ is obtained from $\mathbf{f}$ by replacing $q$ with $p$. Notice that if $p = 2$ then $\mathbf{f}$ is the identity function.

The Bregman divergence associated with the gradient mapping $\mathbf{f}$ given in (2) is

$$d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}) = P_{\mathbf{f}}(\boldsymbol{u}) - P_{\mathbf{f}}(\boldsymbol{w}) - (\boldsymbol{u} - \boldsymbol{w}) \cdot \mathbf{f}(\boldsymbol{w}), \tag{3}$$

where $P_{\mathbf{f}}$ is as above. Observe that from the strict convexity of $P_{\mathbf{f}}$ it follows that $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}) \geq 0$ with equality holding if and only if $\boldsymbol{u} = \boldsymbol{w}$.

The Bregman divergence defined in (3) has also been used in [GL99]. It is easy to check [GL99] (see also Gordon [Go99]) that $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w})$ can be rewritten as

$$d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}) = \tfrac{1}{2}\,||\boldsymbol{u}||_q^2 + \tfrac{1}{2}\,||\boldsymbol{w}||_q^2 - \boldsymbol{u} \cdot \mathbf{f}(\boldsymbol{w}). \tag{4}$$

Also, notice that the special case $p = 2$ yields $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}) = \frac{1}{2}||\boldsymbol{u} - \boldsymbol{w}||_2^2$.

The following lemma is a Bregman divergence version of a classical result about projection operators. This lemma has also been used in the context of on-line learning by Herbster and Warmuth [HW98].

**Lemma 3** *[Bre67, CL81] Let $\mathcal{W} \subseteq \mathcal{R}^n$ be closed and convex, $\boldsymbol{w} \in \mathcal{R}^n$, and let $\boldsymbol{w}' = \mathrm{argmin}_{\boldsymbol{u}\in\mathcal{W}}\, d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w})$ be the* projection *of $\boldsymbol{w}$ onto $\mathcal{W}$ w.r.t. the Bregman divergence $d_{\mathbf{f}}$ given in (3). Then for any $\boldsymbol{u} \in \mathcal{W}$ the following holds:*

$$d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}) \geq d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}') + d_{\mathbf{f}}(\boldsymbol{w}', \boldsymbol{w}) \geq d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}').\ \square$$

In this section we take our comparison class to be the convex set $\mathcal{W}_U = \{\boldsymbol{w} \in \mathcal{R}^n : ||\boldsymbol{w}||_q \leq U\}$ and we will always be projecting onto $\mathcal{W}_U$. By a simple Kuhn-Tucker analysis it is not hard to verify that in such a case: $\boldsymbol{w}' = (\boldsymbol{w}U)/||\boldsymbol{w}||_q$ if $||\boldsymbol{w}||_q > U$ and $\boldsymbol{w}' = \boldsymbol{w}$, otherwise. (This specific projection occurs in the algorithms of Figures 2 and 3.) We also have the following lemma, whose proof is in the appendix.

**Lemma 4** *If $\boldsymbol{u}, \boldsymbol{w} \in \mathcal{W}_U$ then $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}) \leq 2\,U^2$.* $\square$

The $p$-norm algorithms are a versatile on-line learning tool. It is noticed in [GLS97, GL99] that by varying $p$ these algorithms can behave in a radically different manner. Consider, for instance, the case of the square loss. Here $p = 2$ yields the Widrow-Hoff rule, while $p = 2 \ln n$ gives rise to an algorithm which is very similar to EG.

We now describe the self-confident $p$-norm algorithms for square loss and absolute loss. Both algorithms are assumed to know [2] a bound $U$ on the $q$-norm of the comparison vector. This assumption could be removed by applying the results we mention in Section 3.2.

We observe that, unlike previous on-line regression analyses [KW97, HKW95, KW98b, GL99], our algorithms do

[2] It is worth noticing that, in order to make a constant learning rate algorithm achieve bounds of the form of Theorems 8 and 9, both the norm of the best $\boldsymbol{u}$ and its loss seem to be a necessary prior knowledge [KW97, AW98].

not have any prior knowledge about the norm of the instances.

The algorithms are given in Figures 2 and 3. In Figure 2 we denote by $L_t$ the cumulative square loss of the algorithm up to trial $t$, i.e., $L_t = \sum_{i=1}^{t} l_i$, where $l_t = \frac{1}{2}(y_t - \hat{y}_t)^2$. In Figure 3 we denote by $L_t$ the cumulative absolute loss of the algorithm up to trial $t$, i.e., $L_t = \sum_{i=1}^{t} l_i$, where $l_t = \frac{1}{2}|y_t - \hat{y}_t|$. Also, in both figures $k_t = (p-1)\, X_t^2\, U^2$, where $X_t = \max_{i\,:\, i \le t,\, l_i > 0} ||\boldsymbol{x}_i||_p$.

**Initialization:** Initial weight vector $\boldsymbol{w}_1 \in \mathcal{W}_U$.
**For** $t = 1, 2, \ldots, T$ :

- Get instance $\boldsymbol{x}_t \in \mathcal{R}^n$;
- Predict with $\hat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$;
- Get label $y_t \in \mathcal{R}$;
  Incur loss $l_t = \frac{1}{2}(y_t - \hat{y}_t)^2$;
- **If** $l_t > 0$ **then** update weights as follows:

$$\boldsymbol{w}_t^m = \mathbf{f}^{-1}(\mathbf{f}(\boldsymbol{w}_t) + \eta_t\,(y_t - \hat{y}_t)\,\boldsymbol{x}_t),$$

$$\boldsymbol{w}_{t+1} = \begin{cases} \boldsymbol{w}_t^m & \text{if } ||\boldsymbol{w}_t^m||_q \le U, \\ \frac{\boldsymbol{w}_t^m\, U}{||\boldsymbol{w}_t^m||_q} & \text{otherwise,} \end{cases}$$

where

$$\begin{aligned} \eta_t &= \frac{c_t}{1 + c_t}\,\frac{1}{(p-1)\, X_t^2}, \\ c_t &= \frac{\sqrt{k_t}}{\sqrt{k_t + L_t} - \sqrt{k_t}}, \\ k_t &= (p-1)\, X_t^2\, U^2, \\ X_t &= \max_{i\,:\, i \le t,\, l_i > 0} ||\boldsymbol{x}_i||_p. \end{aligned}$$

Figure 2: The self-confident $p$-norm algorithm for square loss.

The algorithms maintain an $n$-dimensional weight vector. They start from $\boldsymbol{w}_1 \in \mathcal{W}_U$, and in the generic trial $t$ they are required to predict the unknown label $y_t$ associated with the instance $\boldsymbol{x}_t$. The square loss algorithm predicts the label $y_t \in \mathcal{R}$ through the linear combination $\hat{y}_t = \boldsymbol{w}_t \cdot \boldsymbol{x}_t$, while the absolute loss algorithm predicts the label $y_t \in \{-1, +1\}$ through the clipped linear combination $\hat{y}_t = \sigma_{c_t}(\boldsymbol{w}_t \cdot \boldsymbol{x}_t)$, as specified in Figure 3. Notice that the knot $c_t$ of the clipping function $\sigma_{c_t}$ tends to get close to 1 as the cumulative absolute loss $L_t$ grows. When the label $y_t$ is received, the algorithms incur a loss $l_t$. As we already said, this loss is the square loss for the algorithm in Figure 2 and the absolute loss for the algorithm in Figure 3. Finally, the algorithms update their weights as indicated. In both figures the update has two steps. The first step computes $\boldsymbol{w}_t^m$ by the conventional update of the $p$-norm algorithms, as in [GL99]. The second step computes $\boldsymbol{w}_{t+1}$ by projecting $\boldsymbol{w}_t^m$ onto $\mathcal{W}_U$ w.r.t. $d_{\mathbf{f}}$. Notice that the algorithms do not update their weights if $l_t = 0$.[3]

[3]The algorithms do not update also in the degenerate case that $\boldsymbol{x}_t = \mathbf{0}$.

**Initialization:** Initial weight vector $\boldsymbol{w}_1 \in \mathcal{W}_U$.
**For** $t = 1, 2, \ldots, T$ :

- Get instance $\boldsymbol{x}_t \in \mathcal{R}^n$;
- let

$$\begin{aligned} \alpha_t &= \sqrt{\frac{k_t/4}{k_t/4 + L_{t-1} + 1}}, \\ k_t &= (p-1)\, X_t^2\, U^2, \\ X_t &= \max_{i\,:\, i \le t,\, l_i > 0} ||\boldsymbol{x}_i||_p, \\ c_t &= 1 - \alpha_t; \end{aligned}$$

- Predict with $\hat{y}_t = \sigma_{c_t}(\boldsymbol{w}_t \cdot \boldsymbol{x}_t)$

$$= \begin{cases} 1 & \text{if } \boldsymbol{w}_t \cdot \boldsymbol{x}_t \ge c_t, \\ \frac{\boldsymbol{w}_t \cdot \boldsymbol{x}_t}{c_t} & \text{if } \boldsymbol{w}_t \cdot \boldsymbol{x}_t \in (-c_t, c_t), \\ -1 & \text{if } \boldsymbol{w}_t \cdot \boldsymbol{x}_t \le -c_t; \end{cases}$$

- Get label $y_t \in \{-1, +1\}$;
  Incur loss $l_t = \frac{1}{2}|y_t - \hat{y}_t|$;
- **If** $l_t > 0$ **then** update weights as follows:

$$\boldsymbol{w}_t^m = \mathbf{f}^{-1}(\mathbf{f}(\boldsymbol{w}_t) + \eta_t\, y_t\, \boldsymbol{x}_t),$$

$$\boldsymbol{w}_{t+1} = \begin{cases} \boldsymbol{w}_t^m & \text{if } ||\boldsymbol{w}_t^m||_q \le U, \\ \frac{\boldsymbol{w}_t^m\, U}{||\boldsymbol{w}_t^m||_q} & \text{otherwise,} \end{cases}$$

where $\eta_t = \frac{2\,\alpha_t}{(p-1)\, X_t^2}$.

Figure 3: The self-confident $p$-norm algorithm for absolute loss and binary labels.

We need the following three technical lemmas. The first lemma is taken from [GL99], but it essentially follows from a combination of [KW97] and [GLS97]. The second lemma appears in various forms in [JW98, KW98b, GW98, GL99]. The third lemma is a simple technical tool for our self-confident analysis. Its proof is given in the appendix.

**Lemma 5** *Let $\boldsymbol{u}, \boldsymbol{w}_t \in \mathcal{R}^n$, $\boldsymbol{x}_t \in \mathcal{R}^n$, with $||\boldsymbol{x}_t||_p \le X_t$, and set $\boldsymbol{w}_t^m = \mathbf{f}^{-1}(\mathbf{f}(\boldsymbol{w}_t) + \eta_t\,(y_t - \hat{y}_t)\,\boldsymbol{x}_t)$, with $\eta_t = \frac{c_t}{(1+c_t)(p-1)\, X_t^2}$, $c_t \ge 0$, $X_t > 0$ and $y_t, \hat{y}_t \in \mathcal{R}$. Then the following inequality holds:* [4]

$$\frac{c_t}{1 + c_t}\,\tfrac{1}{2}(y_t - \hat{y}_t)^2 - c_t\,\tfrac{1}{2}(y_t - \boldsymbol{u} \cdot \boldsymbol{x}_t)^2 \le (p-1)\, X_t^2 (d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t^m)).\ \square$$

**Lemma 6** *Let $\boldsymbol{u}, \boldsymbol{w}_t, \boldsymbol{x}_t \in \mathcal{R}^n$, $\eta_t > 0$ and set $\boldsymbol{w}_t^m = \mathbf{f}^{-1}(\mathbf{f}(\boldsymbol{w}_t) + \eta_t\, y_t\, \boldsymbol{x}_t)$. Then the following equality holds:*

$$y_t\,(\boldsymbol{u} \cdot \boldsymbol{x}_t - \boldsymbol{w}_t \cdot \boldsymbol{x}_t) = \tfrac{1}{\eta_t}\,(d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t^m) + d_{\mathbf{f}}(\boldsymbol{w}_t, \boldsymbol{w}_t^m))\ .\ \square$$

[4]In the degenerate case that $X_t = 0$ we have $\boldsymbol{w}_t^m = \boldsymbol{w}_t$. In such a case $\eta_t$ is not defined, but the inequality of the lemma trivially holds true for any $c_t \ge 0$.

**Lemma 7** *Let $l_1, l_2, \ldots, l_T$ and $\delta$ be nonnegative real numbers. Then*

$$\sum_{t=1}^{T} \frac{l_t}{\sqrt{\delta + \sum_{i=1}^{t} l_i}} \leq 2 \left( \sqrt{\delta + \sum_{i=1}^{T} l_t} - \sqrt{\delta} \right),$$

*where* $\frac{0}{\sqrt{0}} = 0$. $\Box$

The bounds of Theorems 8 and 9 below have the same form of those proven for algorithms whose constant learning rate has been optimized in terms of the total loss of the comparison class (e.g., [KW97, KW98b, AW98]). We did not optimize the constants in our proofs. Notice that the bound for the absolute loss algorithm is in terms of the *deviation* $D(\boldsymbol{u}; (\boldsymbol{x}_t, y_t))$ of a linear threshold classifier $\boldsymbol{u}$ with threshold 0 on example $(\boldsymbol{x}_t, y_t)$, defined as $D(\boldsymbol{u}; (\boldsymbol{x}_t, y_t)) = \max\{0, 1 - y_t\, \boldsymbol{u} \cdot \boldsymbol{x}_t\}$. The quantity $\frac{1}{2} D(\boldsymbol{u}; (\boldsymbol{x}_t, y_t))$ is related to a "loss" of $\boldsymbol{u}$ on example $(\boldsymbol{x}_t, y_t)$. For instance, if $\boldsymbol{u}$ is the $i$-th unit vector and $\boldsymbol{x}_t \in [-1, +1]^n$, we obtain the finite expert framework considered in [CBFH$^+$97]. Here the $t$-th prediction of the $i$-th expert is $x_{t,i} = \boldsymbol{u} \cdot \boldsymbol{x}_t$, and $\frac{1}{2} D(\boldsymbol{u}; (\boldsymbol{x}_t, y_t)) = \frac{1}{2} |y_t - x_{t,i}| \in [0, 1]$ is exactly the absolute loss suffered by the $i$-th expert in the $t$-th trial. As another example, if $\boldsymbol{u}$ and $\boldsymbol{x}_t$ are $n$-dimensional $\{0, 1\}$-vectors then $\frac{1}{2} D(\boldsymbol{u}; (\boldsymbol{x}_t, y_t))$ corresponds to the so-called attribute error [Lit91] of $\boldsymbol{u}$ on $(\boldsymbol{x}_t, y_t)$. This quantity counts the minimal number of components of $\boldsymbol{x}_t$ that need to be flipped to make $\boldsymbol{u}$ classify $(\boldsymbol{x}_t, y_t)$ correctly.

**Theorem 8** *Let $\mathcal{W}_U = \{\boldsymbol{w} \in \mathcal{R}^n : ||\boldsymbol{w}||_q \leq U\}$ and $S = (\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots$, where $(\boldsymbol{x}_t, y_t) \in \mathcal{R}^n \times \mathcal{R}$. Then for any $\boldsymbol{u} \in \mathcal{W}_U$ the regression algorithm in Figure 2, run on a prefix of $S$ of arbitrary length $T$, achieves the following cumulative square loss bound:*

$$L_T \leq L_{\boldsymbol{u},T} + 4\, k_T + 4 \sqrt{k_T\, L_{\boldsymbol{u},T} + k_T^2},$$

*where*

$$L_{\boldsymbol{u},T} = \sum_{t=1}^{T} l_{\boldsymbol{u},t}$$

*and* $l_{\boldsymbol{u},t} = \frac{1}{2} (y_t - \boldsymbol{u} \cdot \boldsymbol{x}_t)^2$.

*Proof.* We notice that Lemma 5 applies. We can write

$$\frac{c_t}{1 + c_t}\, l_t - c_t\, l_{\boldsymbol{u},t} \leq (p - 1)\, X_t^2 (d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t^m)).$$

We divide by $c_t$ and apply Lemma 3. This lemma allows us to lower bound $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t^m)$ by $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1})$. We obtain

$$\frac{1}{1 + c_t}\, l_t - l_{\boldsymbol{u},t} \leq \frac{(p - 1)\, X_t^2}{c_t} \left( d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1}) \right). \tag{5}$$

We need to upper bound the RHS of (5). For this purpose, we first claim that the following inequalities hold:

$$\frac{X_{t+1}^2}{c_{t+1}} \geq \frac{X_t^2}{c_t}, \quad t = 1, \ldots, T. \tag{6}$$

To prove this claim, we define the function $g(x, L) = x \left( \sqrt{x^2 + L} - x \right)$. Computing its first derivatives, it is easy to see that $g(x, L)$ is nondecreasing in $x \geq 0$ for any $L \geq 0$ and nondecreasing in $L \geq 0$ for any $x \geq 0$. Now simple algebra gives

$$\begin{aligned} \frac{X_{t+1}^2}{c_{t+1}} &= g\left( X_{t+1}, \frac{L_{t+1}}{(p - 1)\, U^2} \right) \\ &\geq g\left( X_t, \frac{L_t}{(p - 1)\, U^2} \right) = \frac{X_t^2}{c_t}, \end{aligned}$$

thereby proving (6).

Therefore the RHS of (5) can be bounded as follows.

$$\begin{aligned} &\frac{X_t^2}{c_t} \left( d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1}) \right) \\ &\quad = \frac{X_t^2}{c_t}\, d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - \frac{X_{t+1}^2}{c_{t+1}}\, d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1}) \\ &\qquad + d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1}) \left( \frac{X_{t+1}^2}{c_{t+1}} - \frac{X_t^2}{c_t} \right) \\ &\quad \leq \frac{X_t^2}{c_t}\, d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - \frac{X_{t+1}^2}{c_{t+1}}\, d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1}) \\ &\qquad + 2\, U^2 \left( \frac{X_{t+1}^2}{c_{t+1}} - \frac{X_t^2}{c_t} \right), \end{aligned}$$

where in the inequality we applied (6) and Lemma 4.

Plugging back into (5) and summing over $t = 1, \ldots, T$ gives

$$\begin{aligned} &\sum_{t=1}^{T} \left( 1 - \frac{c_t}{1 + c_t} \right) l_t - L_{\boldsymbol{u},T} \\ &\quad \leq \frac{(p - 1)\, X_1^2}{c_1}\, d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_1) - \frac{(p - 1)\, X_{T+1}^2}{c_{T+1}}\, d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{T+1}) \\ &\qquad + 2 \left( \frac{k_{T+1}}{c_{T+1}} - \frac{k_1}{c_1} \right). \end{aligned}$$

Since $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{T+1}) \geq 0$ and $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_1) \leq 2\, U^2$ this implies

$$L_T - \sum_{t=1}^{T} \frac{c_t}{1 + c_t}\, l_t \leq L_{\boldsymbol{u},T} + \frac{2\, k_T}{c_T}, \tag{7}$$

where we set $c_{T+1} = c_T$ and $X_{T+1} = X_T$ (so that (6) is not violated and $k_{T+1} = k_T$).

Since

$$\frac{c_t}{1 + c_t} = \sqrt{\frac{k_t}{k_t + L_t}} \leq \sqrt{\frac{k_T}{k_T + L_t}},$$

we can apply Lemma 7 to the second term of the LHS of (7), along with the bound on $\frac{c_t}{1 + c_t}$ just given. We also substitute the value $c_T = \frac{\sqrt{k_T}}{\sqrt{k_T + L_T} - \sqrt{k_T}}$ into the RHS. This results in

$$\begin{aligned} &L_T - 2\sqrt{k_T} \left( \sqrt{k_T + L_T} - \sqrt{k_T} \right) \\ &\quad \leq L_{\boldsymbol{u},T} + 2\sqrt{k_T} \left( \sqrt{k_T + L_T} - \sqrt{k_T} \right). \end{aligned}$$

Simplifying and rearranging gets

$$k_T + L_T \leq 4\sqrt{k_T}\, \sqrt{k_T + L_T} + L_{\boldsymbol{u},T} - 3\, k_T.$$

We solve for $L_T + k_T$ and simplify. The larger of the roots of the equation obtained by using "=" instead of "$\leq$" gives the desired bound. $\Box$

**Theorem 9** *Let $\mathcal{W}_U = \{\boldsymbol{w} \in \mathcal{R}^n : ||\boldsymbol{w}||_q \leq U\}$ and $S = (\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots$, where $(\boldsymbol{x}_t, y_t) \in \mathcal{R}^n \times \{-1, +1\}$. Then for any $\boldsymbol{u} \in \mathcal{W}_U$ the algorithm in Figure 3, run on a prefix of $S$ of arbitrary length $T$, achieves the following cumulative absolute loss bound:*

$$L_T \leq \tfrac{1}{2} D_{\boldsymbol{u},T} + O\left(k_T + \sqrt{k_T\, D_{\boldsymbol{u},T}}\right),$$

*where*

$$D_{\boldsymbol{u},T} = \sum_{t=1,\, l_t>0}^{T} D_{\boldsymbol{u},t}$$

*and $D_{\boldsymbol{u},t} = D(\boldsymbol{u}; (\boldsymbol{x}_t, y_t))$.*

*Proof.* We denote by $\mathcal{M}$ the set of trials where the algorithm incurs a nonzero loss. Let us focus on a single trial $t \in \mathcal{M}$ and let $\boldsymbol{w}_t^m$ be as in Figure 3. We apply Lemma 6 and upper bound the last term $d_{\mathbf{f}}(\boldsymbol{w}_t, \boldsymbol{w}_t^m)$ as in [GLS97, GL99]: $d_{\mathbf{f}}(\boldsymbol{w}_t, \boldsymbol{w}_t^m) \leq \frac{\eta_t^2}{2}(p-1)\, X_t^2$. This yields

$$\begin{aligned} & y_t\, (\boldsymbol{u} \cdot \boldsymbol{x}_t - \boldsymbol{w}_t \cdot \boldsymbol{x}_t) \\ & \quad \leq \frac{1}{\eta_t}(d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t^m) + \frac{\eta_t^2}{2}(p-1)\, X_t^2). \end{aligned}$$

Next, we exploit the definition of $D(\boldsymbol{u}; (\boldsymbol{x}_t, y_t))$, from which it follows that $y_t\, \boldsymbol{u} \cdot \boldsymbol{x}_t \geq 1 - D(\boldsymbol{u}; (\boldsymbol{x}_t, y_t))$. Then, by Lemma 3, we lower bound $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t^m)$ through $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1})$ and rearrange:

$$\begin{aligned} & 1 - y_t\, \boldsymbol{w}_t \cdot \boldsymbol{x}_t - \frac{\eta_t}{2}(p-1)\, X_t^2 \\ & \quad \leq D_{\boldsymbol{u},t} + \frac{1}{\eta_t}\left(d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1})\right). \end{aligned}$$

Next, we claim that for any trial $t$ such that $l_t > 0$ the LHS of the last inequality is at least $2c_t\, l_t$, where $c_t = 1 - \alpha_t$. This would give us the one-trial loss bound

$$2c_t\, l_t \leq D_{\boldsymbol{u},t} + \frac{1}{\eta_t}\left(d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1})\right), \qquad (8)$$

holding for any $t$ such that $l_t > 0$ and any $\boldsymbol{u} \in \mathcal{W}_U$. We prove this claim by a case analysis over $y_t = -1,\ +1$. We work out the details only for $y_t = -1$, the other case being similar. If $y_t = -1$ then $l_t = \frac{1}{2}(1 + \hat{y}_t)$. Hence it suffices to prove that for any $r > -c_t$ we have

$$1 + r - \frac{\eta_t}{2}(p-1)\, X_t^2 \geq c_t\, (1 + \sigma_{c_t}(r)). \qquad (9)$$

(The case $r \leq -c_t$ is not our concern, as this would get $l_t = 0$.) We split into two subcases: $r \geq c_t$ and $-c_t < r < c_t$. If $r \geq c_t$ then the RHS of (9) is $2c_t$, since $\sigma_{c_t}(r) = 1$. On the other hand, recalling the value of $\eta_t$ in Figure 3, the LHS of (9) is at least $1 + c_t - \alpha_t = 2c_t$. If $-c_t < r < c_t$ then $\sigma_{c_t}(r) = r/c_t$. It is easy to see that in this case both sides of (9) are equal to $c_t + r$. This concludes the proof of (8).

The RHS of (8) is upper bounded as follows.

$$\begin{aligned} & \frac{1}{\eta_t}\left(d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1})\right) \\ & \quad = \frac{(p-1)\, X_t^2}{2\, \alpha_t}\left(d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1})\right) \\ & \quad = \frac{p-1}{2}\left(\frac{X_t^2}{\alpha_t} d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - \frac{X_{t+1}^2}{\alpha_{t+1}} d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1})\right. \\ & \qquad \left. + d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1})\left(\frac{X_{t+1}^2}{\alpha_{t+1}} - \frac{X_t^2}{\alpha_t}\right)\right) \\ & \quad \leq \frac{p-1}{2}\left(\frac{X_t^2}{\alpha_t} d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - \frac{X_{t+1}^2}{\alpha_{t+1}} d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1})\right. \\ & \qquad \left. + 2\, U^2\left(\frac{X_{t+1}^2}{\alpha_{t+1}} - \frac{X_t^2}{\alpha_t}\right)\right), \end{aligned}$$

where the last inequality follows from Lemma 4 and the fact (straightforward to check) that

$$\frac{X_{t+1}^2}{\alpha_{t+1}} \geq \frac{X_t^2}{\alpha_t}, \quad t = 1, \ldots, T. \qquad (10)$$

We plug this back into (8), sum over all $t \in \mathcal{M}$, drop the nonnegative term involving $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{T+1})$, divide by 2 and rearrange. We obtain

$$\begin{aligned} & L_T - \sum_{t=1}^{T} \alpha_t\, l_t \\ & \quad \leq \frac{D_{\boldsymbol{u},T}}{2} + \frac{p-1}{4}\left(\frac{X_1^2}{\alpha_1} d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_1)\right. \\ & \qquad \left. + 2\, U^2\left(\frac{X_{T+1}^2}{\alpha_{T+1}} - \frac{X_1^2}{\alpha_1}\right)\right) \\ & \quad = \frac{D_{\boldsymbol{u},T}}{2} + \frac{k_{T+1}}{2\, \alpha_{T+1}} - \frac{A}{\alpha_1}, \end{aligned} \qquad (11)$$

where we have set

$$A = \tfrac{1}{2} k_1 - \frac{(p-1)\, X_1^2\, d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_1)}{4}.$$

We now handle (11) as follows. We set $X_{T+1} = X_T$, so that (10) is not violated and $\alpha_{T+1} = \sqrt{\frac{k_T/4}{k_T/4 + L_t + 1}}$. Next, since $A \geq 0$ (Lemma 4) and $\alpha_1 \leq 1$, we upper bound $\frac{A}{\alpha_1}$ by $A$. Finally, we upper bound $\alpha_t$ in the left-most side of (11) by $\sqrt{\frac{k_T/4}{k_T/4 + L_t}}$ and then apply Lemma 7 with the bound on $\alpha_t$ just given. This leads to

$$\begin{aligned} & L_T - \sqrt{k_T}\left(\sqrt{k_T/4 + L_T} - \tfrac{1}{2}\sqrt{k_T}\right) \\ & \qquad \leq \frac{D_{\boldsymbol{u},T}}{2} + \sqrt{k_T}\sqrt{k_T/4 + L_T + 1} - A. \end{aligned}$$

By virtue of the inequality $\sqrt{1 + x} \leq \sqrt{x} + \frac{1}{2\sqrt{x}}$, $x \geq 0$, the second term of the RHS is bounded from above by

$$\begin{aligned} & \sqrt{k_T}\left(\sqrt{k_T/4 + L_T} + \frac{1}{2\sqrt{k_T/4 + L_T}}\right) \\ & \qquad \leq \sqrt{k_T}\sqrt{k_T/4 + L_T} + 1. \end{aligned}$$

Simple algebra then gives

$$k_T/4 + L_T \leq \frac{D_{\boldsymbol{u},T}}{2} + 2\sqrt{k_T}\,\sqrt{k_T/4 + L_T} - k_T/4 + 1 - A.$$

We solve for $L_T + k_T/4$ and simplify. Again, we compute the larger of the roots of the equation obtained by using "=" instead of "$\leq$". This gives the bound of the theorem. □

These two results have some interesting properties. The dual norms quantity $k_T$ is a function of the norm $p$. This affects the dependence of the $p$-norm algorithm on the dimension $n$ of the input space. Recall that $||\boldsymbol{x}_t||_p \leq X$ for all $t$ implies that $k_T \leq (p-1)\,X^2\,U^2$. For instance, if $p = 2\ln n$ [GL99] then[5] $k_T < 2\,e\,\ln n\,U_1^2 X_\infty^2$, where $U_1$ is an upper bound on the 1-norm of $\boldsymbol{u}$ and $X_\infty$ is an upper bound on the $\infty$-norm of the instances. In the expert case we have $U_1 = X_\infty = 1$. Thus $p = 2\ln n$ yields $k_T < 2e\ln n$. This gives rise to a loss bound which is similar to the one we have proven in Section 2 for the adaptive Weighted Majority algorithm. (The constants hidden in the bound of Theorem 9 are actually slightly larger.)

### 3.2 Self-confident Winnow-EG-like algorithms

From the proofs of Theorems 8 and 9 the reader can see that our technique applies to a generic quasi-additive algorithm with mapping $\mathbf{f}$, as long as we can both: find a constant upper bound [6] on the divergence terms $d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1})$, and show a one-trial loss bound of the form

$$(1-\alpha)\,l_t - l_{\boldsymbol{u},t} \leq \tfrac{c}{\alpha}(d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_t) - d_{\mathbf{f}}(\boldsymbol{u}, \boldsymbol{w}_{t+1})),$$

where $l_t$ is the loss of the algorithm in trial $t$, $l_{\boldsymbol{u},t}$ is the loss of the generic off-line predictor $\boldsymbol{u}$ in trial $t$, $\alpha \in (0,1)$ is a constant proportional to $\eta$ and $c > 0$.

Consider applying this proof technique to algorithms in the Winnow-EG family [Lit88, Lit89, KW97], such as Weighted Majority. The measure of progress typically associated with these algorithms is a relative entropy-like divergence. Proving the required one-trial loss bound is quite standard. Rather, the difficulty here stems from the fact that the relative entropy is hardly upper bounded, unless we introduce a lower bound constraint on the weights of the algorithm. Via this proof technique, it is nevertheless possible to prove self-confident bounds for these algorithms. These bounds, however, have larger lower-order terms then those of the corresponding self-confident $p$-norm bounds in Theorems 8 and 9.

On the other hand, algorithms like Weighted Majority can be used in frameworks where the $p$-norm algorithms are harder to apply. As a relevant example, both the standard analysis [LW94, Vov98] and our self-confident analysis for Weighted Majority still hold when the dimension of the input space (the number of experts) is countably infinite. This is an interesting setting, since it can clearly formalize an on-line model selection problem with countably many models. (For instance, this can be used back in Section 3.1 to find a good value for the parameter $U$ when we ignore a bound on the norm of the comparison vector.)

[5] Here $e$ is the base of natural logarithms.

[6] In the case of the $p$-norm algorithms this is achieved through Lemma 4.

## 4 Conclusions and ongoing research

We have studied on-line learning algorithms with an incrementally adaptive learning rate. We have provided the first analysis of an adaptive Weighted Majority in the finite expert framework to date. This result compares favourably with previous bounds for Weighted Majority based on doubling strategies. For more general regression tasks we sketched a versatile and general technique to turn a constant learning rate algorithm into a variable learning rate algorithm, called self-confident algorithm. We focused on the analysis of self-confident $p$-norm algorithms proving regret bounds that are easily generalizable to a wide range of convex losses.

As pointed out before, our analysis yields suboptimal results when applied to Winnow-EG-like algorithms, and we are currently investigating the extent to which this problem could be fixed. Further ongoing research concerns the application of our techniques to settings more general than the one studied here, e.g., the so-called shifting target setting [HW98, AW98, HW98].

Finally, we are performing experiments applying our variable learning rate algorithms to the problem of handwritten digit recognition. The preliminary results we have obtained so far are very encouraging. We are reporting on them in [G00].

**Acknowledgments**

Claudio Gentile is supported by a post-doctoral fellowship from Università degli Studi di Milano and by MURST, Cofin99 (Project: Stochastic Processes with Spatial Structure). Both authors acknowledge partial support of ESPRIT Working Group EP 27150, Neural and Computational Learning II (NeuroCOLT II).

## References

[Ang88] Angluin, D. Queries and concept learning, *Machine Learning*, 2(4): 319-342, 1988.

[AW98] P. Auer and M. K. Warmuth. Tracking the best disjunction. *Machine Learning*, 32(2): 127–150, August 1998. Special issue on concept drift.

[AW99] K. Azoury and M. K. Warmuth. Relative loss bounds for on-line density estimation with the exponential family of distributions". In *Proc. Uncertainty in Artificial Intelligence*, 1999.

[Blo62] H. D. Block. The perceptron: A model for brain functioning. *Reviews of Modern Physics*, 34:123–135, 1962. Reprinted in Neurocomputing by Anderson and Rosenfeld.

[Bre67] L.M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Physics*, 7:200–217, 1967.

[Byl97] T. Bylander. The binary exponentiated gradient algorithm for learning linear functions. In *Proc. 8th Annu. Conf. on Comput. Learning Theory*, pages 184–192. ACM, 1997.

[CBFH+97] N. Cesa-Bianchi, Y. Freund, D. Haussler, D. P. Helmbold, R. E. Schapire, and M. K. Warmuth.

How to use expert advice. *Journal of the ACM*, 44(3):427–485, 1997.

[CBLW96] N. Cesa-Bianchi and P. Long and M. K. Warmuth, Worst-case quadratic loss bounds for on-line prediction of linear functions by gradient descent, *IEEE Transactions on Neural Networks*, 7:604–619, 1996.

[CB99] N. Cesa-Bianchi. Analysis of two gradient-based algorithms for on-line regression, *Journal of Computer and System Sciences*, 59 (3): 392–411, 1999.

[CL81] Y. Censor and A. Lent. An iterative row-action method for interval convex programming. *Journal of Optimization Theory and Applications*, 34(3):321–353, July 1981.

[GW98] C. Gentile, and M. K. Warmuth. Linear hinge loss and average margin. In *M. S. Kearns, S. A. Solla, and D. A. Cohn, editors, Advances in Neural Information Processing Systems 11, MIT Press, Cambridge, MA, 1999*, pages 225–231.

[GL99] C. Gentile, and N. Littlestone. The robustness of the $p$-norm algorithms. In *Proc. 12th Annu. Conf. on Comput. Learning Theory*, pages 1–11.

[G00] C. Gentile. Approximate maximal margin classification with respect to an arbitrary norm. Manuscript, May, 2000.

[Go99] G. J. Gordon. Regret bounds for prediction problems. In *Proc. 12th Annu. Conf. on Comput. Learning Theory*, pages 29–40. ACM, 1999.

[GLS97] A. J. Grove, N. Littlestone, and D. Schuurmans. General convergence results for linear discriminant updates. In *Proc. 10th Annu. Conf. on Comput. Learning Theory*, pages 171–183. ACM, 1997.

[HKW95] D. P. Helmbold, J. Kivinen, and M. K. Warmuth. Worst-case loss bounds for sigmoided linear neurons. *IEEE Transactions on Neural Networks*, 10(6): 1291–1304, November 1999.

[HW98] M. Herbster, and M. K. Warmuth. Tracking the best expert. *Machine Learning*, 32(2): 151–178, August 1998. Special issue on concept drift.

[HW98] M. Herbster and M. K. Warmuth. Tracking the best regressor. In *Proc. 11th Annu. Conf. on Comput. Learning Theory*, pages 24–31. ACM, 1998.

[JW98] A. Jagota and M. K. Warmuth. Continuous and discrete time nonlinear gradient descent: relative loss bounds and convergence. In R. Greiner E. Boros, editor, *Electronic Proceedings of Fifth International Symposium on Artificial Intelligence and Mathematics*. Electronic,http://rutcor.rutgers.edu/˜amai, 1998.

[KW97] J. Kivinen and M. K. Warmuth. Additive versus exponentiated gradient updates for linear prediction. *Information and Computation*, 132(1):1–64, January 1997.

[KW98b] J. Kivinen and M. K. Warmuth. Relative loss bounds for multidimensional regression problems. In *M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, Advances in Neural Information Processing Systems 10 (NIPS*97)*, pages 287–293. MIT Press, Cambridge, MA, 1998. To appear in *Machine Learning*.

[Lit88] N. Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning*, 2:285–318, 1988.

[Lit89] N. Littlestone. *Mistake Bounds and Logarithmic Linear-threshold Learning Algorithms*. PhD thesis, TR UCSC-CRL-89-11, University of California Santa Cruz, 1989.

[Lit91] N. Littlestone. Redundant noisy attributes, attribute errors, and linear threshold learning using Winnow. In *Proc. 4th Annu. Workshop on Comput. Learning Theory*, pages 147–156, San Mateo, CA, 1991. Morgan Kaufmann.

[LW94] N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

[Nov62] A. B. J. Novikov. On convergence proofs on perceptrons. In *Proc. of the Symposium on the Mathematical Theory of Automata, vol. XII*, pages 615–622, 1962.

[Ros62] F. Rosenblatt. *Principles of neurodynamics: Perceptrons and the theory of brain mechanisms.* Spartan Books, Washington, D.C., 1962.

[Vov90] V. Vovk. Aggregating strategies. In *Proc. 3rd Annu. Workshop on Comput. Learning Theory*, pages 371–383. Morgan Kaufmann, 1990.

[Vov97] V. Vovk, Competitive on-line linear regression, TR, Royal Holloway, University of London, CSD-TR-97-13, 1997, http://www.cs.rhbnc.ac.uk/research/compint /areas/comp_learn/aa/nips.ps. Preliminary version in *M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, Advances in Neural Information Processing Systems 10 (NIPS*97)*, pages 364–370. MIT Press, Cambridge, MA, 1998.

[Vov98] V. Vovk. A game of prediction with expert advice. *Journal of Computer and System Sciences*, 56(2):153–173, 1998.

[Vov99] V. Vovk, Derandomizing stochastic prediction strategies. *Machine Learning*, 35(3): 247-282, June 1999.

[WH60] B. Widrow and M. E. Hoff, Adaptive switching circuits. In *1960 IRE WESCON Conv. Record, Part 4*, pages 96–104, 1960.

[Yam98] K. Yamanishi, A decision-theoretic extension of stochastic complexity and its applications to learning. *IEEE Information Theory*, 44(4):1424–1439, 1998.

# A Appendix

This appendix contains the proofs of Lemmas 2, 4 and 7.

**Proof of Lemma 2**
First we show by induction that

$$\frac{1+\sum_{i=1}^{n-1}\beta^{-\ell_i}}{1+\sum_{i=1}^{n-1}\alpha^{-\ell_i}} = \frac{1+(n-1)\beta^{-\ell}}{1+(n-1)\alpha^{-\ell}} \tag{12}$$

for some $\ell \geq 0$. Assume that we have

$$\frac{1+\sum_{i=1}^{n-1}\beta^{-\ell_i}}{1+\sum_{i=1}^{n-1}\alpha^{-\ell_i}} = \frac{1+(k-1)\beta^{-\ell}+\sum_{i=k}^{n-1}\beta^{-\ell_i}}{1+(k-1)\alpha^{-\ell}+\sum_{i=k}^{n-1}\alpha^{-\ell_i}}$$

for some $k \geq 1$ and $\ell \geq 0$. Now we prove that there is an $\ell' \geq 0$ such that

$$\begin{aligned}&\frac{1+(k-1)\beta^{-\ell}+\sum_{i=k}^{n-1}\beta^{-\ell_i}}{1+(k-1)\alpha^{-\ell}+\sum_{i=k}^{n-1}\alpha^{-\ell_i}}\\ &\quad = \quad \frac{1+k\beta^{-\ell'}+\sum_{i=k+1}^{n-1}\beta^{-\ell_i}}{1+k\alpha^{-\ell'}+\sum_{i=k+1}^{n-1}\alpha^{-\ell_i}}.\end{aligned} \tag{13}$$

We set

$$f(\lambda_1,\lambda_2) = \frac{1+(k-1)\beta^{-\lambda_1}+\beta^{-\lambda_2}+\sum_{i=k+1}^{n-1}\beta^{-\ell_i}}{1+(k-1)\alpha^{-\lambda_1}+\beta^{-\lambda_2}+\sum_{i=k+1}^{n-1}\alpha^{-\ell_i}}$$

and find by a simple calculation that

$$\begin{aligned}&\frac{f(\ell,\ell_k)-f(\ell_k,\ell_k)}{f(\ell,\ell_k)-f(\ell,\ell)}\\ &\quad = \quad -(k-1)\frac{1+k\alpha^{-\ell}+\sum_{i=k+1}^{n-1}\alpha^{-\ell_i}}{1+k\alpha^{-\ell_k}+\sum_{i=k+1}^{n-1}\alpha^{-\ell_i}}.\end{aligned}$$

Thus $f(\ell,\ell_k)-f(\ell_k,\ell_k)$ and $f(\ell,\ell_k)-f(\ell,\ell)$ have different sign. Then, since $f$ is a continuous function, there is an $\ell'$ between $\ell$ and $\ell_k$ such that $f(\ell,\ell_k)-f(\ell',\ell') = 0$. This gives (13) and by induction (12).

We proceed with the proof of Lemma 2 by upper bounding $\ln\left(\frac{1+(n-1)\alpha^{-\ell}}{1+(n-1)\beta^{-\ell}}\right)$. Since this term is increasing in $\beta$ and $\ln(\cdot)$ is a monotone and concave function we get

$$\begin{aligned}&\ln\left(\frac{1+(n-1)\alpha^{-\ell}}{1+(n-1)\beta^{-\ell}}\right)\\ &\quad \leq \quad (\beta-\alpha)\,\frac{\partial}{\partial\beta}\ln\left(\frac{1+(n-1)\alpha^{-\ell}}{1+(n-1)\beta^{-\ell}}\right)\Bigg|_{\beta=\alpha}\\ &\quad = \quad (\beta-\alpha)\frac{(n-1)\ell\alpha^{-\ell-1}}{1+(n-1)\alpha^{-\ell}}\\ &\quad \leq \quad (\beta-\alpha)\frac{(n-1)\ell\alpha^{-\ell}}{1+(n-1)\alpha^{-\ell}}.\end{aligned}$$

In the following we use the fact that $\ell\alpha^{-\ell}$ is maximal when $\ell = 1/\ln\alpha$, is increasing in $\ell$ for smaller $\ell$, and is decreasing in $\ell$ for larger $\ell$. Assume that $\ell \geq \frac{\ln n}{\ln\alpha}$. Then for $n > 2$ we have $\ell \geq 1/\ln\alpha$ and

$$\frac{(n-1)\ell\alpha^{-\ell}}{1+(n-1)\alpha^{-\ell}} \leq \frac{\ln n}{\ln\alpha}.$$

For $n = 2$ we get

$$\frac{(n-1)\ell\alpha^{-\ell}}{1+(n-1)\alpha^{-\ell}} \leq \frac{1}{(e+1)\ln\alpha} \leq \frac{\ln 2}{\ln\alpha}.$$

Now assume $\ell \leq \frac{\ln n}{\ln\alpha}$. Then

$$\frac{(n-1)\ell\alpha^{-\ell}}{1+(n-1)\alpha^{-\ell}} \leq \ell \leq \frac{\ln n}{\ln\alpha}.$$

This completes the proof of the lemma. □

**Proof of Lemma 4**
The assertion follows from (4) and Holder's inequality on the term $\boldsymbol{u}\cdot\mathbf{f}(\boldsymbol{w})$, for $\boldsymbol{u}\cdot\mathbf{f}(\boldsymbol{w}) \leq ||\boldsymbol{u}||_q\,||\mathbf{f}(\boldsymbol{w})||_p = ||\boldsymbol{u}||_q\,||\boldsymbol{w}||_q \leq U^2$, where the equality uses the fact that $||\mathbf{f}(\boldsymbol{w})||_p = ||\boldsymbol{w}||_q$ (see Lemma 1, part 3 in [GL99]). □

**Proof of Lemma 7**
Let $l_0 = \delta$ and $L_t = \sum_{i=0}^{t} l_i$, $t = 0,\ldots,T$. In the inequality $\frac{1}{2}x \leq 1-\sqrt{1-x},\ x \leq 1$, set $x = \frac{l_t}{L_t}$ and then multiply both terms of the resulting inequality by $\sqrt{L_t}$. This yields

$$\frac{1}{2}\frac{l_t}{\sqrt{L_t}} \leq \sqrt{L_t}-\sqrt{L_{t-1}}.$$

The claim is then obtained by summing over $t = 1,\ldots,T$. □