Adaptive Hedge

Tim van Erven
Department of Mathematics
VU University
De Boelelaan 1081a
1081 HV Amsterdam, the Netherlands
tim@timvanerven.nl

Peter Grünwald
Centrum Wiskunde & Informatica (CWI)
Science Park 123, P.O. Box 94079
1090 GB Amsterdam, the Netherlands
pdg@cwi.nl

Wouter M. Koolen
CWI and Department of Computer Science
Royal Holloway, University of London
Egham Hill, Egham, Surrey
TW20 0EX, United Kingdom
wouter@cs.rhul.ac.uk

Steven de Rooij
Centrum Wiskunde & Informatica (CWI)
Science Park 123, P.O. Box 94079
1090 GB Amsterdam, the Netherlands
s.de.rooij@cwi.nl

Abstract

Most methods for decision-theoretic online learning are based on the Hedge algorithm, which takes a parameter called the *learning rate*. In most previous analyses the learning rate was carefully tuned to obtain optimal worst-case performance, leading to suboptimal performance on easy instances, for example when there exists an action that is significantly better than all others. We propose a new way of setting the learning rate, which adapts to the difficulty of the learning problem: in the worst case our procedure still guarantees optimal performance, but on easy instances it achieves much smaller regret. In particular, our adaptive method achieves constant regret in a probabilistic setting, when there exists an action that on average obtains strictly smaller loss than all other actions. We also provide a simulation study comparing our approach to existing methods.

1 Introduction

Decision-theoretic online learning (DTOL) is a framework to capture learning problems that proceed in rounds. It was introduced by Freund and Schapire [1] and is closely related to the paradigm of *prediction with expert advice* [2, 3, 4]. In DTOL an agent is given access to a fixed set of K actions, and at the start of each round must make a decision by assigning a probability to every action. Then all actions incur a loss from the range $[0, 1]$, and the agent's loss is the expected loss of the actions under the probability distribution it produced. Losses add up over rounds and the goal for the agent is to minimize its *regret* after T rounds, which is the difference in accumulated loss between the agent and the action that has accumulated the least amount of loss.

The most commonly studied strategy for the agent is called the *Hedge* algorithm [1, 5]. Its performance crucially depends on a parameter η called the *learning rate*. Different ways of tuning the learning rate have been proposed, which all aim to minimize the regret for the worst possible sequence of losses the actions might incur. If T is known to the agent, then the learning rate may be tuned to achieve worst-case regret bounded by $\sqrt{T\ln(K)/2}$, which is known to be optimal as T and K become large [4]. Nevertheless, by slightly relaxing the problem, one can obtain better guarantees. Suppose for example that the cumulative loss L_T^* of the best action is known to the agent beforehand. Then, if the learning rate is set appropriately, the regret is bounded by $\sqrt{2L_T^*\ln(K)} + \ln(K)$ [4], which has the same asymptotics as the previous bound in the worst case

(because $L_T^* \leq T$) but may be much better when L_T^* turns out to be small. Similarly, Hazan and Kale [6] obtain a bound of $8\sqrt{\text{VAR}_T^{\max} \ln(K)} + 10 \ln(K)$ for a modification of Hedge if the cumulative empirical variance $\text{VAR}_T^{\max}$ of the best expert is known. In applications it may be unrealistic to assume that T or (especially) L_T^* or $\text{VAR}_T^{\max}$ is known beforehand, but at the cost of slightly worse constants such problems may be circumvented using either the *doubling trick* (setting a budget on the unknown quantity and restarting the algorithm with a double budget when the budget is depleted) [4, 7, 6], or a *variable learning rate* that is adjusted each round [4, 8].

Bounding the regret in terms of L_T^* or $\text{VAR}_T^{\max}$ is based on the idea that worst-case performance is not the only property of interest: such bounds give essentially the same guarantee in the worst case, but a much better guarantee in a plausible favourable case (when L_T^* or $\text{VAR}_T^{\max}$ is small). In this paper, we pursue the same goal for a different favourable case. To illustrate our approach, consider the following simplistic example with two actions: let $0 < a < b < 1$ be such that $b - a > 2\epsilon$. Then in odd rounds the first action gets loss $a + \epsilon$ and the second action gets loss $b - \epsilon$; in even rounds the actions get losses $a - \epsilon$ and $b + \epsilon$, respectively. Informally, this seems like a very easy instance of DTOL, because the cumulative losses of the actions diverge and it is easy to see from the losses which action is the best one. In fact, the *Follow-the-Leader* strategy, which puts all probability mass on the action with smallest cumulative loss, gives a regret of at most 1 in this case — the worst-case bound $O(\sqrt{L_T^* \ln(K)})$ is very loose by comparison, and so is $O(\sqrt{\text{VAR}_T^{\max} \ln(K)})$, which is of the same order $\sqrt{T \ln(K)}$. On the other hand, for Follow-the-Leader one cannot guarantee sublinear regret for worst-case instances. (For example, if one out of two actions yields losses $\frac{1}{2}, 0, 1, 0, 1, \ldots$ and the other action yields losses $0, 1, 0, 1, 0, \ldots$, its regret will be at least $T/2 - 1$.) To get the best of both worlds, we introduce an adaptive version of Hedge, called *AdaHedge*, that automatically adapts to the difficulty of the problem by varying the learning rate appropriately. As a result we obtain constant regret for the simplistic example above and other 'easy' instances of DTOL, while at the same time guaranteeing $O(\sqrt{L_T^* \ln(K)})$ regret in the worst case.

It remains to characterise what we consider easy problems, which we will do in terms of the probabilities produced by Hedge. As explained below, these may be interpreted as a generalisation of Bayesian posterior probabilities. We measure the difficulty of the problem in terms of the speed at which the posterior probability of the best action converges to one. In the previous example, this happens at an exponential rate, whereas for worst-case instances the posterior probability of the best action does not converge to one at all.

Outline In the next section we describe a new way of tuning the learning rate, and show that it yields essentially optimal performance guarantees in the worst case. To construct the AdaHedge algorithm, we then add the doubling trick to this idea in Section 3, and analyse its worst-case regret. In Section 4 we show that AdaHedge in fact incurs much smaller regret on easy problems. We compare AdaHedge to other instances of Hedge by means of a simulation study in Section 5. The proof of our main technical lemma is postponed to Section 6, and open questions are discussed in the concluding Section 7. Finally, longer proofs are only available as Additional Material in the full version at arXiv.org.

2 Tuning the Learning Rate

Setting Let the available actions be indexed by $k \in \{1, \ldots, K\}$. At the start of each round $t = 1, 2, \ldots$ the agent A is to assign a probability w_t^k to each action k by producing a vector $\boldsymbol{w}_t = (w_t^1, \ldots, w_t^K)$ with nonnegative components that sum up to 1. Then every action k incurs a loss $\ell_t^k \in [0, 1]$, which we collect in the loss vector $\boldsymbol{\ell}_t = (\ell_t^1, \ldots, \ell_t^K)$, and the loss of the agent is $\boldsymbol{w}_t \cdot \boldsymbol{\ell}_t = \sum_{k=1}^K w_t^k \ell_t^k$. After T rounds action k has accumulated loss $L_T^k = \sum_{t=1}^T \ell_t^k$, and the agent's regret is

$$R_A(T) = \sum_{t=1}^T \boldsymbol{w}_t \cdot \boldsymbol{\ell}_t - L_T^*,$$

where $L_T^* = \min_{1 \leq k \leq K} L_T^k$ is the cumulative loss of the best action.

Hedge The Hedge algorithm chooses the weights w_{t+1}^k proportional to $e^{-\eta L_t^k}$, where $\eta > 0$ is the learning rate. As is well-known, these weights may essentially be interpreted as Bayesian posterior probabilities on actions, relative to a uniform prior and pseudo-likelihoods $P_t^k = e^{-\eta L_t^k} = \prod_{s=1}^t e^{-\eta \ell_s^k}$ [9, 10, 4]:

$$w_{t+1}^k = \frac{e^{-\eta L_t^k}}{\sum_{k'} e^{-\eta L_t^{k'}}} = \frac{\frac{1}{K} \cdot P_t^k}{B_t},$$

where

$$B_t = \sum_k \tfrac{1}{K} \cdot P_t^k = \sum_k \tfrac{1}{K} \cdot e^{-\eta L_t^k} \tag{1}$$

is a generalisation of the Bayesian *marginal likelihood*. And like the ordinary marginal likelihood, B_t factorizes into sequential per-round contributions:

$$B_t = \prod_{s=1}^t \boldsymbol{w}_s \cdot e^{-\eta \boldsymbol{\ell}_s}. \tag{2}$$

We will sometimes write $\boldsymbol{w}_t(\eta)$ and $B_t(\eta)$ instead of $\boldsymbol{w}_t$ and B_t in order to emphasize the dependence of these quantities on η.

The Learning Rate and the Mixability Gap A key quantity in our and previous [4] analyses is the gap between the per-round loss of the Hedge algorithm and the per-round contribution to the negative logarithm of the "marginal likelihood" B_T, which we call the *mixability gap*:

$$\delta_t(\eta) = \boldsymbol{w}_t(\eta) \cdot \boldsymbol{\ell}_t - \Big(-\tfrac{1}{\eta} \ln(\boldsymbol{w}_t(\eta) \cdot e^{-\eta \boldsymbol{\ell}_t}) \Big).$$

In the setting of prediction with expert advice, the subtracted term coincides with the loss incurred by the Aggregating Pseudo-Algorithm (APA) which, by allowing the losses of the actions to be *mixed* with optimal efficiency, provides an idealised lower bound for the actual loss of any prediction strategy [9]. The mixability gap measures how closely we approach this ideal. As the same interpretation still holds in the more general DTOL setting of this paper, we can measure the difficulty of the problem, and tune η, in terms of the cumulative mixability gap:

$$\Delta_T(\eta) = \sum_{t=1}^T \delta_t(\eta) = \sum_{t=1}^T \boldsymbol{w}_t(\eta) \cdot \boldsymbol{\ell}_t + \tfrac{1}{\eta} \ln B_T(\eta).$$

We proceed to list some basic properties of the mixability gap. First, it is nonnegative and bounded above by a constant that depends on η:

Lemma 1. *For any t and $\eta > 0$ we have $0 \le \delta_t(\eta) \le \eta/8$.*

Proof. The lower bound follows by applying Jensen's inequality to the concave function $\ln$, the upper bound from Hoeffding's bound on the cumulant generating function [4, Lemma A.1]. □

Further, the cumulative mixability gap $\Delta_T(\eta)$ can be related to L_T^* via the following upper bound, proved in the Additional Material:

Lemma 2. *For any T and $\eta \in (0, 1]$ we have $\Delta_T(\eta) \le \dfrac{\eta L_T^* + \ln(K)}{e - 1}$.*

This relationship will make it possible to provide worst-case guarantees similar to what is possible when η is tuned in terms of L_T^*. However, for easy instances of DTOL this inequality is very loose, in which case we can prove substantially better regret bounds. We could now proceed by optimizing the learning rate η given the rather awkward assumption that $\Delta_T(\eta)$ is bounded by a known constant b for all η, which would be the natural counterpart to an analysis that optimizes η when a bound on L_T^* is known. However, as $\Delta_T(\eta)$ varies with η and is unknown a priori anyway, it makes more sense to turn the analysis on its head and start by fixing η. We can then simply run the Hedge algorithm until the smallest T such that $\Delta_T(\eta)$ exceeds an appropriate budget $b(\eta)$, which we set to

$$b(\eta) = \Big(\tfrac{1}{\eta} + \tfrac{1}{e-1} \Big) \ln(K). \tag{3}$$

When at some point the budget is depleted, i.e. $\Delta_T(\eta) \geq b(\eta)$, Lemma 2 implies that

$$\eta \geq \sqrt{(e-1)\ln(K)/L_T^*}, \tag{4}$$

so that, up to a constant factor, the learning rate used by AdaHedge is at least as large as the learning rates proportional to $\sqrt{\ln(K)/L_T^*}$ that are used in the literature. On the other hand, it is not *too* large, because we can still provide a bound of order $O(\sqrt{L_T^* \ln(K)})$ on the worst-case regret:

Theorem 3. *Suppose the agent runs Hedge with learning rate $\eta \in (0, 1]$, and after T rounds has just used up the budget* (3)*, i.e. $b(\eta) \leq \Delta_T(\eta) < b(\eta) + \eta/8$. Then its regret is bounded by*

$$R_{\textit{Hedge}(\eta)}(T) < \sqrt{\tfrac{4}{e-1} L_T^* \ln(K)} + \tfrac{1}{e-1}\ln(K) + \tfrac{1}{8}.$$

Proof. The cumulative loss of Hedge is bounded by

$$\sum_{t=1}^{T} \boldsymbol{w}_t \cdot \boldsymbol{\ell}_t = \Delta_T(\eta) - \tfrac{1}{\eta}\ln B_T < b(\eta) + \eta/8 - \tfrac{1}{\eta}\ln B_T \leq \tfrac{1}{e-1}\ln(K) + \tfrac{1}{8} + \tfrac{2}{\eta}\ln(K) + L_T^*, \tag{5}$$

where we have used the bound $B_T \geq \frac{1}{K} e^{-\eta L_T^*}$. Plugging in (4) completes the proof. □

3 The AdaHedge Algorithm

We now introduce the AdaHedge algorithm by adding the doubling trick to the analysis of the previous section. The doubling trick divides the rounds in segments $i = 1, 2, \ldots$, and on each segment restarts Hedge with a different learning rate η_i. For AdaHedge we set $\eta_1 = 1$ initially, and scale down the learning rate by a factor of $\phi > 1$ for every new segment, such that $\eta_i = \phi^{1-i}$. We monitor $\Delta_t(\eta_i)$, measured only on the losses in the i-th segment, and when it exceeds its budget $b_i = b(\eta_i)$ a new segment is started. The factor ϕ is a parameter of the algorithm. Theorem 5 below suggests setting its value to the golden ratio $\phi = (1 + \sqrt{5})/2 \approx 1.62$ or simply to $\phi = 2$.

Algorithm 1 AdaHedge(ϕ) ▷ *Requires $\phi > 1$*

```
η ← φ
for t = 1, 2, ... do
    if t = 1 or Δ ≥ b then
        ▷ Start a new segment
        η ← η/φ;  b ← (1/(e−1) + 1/η) ln(K)
        Δ ← 0;  w = (w^1, ..., w^K) ← (1/K, ..., 1/K)
    end if
    ▷ Make a decision
    Output probabilities w for round t
    Actions receive losses ℓ_t
    ▷ Prepare for the next round
    Δ ← Δ + w · ℓ_t + (1/η) ln(w · e^{−η ℓ_t})
    w ← (w^1 · e^{−η ℓ_t^1}, ..., w^K · e^{−η ℓ_t^K}) / (w · e^{−η ℓ_t})
end for
end
```

The regret of AdaHedge is determined by the number of segments it creates: the fewer segments there are, the smaller the regret.

Lemma 4. *Suppose that after T rounds, the AdaHedge algorithm has started m new segments. Then its regret is bounded by*

$$R_{\textit{AdaHedge}}(T) < 2\ln(K)\Big(\frac{\phi^m - 1}{\phi - 1}\Big) + m\Big(\tfrac{1}{e-1}\ln(K) + \tfrac{1}{8}\Big).$$

Proof. The regret per segment is bounded as in (5). Summing over all m segments, and plugging in $\sum_{i=1}^{m} 1/\eta_i = \sum_{i=0}^{m-1} \phi^i = (\phi^m - 1)/(\phi - 1)$ gives the required inequality. □

Using (4), one can obtain an upper bound on the number of segments that leads to the following guarantee for AdaHedge:

Theorem 5. *Suppose the agent runs AdaHedge for T rounds. Then its regret is bounded by*

$$R_{\textit{AdaHedge}}(T) \leq \frac{\phi\sqrt{\phi^2-1}}{\phi-1}\sqrt{\tfrac{4}{e-1}L_T^*\ln(K)} + O\big(\ln(L_T^*+2)\ln(K)\big),$$

For details see the proof in the Additional Material. The value for ϕ that minimizes the leading factor is the golden ratio $\phi = (1+\sqrt{5})/2$, for which $\phi\sqrt{\phi^2-1}/(\phi-1) \approx 3.33$, but simply taking $\phi = 2$ leads to a very similar factor of $\phi\sqrt{\phi^2-1}/(\phi-1) \approx 3.46$.

4 Easy Instances

While the previous sections reassure us that AdaHedge performs well for the worst possible sequence of losses, we are also interested in its behaviour when the losses are not maximally antagonistic. We will characterise such sequences in terms of convergence of the Hedge posterior probability of the best action:

$$w_t^*(\eta) = \max_{1\leq k\leq K} w_t^k(\eta).$$

(Recall that w_t^k is proportional to $e^{-\eta L_{t-1}^k}$, so w_t^* corresponds to the posterior probability of the action with smallest cumulative loss.) Technically, this is expressed by the following refinement of Lemma 1, which is proved in Section 6.

Lemma 6. *For any t and $\eta \in (0,1]$ we have $\delta_t(\eta) \leq (e-2)\eta\big(1-w_t^*(\eta)\big)$.*

This lemma, which may be of independent interest, is a variation on Hoeffding's bound on the cumulant generating function. While Lemma 1 leads to a bound on $\Delta_T(\eta)$ that grows linearly in T, Lemma 6 shows that $\Delta_T(\eta)$ may grow much slower. In fact, if the posterior probabilities w_t^* converge to 1 sufficiently quickly, then $\Delta_T(\eta)$ is bounded, as shown by the following lemma. Recall that $L_T^* = \min_{1\leq k\leq K} L_T^k$.

Lemma 7. *Let α and β be positive constants, and let $\tau \in \mathbb{Z}^+$. Suppose that for $t = \tau, \tau+1, \ldots, T$ there exists a single action k^* that achieves minimal cumulative loss $L_t^{k^*} = L_t^*$, and for $k \neq k^*$ the cumulative losses diverge as $L_t^k - L_t^* \geq \alpha t^\beta$. Then for all $\eta > 0$*

$$\sum_{t=\tau}^{T}\big(1-w_{t+1}^*(\eta)\big) \leq C_K\,\eta^{-1/\beta},$$

where $C_K = (K-1)\alpha^{-1/\beta}\Gamma(1+\frac{1}{\beta})$ is a constant that does not depend on η, τ or T.

The lemma is proved in the Additional Material. Together with Lemmas 1 and 6, it gives an upper bound on $\Delta_T(\eta)$, which may be used to bound the number of segments started by AdaHedge. This leads to the following result, whose proof is also delegated to the Additional Material.

Let $s(m)$ denote the round in which AdaHedge starts its m-th segment, and let $L_r^k(m) = L_{s(m)+r-1}^k - L_{s(m)-1}^k$ denote the cumulative loss of action k in that segment.

Lemma 8. *Let $\alpha > 0$ and $\beta > 1/2$ be constants, and let C_K be as in Lemma 7. Suppose there exists a segment $m^* \in \mathbb{Z}^+$ started by AdaHedge, such that $\tau := \lfloor 8\ln(K)\phi^{(m^*-1)(2-1/\beta)} - 8(e-2)C_K + 1\rfloor \geq 1$ and for some action k^* the cumulative losses in segment m^* diverge as*

$$L_r^k(m^*) - L_r^{k^*}(m^*) \geq \alpha r^\beta \qquad \textit{for all } r \geq \tau \textit{ and } k \neq k^*. \tag{6}$$

Then AdaHedge starts at most m^* segments, and hence by Lemma 4 its regret is bounded by a constant:

$$R_{\textit{AdaHedge}}(T) = O(1).$$

In the simplistic example from the introduction, we may take $\alpha = b-a-2\epsilon$ and $\beta = 1$, such that (6) is satisfied for any $\tau \geq 1$. Taking m^* large enough to ensure that $\tau \geq 1$, we find that AdaHedge never starts more than $m^* = 1 + \lceil \log_\phi(\frac{e-2}{\alpha\ln(2)} + \frac{1}{8\ln(2)})\rceil$ segments. Let us also give an example of a probabilistic setting in which Lemma 8 applies:

Theorem 9. *Let $\alpha > 0$ and $\delta \in (0, 1]$ be constants, and let k^* be a fixed action. Suppose the loss vectors $\boldsymbol{\ell}_t$ are independent random variables such that the expected differences in loss satisfy*

$$\min_{k \neq k^*} \mathbb{E}[\ell_t^k - \ell_t^{k^*}] \geq 2\alpha \qquad \textit{for all } t \in \mathbb{Z}^+. \tag{7}$$

Then, with probability at least $1 - \delta$, AdaHedge starts at most

$$m^* = 1 + \left\lceil \log_\phi \Big(\frac{(K-1)(e-2)}{\alpha \ln(K)} + \frac{\ln\big(2K/(\alpha^2\delta)\big)}{4\alpha^2 \ln(K)} + \frac{1}{8\ln(K)} \Big) \right\rceil \tag{8}$$

segments and consequently its regret is bounded by a constant:

$$R_{\textit{AdaHedge}}(T) = O\big(K + \log(1/\delta)\big).$$

This shows that the probabilistic setting of the theorem is much easier than the worst case, for which only a bound on the regret of order $O(\sqrt{T \ln(K)})$ is possible, and that AdaHedge automatically adapts to this easier setting. The proof of Theorem 9 is in the Additional Material. It verifies that the conditions of Lemma 8 hold with sufficient probability for $\beta = 1$, and α and m^* as in the theorem.

5 Experiments

We compare AdaHedge to other hedging algorithms in two experiments involving simulated losses.

5.1 Hedging Algorithms

Follow-the-Leader. This algorithm is included because it is simple and very effective if the losses are not antagonistic, although as mentioned in the introduction its regret is linear in the worst case.

Hedge with fixed learning rate. We also include Hedge with a fixed learning rate

$$\eta = \sqrt{2\ln(K)/L_T^*}, \tag{9}$$

which achieves the regret bound $\sqrt{2\ln(K)L_T^*} + \ln(K)$[1]. Since η is a function of L_T^*, the agent needs to use post-hoc knowledge to use this strategy.

Hedge with doubling trick. The common way to apply the doubling trick to L_T^* is to set a budget on L_T^* and multiply it by some constant ϕ' at the start of each new segment, after which η is optimized for the new budget [4, 7]. Instead, we proceed the other way around and with each new segment first divide η by $\phi = 2$ and then calculate the new budget such that (9) holds when $\Delta_t(\eta)$ reaches the budget. This way we keep the same invariant (η is never larger than the right-hand side of (9), with equality when the budget is depleted), and the frequency of doubling remains logarithmic in L_T^* with a constant determined by ϕ, so both approaches are equally valid. However, controlling the sequence of values of η allows for easier comparison to AdaHedge.

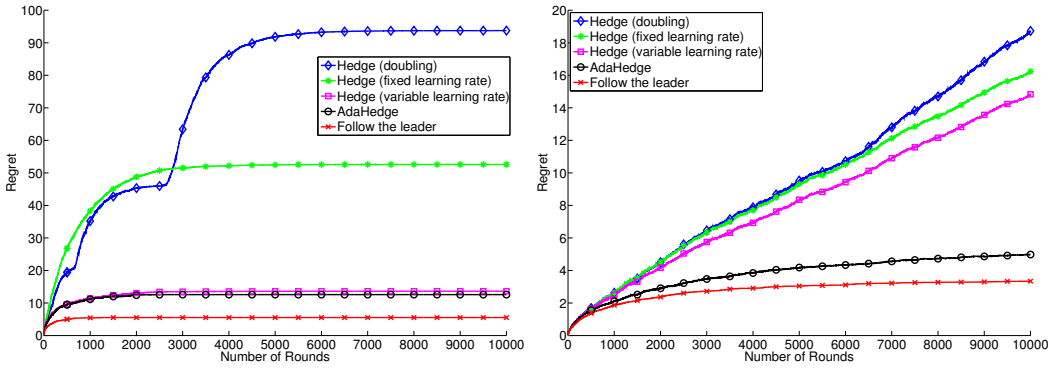
AdaHedge (Algorithm 1). Like in the previous algorithm, we set $\phi = 2$. Because of how we set up the doubling, both algorithms now use the same sequence of learning rates $1, 1/2, 1/4, \ldots$; the only difference is *when* they decide to start a new segment.

Hedge with variable learning rate. Rather than using the doubling trick, this algorithm, described in [8], changes the learning rate each round as a function of L_t^*. This way there is no need to relearn the weights of the actions in each block, which leads to a better worst-case bound and potentially better performance in practice. Its behaviour on easy problems, as we are currently interested in, has not been studied.

5.2 Generating the Losses

In both experiments we choose losses in $\{0, 1\}$. The experiments are set up as follows.

[1] Cesa-Bianchi and Lugosi use $\eta = \ln(1 + \sqrt{2\ln K / L_T^*})$ [4], but the same bound can be obtained for the simplified expression we use.



(a) I.I.D. losses (b) Correlated losses

Figure 1: Simulation results

I.I.D. losses. In the first experiment, all $T = 10\,000$ losses for all $K = 4$ actions are independent, with distribution depending only on the action: the probabilities of incurring loss 1 are 0.35, 0.4, 0.45 and 0.5, respectively. The results are then averaged over 50 repetitions of the experiment.

Correlated losses. In the second experiment, the $T = 10\,000$ loss vectors are still independent, but no longer identically distributed. In addition there are dependencies *within* the loss vectors $\boldsymbol{\ell}_t$, between the losses for the $K = 2$ available actions: each round is *hard* with probability 0.3, and *easy* otherwise. If round t is hard, then action 1 yields loss 1 with probability $1 - 0.01/t$ and action 2 yields loss 1 with probability $1 - 0.02/t$. If the round is easy, then the probabilities are flipped and the actions yield loss 0 with the same probabilities. The results are averaged over 200 repetitions.

5.3 Discussion and Results

Figure 1 shows the results of the experiments above. We plot the regret (averaged over repetitions of the experiment) as a function of the number of rounds, for each of the considered algorithms.

I.I.D. Losses. In the first considered regime, the accumulated losses for each action diverge linearly with high probability, so that the regret of Follow-the-Leader is bounded. Based on Theorem 9 we expect AdaHedge to incur bounded regret also; this is confirmed in Figure 1(a). Hedge with a fixed learning rate shows much larger regret. This happens because the learning rate, while it optimizes the worst-case bound, is much too small for this easy regime. In fact, if we would include more rounds, the learning rate would be set to an even smaller value, clearly showing the need to determine the learning rate adaptively. The doubling trick provides one way to adapt the learning rate; indeed, we observe that the regret of Hedge with the doubling trick is initially smaller than the regret of Hedge with fixed learning rate. However, unlike AdaHedge, the algorithm never detects that its current value of η is working well; instead it keeps exhausting its budget, which leads to a sequence of clearly visible bumps in its regret. Finally, it appears that the Hedge algorithm with variable learning rate also achieves bounded regret. This is surprising, as the existing theory for this algorithm only considers its worst-case behaviour, and the algorithm was not designed to do specifically well in easy regimes.

Correlated Losses. In the second simulation we investigate the case where the mean cumulative loss of two actions is extremely close — within $O(\log t)$ of one another. If the losses of the actions where independent, such a small difference would be dwarfed by random fluctuations in the cumulative losses, which would be of order $O(\sqrt{t})$. Thus the two actions can only be distinguished because we have made their losses dependent. Depending on the application, this may actually be a more natural scenario than complete independence as in the first simulation; for example, we can think of the losses as mistakes of two binary classifiers, say, two naive Bayes classifiers with different smoothing parameters. In such a scenario, losses will be dependent, and the difference in cumulative loss will be much smaller than $O(\sqrt{t})$. In the previous experiment, the posterior weights of the actions

converged relatively quickly for a large range of learning rates, so that the exact value of the learning rate was most important at the start (e.g., from 3000 rounds onward Hedge with fixed learning rate does not incur much additional regret any more). In this second setting, using a high learning rate remains important throughout. This explains why in this case Hedge with variable learning rate can no longer keep up with Follow-the-Leader. The results for AdaHedge are also interesting: although Theorem 9 does not apply in this case, we may still hope that $\Delta_t(\eta)$ grows slowly enough that the algorithm does not start too many segments. This turns out to be the case: over the 200 repetitions of the experiment, AdaHedge started only 2.265 segments on average, which explains its excellent performance in this simulation.

6 Proof of Lemma 6

Our main technical tool is Lemma 6. Its proof requires the following intermediate result:

Lemma 10. *For any $\eta > 0$ and any time t, the function $f(\boldsymbol{\ell}_t) = \ln\left(\boldsymbol{w}_t \cdot e^{-\eta \boldsymbol{\ell}_t}\right)$ is convex.*

This may be proved by observing that f is the convex conjugate of the Kullback-Leibler divergence. An alternative proof based on log-convexity is provided in the Additional Material.

Proof of Lemma 6. We need to bound $\delta_t = \boldsymbol{w}_t(\eta) \cdot \boldsymbol{\ell}_t + \frac{1}{\eta}\ln(\boldsymbol{w}_t(\eta) \cdot e^{-\eta \boldsymbol{\ell}_t})$, which is a convex function of $\boldsymbol{\ell}_t$ by Lemma 10. As a consequence, its maximum is achieved when $\boldsymbol{\ell}_t$ lies on the boundary of its domain, such that the losses ℓ_t^k are either 0 or 1 for all k, and in the remainder of the proof we will assume (without loss of generality) that this is the case. Now let $\alpha_t = \boldsymbol{w}_t \cdot \boldsymbol{\ell}_t$ be the posterior probability of the actions with loss 1. Then

$$\delta_t = \alpha_t + \frac{1}{\eta}\ln\left((1-\alpha_t) + \alpha_t e^{-\eta}\right) = \alpha_t + \frac{1}{\eta}\ln\left(1 + \alpha_t(e^{-\eta} - 1)\right).$$

Using $\ln x \leq x - 1$ and $e^{-\eta} \leq 1 - \eta + \frac{1}{2}\eta^2$, we get $\delta_t \leq \frac{1}{2}\alpha_t\eta$, which is tight for α_t near 0. For α_t near 1, rewrite

$$\delta_t = \alpha_t - 1 + \frac{1}{\eta}\ln(e^{\eta}(1-\alpha_t) + \alpha_t)$$

and use $\ln x \leq x - 1$ and $e^{\eta} \leq 1 + \eta + (e-2)\eta^2$ for $\eta \leq 1$ to obtain $\delta_t \leq (e-2)(1-\alpha_t)\eta$. Combining the bounds, we find

$$\delta_t \leq (e-2)\eta \min\{\alpha_t, 1-\alpha_t\}.$$

Now, let k^* be an action such that $w_t^* = w_t^{k^*}$. Then $\ell_t^{k^*} = 0$ implies $\alpha_t \leq 1 - w_t^*$. On the other hand, if $\ell_t^{k^*} = 1$, then $\alpha_t \geq w_t^*$ so $1-\alpha_t \leq 1-w_t^*$. Hence, in both cases $\min\{\alpha_t, 1-\alpha_t\} \leq 1-w_t^*$, which completes the proof. □

7 Conclusion and Future Work

We have presented a new algorithm, AdaHedge, that adapts to the difficulty of the DTOL learning problem. This difficulty was characterised in terms of convergence of the posterior probability of the best action. For hard instances of DTOL, for which the posterior does not converge, it was shown that the regret of AdaHedge is of the optimal order $O(\sqrt{L_T^* \ln(K)})$; for easy instances, for which the posterior converges sufficiently fast, the regret was bounded by a constant. This behaviour was confirmed in a simulation study, where the algorithm outperformed existing versions of Hedge.

A surprising observation in the experiments was the good performance of Hedge with a variable learning rate on some easy instances. It would be interesting to obtain matching theoretical guarantees, like those presented here for AdaHedge. A starting point might be to consider how fast the posterior probability of the best action converges to one, and plug that into Lemma 6.

Acknowledgments

The authors would like to thank Wojciech Kotłowski for useful discussions. This work was supported in part by the IST Programme of the European Community, under the PASCAL2 Network of Excellence, IST-2007-216886, and by NWO Rubicon grant 680-50-1010. This publication only reflects the authors' views.

References

[1] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55:119–139, 1997.

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

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

[4] N. Cesa-Bianchi and G. Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006.

[5] Y. Freund and R. E. Schapire. Adaptive game playing using multiplicative weights. *Games and Economic Behavior*, 29:79–103, 1999.

[6] E. Hazan and S. Kale. Extracting certainty from uncertainty: Regret bounded by variation in costs. In *Proceedings of the 21st Annual Conference on Learning Theory (COLT)*, pages 57–67, 2008.

[7] 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.

[8] P. Auer, N. Cesa-Bianchi, and C. Gentile. Adaptive and self-confident on-line learning algorithms. *Journal of Computer and System Sciences*, 64:48–75, 2002.

[9] V. Vovk. Competitive on-line statistics. *International Statistical Review*, 69(2):213–248, 2001.

[10] D. Haussler, J. Kivinen, and M. K. Warmuth. Sequential prediction of individual sequences under general loss functions. *IEEE Transactions on Information Theory*, 44(5):1906–1925, 1998.

[11] A. N. Shiryaev. *Probability*. Springer-Verlag, 1996.