Risk-Sensitive Online Learning

Eyal Even-Dar, Michael Kearns, and Jennifer Wortman

Department of Computer and Information Science
University of Pennsylvania, Philadelphia, PA 19104

Abstract. We consider the problem of online learning in settings in which we want to compete not simply with the rewards of the best expert or stock, but with the best trade-off between rewards and *risk*. Motivated by finance applications, we consider two common measures balancing returns and risk: the *Sharpe ratio* [11] and the *mean-variance* criterion of Markowitz [10]. We first provide negative results establishing the impossibility of no-regret algorithms under these measures, thus providing a stark contrast with the returns-only setting. We then show that the recent algorithm of Cesa-Bianchi et al. [5] achieves nontrivial performance under a modified bicriteria risk-return measure, and give a modified best expert algorithm that achieves no regret for a "localized" version of the mean-variance criterion. We perform experimental comparisons of traditional online algorithms and the new risk-sensitive algorithms on a recent six-year S&P 500 data set. To our knowledge this paper initiates the investigation of explicit risk considerations in the standard models of worst-case online learning.

1 Introduction

Despite the large literature on online learning and the rich collection of algorithms with guaranteed worst-case regret bounds, virtually no attention has been given to the risk (as measured by the volatility in returns or profits) incurred by such algorithms. Partial exceptions are the recent work of Cesa-Bianchi et al. [5] which we analyze in our framework, and the work of Warmuth and Kuzmin [12] which assumes that a covariance matrix is revealed at each time step and focuses on minimizing only risk, ignoring returns. Especially in finance-related applications [8], where consideration of various measures of the volatility of a portfolio are often given equal footing with the returns themselves, this omission is particularly glaring.

It is natural to ask why one would like explicit consideration of volatility or risk in online learning given that we are already blessed with algorithms providing performance guarantees that track various benchmarks (e.g. best single stock or expert) with absolute certainty. However, in many natural circumstances the benchmark may not be sufficiently strong (e.g. tracking the best stock, as opposed to a richer class of strategies) or the guarantees may be sufficiently loose that realistic application of the existing online algorithms will require one to incorporate additional, more traditional, risk criteria. For example, if one

applies the standard EG portfolio management algorithm [8] to the S&P 500 over a recent six year period, its returns are actually worse (for all positive learning rates) than that of the simple uniform constant rebalanced portfolio (UCRP), despite the theoretical guarantees on EG performance (see Section 7). Thus for a variety of reasons, we are motivated to find algorithms that can enjoy guarantees similar to those of the "traditional" approaches such as EG, but that deliberately incorporate risk-return trade-offs. More generally, since such trade-offs are an inherent part of the way Wall Street and the finance community view investment performance, it is interesting to consider online learning through the same lens.

The finance literature on balancing risk and return, and the proposed metrics for doing so, are far too large to survey here (see [2], chapter 4 for a nice overview). Among the most common methods are the *Sharpe ratio* [11], and the *mean-variance (MV)* criterion of which Markowitz was the first proponent [10]. Let $r_t \in [-1, \infty)$ be the return of any given financial instrument (a stock, bond, portfolio, trading strategy, etc.) during time period t. That is, if v_t represents the dollar value of the instrument immediately after period t, we have $v_t = (1 + r_t)v_{t-1}$. Negative values of r_t (down to -1, representing the limiting case of the instrument losing all of its value) are losses, and positive values are gains. For a sequence of returns $\boldsymbol{r} = (r_1, \ldots, r_T)$, suppose $\mu(\boldsymbol{r})$ denotes the (arithmetic) mean and $\sigma(\boldsymbol{r})$ denotes the standard deviation. Then the Sharpe ratio of the instrument on the sequence is simply $\mu(\boldsymbol{r})/\sigma(\boldsymbol{r})$,[1] while the MV is $\mu(\boldsymbol{r}) - \sigma(\boldsymbol{r})$. (The term mean-variance is slightly misleading since the risk is actually measured by the standard deviation, but we use it to adhere to convention.)

A common alternative is to use the mean and standard deviation not of the r_t but of the $\log(1 + r_t)$, which corresponds to geometric rather than arithmetic averaging of returns; we shall refer to the resulting measures as the *geometric* Sharpe ratio and MV. Note that when r_t is close to 0, (as it is generally in finance applications) it is a good approximation of $\log(1 + r_t)$, so maximizing the arithmetic Sharpe ratio or MV is approximately equivalent to maximizing their geometric counterparts. Although it is tempting to claim that $\log(1 + r_t)$ approximates r_t minus the variance of r_t, it actually approximates $r_t - r_t^2/2$, which can be quite different even in financial applications.

Both the Sharpe ratio and the MV are natural, if somewhat different, methods for specifying a trade-off between the risk and returns of a financial instrument. Note that if we have an algorithm (like Weighted Majority [9, 4]) that maintains a dynamically weighted and rebalanced portfolio over K constituent stocks, this algorithm itself has a sequence of returns and thus its own Sharpe ratio and MV. A natural hope for online learning would be to replicate the kind of no-regret results to which we have become accustomed, but for regret in these risk-return measures. Thus (for example) we would like an algorithm

[1] The original definition of the Sharpe ratio also considers the return of a risk-free investment. This term can be safely ignored in analysis if we view returns as already having been shifted by the rate of the risk-free investment.

whose Sharpe ratio or MV at sufficiently long time scales is arbitrarily close to the *best* Sharpe ratio or MV of any of the K stocks. The prospects for these and similar results are the topic of this paper.

Our first results are negative, and show that the specific hope articulated in the last paragraph is unattainable. More precisely, we show that for either the Sharpe ratio or MV, any online learning algorithm must suffer *constant* regret, even when $K = 2$. This is in sharp contrast to the literature on returns alone, where it is known that zero regret can be approached rapidly with increasing time. Furthermore, and perhaps surprisingly, for the case of the Sharpe ratio the proof shows that constant regret is inevitable even for an *offline* algorithm (which knows in advance the specific sequence of returns for the two stocks, but still must compete with the best Sharpe ratio on all time scales).

The fundamental insight in these impossibility results is that the risk term in the different risk-return metrics introduces a "switching cost" not present in the standard return-only settings. Intuitively, in the return-only setting, no matter what decisions an algorithm has made up to time t, it can choose (for instance) to move all of its capital to one stock at time t and *immediately* begin enjoying the *same* returns as that stock from that time forward. However, under the risk-return metrics, if the returns of the algorithm up to time t have been quite different (either higher or lower) than those of the stock, the algorithm pays a "volatility penalty" not suffered by the stock itself.

These strong impossibility results force us to revise our expectations for online learning for risk-return settings. In the second part of the paper, we examine two different approaches to algorithms for MV-like metrics. First we analyze the recent algorithm of Cesa-Bianchi et al. [5] and show that it exhibits a trade-off balancing returns with variance (as opposed to standard deviation) that is additively comparable to a trade-off exhibited by the best stock. This approximation is weaker than competitive ratio or no-regret, but remains nontrivial, especially in light of the strong negative results mentioned above. In the second approach, we give a general transformation of the instantaneous gains given to algorithms (such as Weighted Majority) meeting standard returns-only no-regret criteria. This transformation permits us to incorporate a recent moving window of variance into the gains, yielding an algorithm competitive with a "localized" version of MV in which we are penalized only for volatility on short time scales.

In Section 7 we show the results of an experimental comparison of traditional online algorithms with the risk-sensitive algorithms mentioned above on a six-year S&P 500 data set.

2 Preliminaries

We denote the set of experts as integers $\mathcal{K} = \{1, \ldots, K\}$. For each expert $k \in \mathcal{K}$, we denote its *reward* at time $t \in \{1, \ldots, T\}$ as x_t^k. At each time step t, an algorithm A assigns a weight $w_t^k \geq 0$ to each expert k such that $\sum_{k=1}^{K} w_t^k = 1$. Based on these weights, the algorithm then receives a reward $x_t^A = \sum_{k=1}^{K} w_t^k x_t^k$.

There are multiple ways to define the aforementioned rewards. In a financial setting it is common to define them to be the *simple returns* of some underlying investment. Thus if v_t represents the dollar value of an investment following period t, and $v_t = (1 + r_t)v_{t-1}$ where $r_t \in [-1, \infty)$, one choice is to let $x_t = r_t$. When r_t is close to 0, it is also a good approximation of $\log(1 + r_t)$, so maximizing the arithmetic average of rewards will be very close to maximizing the geometric average. We assume that daily rewards lie in the range $[-M, M]$ for some constant M; some of our bounds depend on M.

Two well-known measures of volatility that we will refer to often are variance and standard deviation. Formally, if $\bar{R}_t(k, \boldsymbol{x})$ is the average reward of expert k on the reward sequence $\boldsymbol{x}$ at time t, then

$$Var_t(k, \boldsymbol{x}) = \frac{1}{t} \sum_{t'=1}^{t} (x_{t'}^k - \bar{R}_t(k, \boldsymbol{x}))^2, \qquad \sigma_t(k, \boldsymbol{x}) = \sqrt{Var_t(k, \boldsymbol{x})}$$

We define $R_t(k, \boldsymbol{x})$ to be the sum of rewards of expert k at times $1, \ldots, t$.

Traditionally in online learning the objective of an algorithm A has been to achieve an average reward at least as good as the best expert over time, yielding results of the form

$$\bar{R}_T(A, \boldsymbol{x}) = \sum_{t=1}^{T} \frac{x_t^A}{T} \geq \max_{k \in \mathcal{K}} \sum_{t=1}^{T} \frac{x_t^k}{T} - \sqrt{\frac{\log K}{T}} = \max_{k \in \mathcal{K}} \bar{R}_T(k, \boldsymbol{x}) - \sqrt{\frac{\log K}{T}}$$

An algorithm achieving this goal is often referred to as a "no regret" algorithm.

Now we are ready to define two standard risk-reward balancing criteria, the Sharpe ratio [11] and the MV of expert k at time t.

$$Sharpe_t(k, \boldsymbol{x}) = \frac{\bar{R}_t(k, \boldsymbol{x})}{\sigma_t(k, \boldsymbol{x})}, \qquad MV_t(k, \boldsymbol{x}) = \bar{R}_t(k, \boldsymbol{x}) - \sigma_t(k, \boldsymbol{x})$$

In the following definitions we use the MV, but all apply mutatis mutandis to Sharpe ratio. We say that an algorithm has *no regret* with respect to MV if

$$MV_t(A, \boldsymbol{x}) \geq \max_{k \in \mathcal{K}} MV_t(k, \boldsymbol{x}) - Regret(t)$$

where $Regret(t)$ is a function that goes to 0 as t approaches infinity. We say that an algorithm A has *constant regret* C for some constant $C > 0$ (that does not depend on time but may depend on M) if for any large t there exists a $t' \geq t$ and a sequence $\boldsymbol{x}$ of expert rewards for which the following is satisfied:

$$MV_{t'}(A, \boldsymbol{x}) > \max_{k \in \mathcal{K}} MV_{t'}(k, \boldsymbol{x}) - C$$

Finally, the *competitive ratio* of an algorithm A is defined as

$$\inf_{\boldsymbol{x}} \inf_{t} \frac{MV_t(A, \boldsymbol{x})}{\max_{k \in \mathcal{K}} MV_t(k, \boldsymbol{x})}$$

where $\boldsymbol{x}$ can be any reward sequence generated for K experts. We sometimes refer to $MV_t(A, \boldsymbol{x}) / \max_{k \in \mathcal{K}} MV_t(k, \boldsymbol{x})$ as the *competitive ratio on* $\boldsymbol{x}$ *at time* t.

Note that for negative concepts like constant regret, it is sufficient to consider a single sequence of expert rewards for which *no* algorithm can perform well.

3 A Lower Bound for the Sharpe Ratio

In this section we show that even an offline policy cannot compete with the best expert with respect to the Sharpe ratio, even when there are only two experts.

Theorem 1. *Any offline algorithm has constant regret with respect to Sharpe ratio. Furthermore, for any $T \geq 30$, there exists an expert reward sequence $\boldsymbol{x}$ of length T and two points in time such that no algorithm can attain more than a $1 - C$ competitive ratio on $\boldsymbol{x}$ at both points, for some constant $C > 0$.*

We give a brief overview the proof here; details are provided in Appendix A.

The lower bound is proved in a setting where there are only two experts and the performance of the algorithm is tested at only two points. The reward sequence used is simple with each expert's reward changing only twice. The performance of the algorithm is tested when the second change occurs and at the end of the sequence. If the algorithm reward at the first checkpoint is too high, it will be the case that the competitive ratio at that point is bad. If it is lower, the competitive ratio at the second checkpoint will be bad. We characterize the optimal offline algorithm and show that it cannot compete with the best stock on this sequence. This, of course, implies that no algorithm can compete.

4 A Lower Bound for MV

A similar bound can be shown for our additive risk-reward measure, the MV.

Theorem 2. *Any online algorithm has constant regret with respect to the MV.*

The proof will again be based on specific sequences that will serve as counterexamples to show that in general it is not possible to compete with the best expert in terms of the MV. We begin by describing how these sequences are generated. Again we consider a scenario in which there are only two experts. For the first n time steps, the first expert receives at each time step a reward of 2 with probability $1/2$ or a reward of 0 with probability $1/2$, while at times $n + 1, ..., 2n$ the reward is always 1. The second expert's reward is always $1/4$ throughout the entire sequence. The algorithm's performance will be tested only at times n and $2n$, and the algorithm is assumed to know the process by which these expert rewards are generated.

This lower bound construction is not a single sequence but is a set of sequences generated according to the distribution over the first expert's rewards. We will refer to the set of all sequences that can be generated by this distribution as S. For any specific sequence in S, the optimal offline algorithm would suffer no regret, but we will show by the probabilistic method that there is no online algorithm that can perform well on all sequences in S at both checkpoints. In contrast to "standard" experts, there are now two randomness sources: the internal randomness of the algorithm and the randomness of the rewards.

We now give a high level overview. First we will consider a "balanced sequence" in S in which expert 1 receives an equal number of rewards that are

2 and rewards that are 0. Assuming such a sequence, it will be the case that the best expert at time n is expert 2 with reward 1/4 and standard deviation 0, while the best expert at time $2n$ is expert 1 with reward 1 and standard deviation $1/\sqrt{2}$. Note that any algorithm that has average reward 1/4 at time n in this scenario will be unable to overcome this start and will have a constant regret at time $2n$. Yet it might be the case on such sequences that a sophisticated adaptive algorithm could have an average reward higher than 1/4 at time n and still suffer no regret at time n. Hence, for the balanced sequence we first look at the case in which the *algorithm* is "balanced" as well, i.e. the weight it puts on expert 1 on days with reward 2 is equal to the weight it puts on expert 1 on days with reward 0. We can later drop this requirement.

In our analysis we show that most sequences in S are "close" to the balanced sequence. If the average reward of an algorithm over all sequences is less than $1/4 + \delta$, for some constant δ, then by the probabilistic method there exists a sequence for which the algorithm will have constant regret at time $2n$. If not, then there exists a sequence for which at time n the algorithm's standard deviation will be larger than δ by some constant factor, so the algorithm will have regret at time n. This argument will also be probabilistic, preventing the algorithm from constantly being "lucky." Details of this proof are given in Appendix B.

In fact we can extend this theorem to the broader class of objective functions of the form $\bar{R}_t(k, \boldsymbol{x}) - \alpha\sigma_t(A, \boldsymbol{x})$, where $\alpha > 0$ is constant. The proof, which is similar to the proof of Theorem 2, is omitted due to space limits. Both the constant and the length of the sequence will depend on α.

Theorem 3. *Let $\alpha \geq 0$ be a constant. The regret of any online algorithm with respect to the metric $\bar{R}_t(k, \boldsymbol{x}) - \alpha\sigma_t(A, \boldsymbol{x})$ is constant for some positive constant that depends on α.*

5 A Bicriteria Upper Bound

In this section we show that the recent algorithm of Cesa-Bianchi et al. [5] can yield a risk-reward balancing bound. Their original result expressed a no-regret bound with respect to *rewards* only, but the regret itself involved a variance term. Here we give an alternate analysis demonstrating that the algorithm actually respects a risk-reward trade-off. The quality of the results here depends on the bound M on the absolute value of expert rewards as we will show.

We first describe the algorithm Prod which takes one parameter η. It maintains a set of K weights, one for each expert. The (unnormalized) weights $\tilde{w}_t^k$ are initialized with $\tilde{w}_1^k = 1$ for every expert k and updated at each time step according to $\tilde{w}_t^k \leftarrow \tilde{w}_{t-1}^k(1 + \eta x_{t-1}^k)$. The normalized weights at each time step are then defined as $w_t^k = \tilde{w}_t^k / \tilde{W}_t$ where $\tilde{W}_t = \sum_{j=1}^k \tilde{w}_t^j$.

Theorem 4. *For any expert $k \in \mathcal{K}$, for the algorithm Prod with $\eta = 1/(LM)$ where $L > 2$ we have at time t*

$$\left(\frac{L\bar{R}_t(A, \boldsymbol{x})}{L-1} - \frac{\eta(3L-2)Var_t(A, \boldsymbol{x})}{6L}\right) \geq \left(\frac{L\bar{R}_t(k, \boldsymbol{x})}{L+1} - \frac{\eta(3L+2)Var_t(k, \boldsymbol{x})}{6L}\right) - \frac{\ln K}{\eta t}$$

for any sequence $\boldsymbol{x}$ in which the absolute value of each reward is bounded by M.

The proof is given in Appendix C. The two large expressions in parentheses in Theorem 4 additively balance rewards and variance of rewards, but with different coefficients. It is tempting but apparently not possible to convert this inequality into a competitive ratio. Nevertheless certain natural settings of the parameters cause the two expressions to give quantitatively similar trade-offs. For example, let $\boldsymbol{x}$ be any sequence of rewards which are bounded in $[-0.1, 0.1]$ and let A be Prod for $\eta = 1$. Then for any time t and expert k we have

$$1.11\bar{R}_t(A, \boldsymbol{x}) - 0.466Var_t(A, \boldsymbol{x}) \geq 0.91\bar{R}_t(k, \boldsymbol{x}) - 0.533Var_t(k, \boldsymbol{x}) - (10 \ln K)/t$$

This gives a relatively even balance between rewards and variance on both sides. We note that the choice of a "reasonable" bound on the rewards magnitudes should be related to the time scale of the process — for instance, returns on the order of $\pm 1\%$ might be entirely reasonable daily but not annually.

6 No-Regret Results for Localized Risk

We now show a no-regret result for an algorithm optimizing an alternative objective function that incorporates both risk and reward. The primary leverage of this alternative objective is that risk is now measured only "locally." The goal is to balance immediate rewards with how far these immediate rewards deviate from the average rewards over some "recent" past. In addition to allowing us to skirt the strong impossibility results for no-regret in the standard risk-return measures, we note that our new objective may be of independent interest, as it incorporates other notions of risk that are commonly considered in finance where short-term volatility is usually of greater concern than long-term. For example, this objective has the flavor of what is sometimes called "maximum draw-down," the largest decline in the price of a stock over a given, usually short, time period.

Consider the following risk measure for an expert k on a reward sequence $\boldsymbol{x}$:

$$P_t(k, \boldsymbol{x}) = \sum_{t'=2}^{t} (x_{t'}^k - \text{AVG}_\ell^*(x_1^k, ..., x_{t'}^k))^2$$

where $\text{AVG}_\ell^*(x_1^k, .., x_t^k) = \sum_{t'=t-\ell+1}^{t} (x_{t'}^k/\ell)$ is the fixed window size average for some window size $\ell > 0$. The new risk-sensitive criterion at time t will be

$$G_t(A, \boldsymbol{x}) = \bar{R}_t(A, \boldsymbol{x}) - P_t(A, \boldsymbol{x})/t$$

. Observe that the measure of risk defined here is very similar to variance. In particular, if for every expert $k \in \mathcal{K}$ we let $p_t^k = (x_t^k - \text{AVG}_t^*(x_1^k, .., x_t^k))^2$, then

$$P_t(k, \boldsymbol{x})/t = \sum_{t'=2}^{t} \frac{p_{t'}^k}{t}, \qquad Var_t(k, \boldsymbol{x}) = \sum_{t'=2}^{n} \frac{p_{t'}^k}{t} \left(1 + \frac{1}{t'-1}\right)$$

Our measure differs from the variance in two aspects. The variance of the sequence will be affected by rewards in the past and the future, whereas our measure depends only on rewards in the past, and for our measure the current reward

is compared only to the rewards in the recent past, and not to all past rewards. While both differences are exploited in the proof, the fixed window size is key. The main obstacle of the algorithms in the previous sections was the "memory" of the variance, which prevented switching between experts. The memory of the penalty is now ℓ and our results will be meaningful when $\ell = o(\sqrt{T})$.

The algorithm we discuss will work by feeding modified instantaneous gains to any best experts algorithm that satisfies the assumption below. This assumption is met by algorithms such as Weighted Majority [9, 4].

Definition 1. *An optimized best expert algorithm is an algorithm that guarantees that for any sequence of reward vectors $\boldsymbol{x}$ over experts $\mathcal{K} = \{1, \ldots, K\}$, the algorithm selects a distribution $\boldsymbol{w_t}$ over $\mathcal{K}$ (using only the previous reward functions) such that*

$$\sum_{t=1}^{T}\sum_{k=1}^{K} w_t^k x_t^k \geq \sum_{t=1}^{T} x_t^k - \sqrt{TM\log K},$$

where $|x_t^k| \leq M$ and k is any expert. Furthermore, we also assume that decision distributions do not change quickly: $\|\boldsymbol{w_t} - \boldsymbol{w_{t+1}}\|_1 \leq \sqrt{\log(K)/t}$.

Since the risk function now has shorter memory, there is hope that a standard best expert algorithm will work. Therefore, we would like to incorporate this risk term into the instantaneous rewards fed to the best experts algorithm. We will define this instantaneous quantity, the *gain* of expert k at time t to be $g_t^k = x_t^k - (x_t^k - AVG_\ell^*(x_1^k, ..., x_{t-1}^k))^2 = x_t^k - p_t^k$, where p_t^k is the *penalty* for expert k at time t. Similarly the penalty for an algorithm A can be defined as $p_t^A = (x_t^A - AVG_\ell^*(x_1^A, ..., x_{t-1}^A))^2$. It is natural to wonder whether $p_t^A = \sum_{k=1}^K w_t^k p_t^k$; unfortunately, this is not the case, but they are similar. To formalize the connection between the measures, we let $\hat{P}(A, \boldsymbol{x}) = \sum_{t=1}^T \sum_{k=1}^K w_t^k p_t^k$ be the weighted penalty function of the experts, and $P(A, \boldsymbol{x}) = \sum_{t=1}^T p_t^A$ be the penalty function observed by the algorithm. The next lemma relates these quantities.

Lemma 1. *Let $\boldsymbol{x}$ be any reward sequence such that all rewards are bounded by M. Then $\hat{P}^T(A, \boldsymbol{x}) \geq P_T(A, \boldsymbol{x}) - 2TM^2\ell\sqrt{\frac{\log K}{T-\ell}}$.*

Proof:

$$\hat{P}^T(A, \boldsymbol{x}) = \sum_{t=1}^{T}\sum_{k=1}^{K} w_t^k (x_t^k - AVG_\ell^*(x_1^k, .., x_t^k))^2$$

$$\geq \sum_{t=1}^{T}\left(\sum_{k=1}^{K} w_t^k \left(x_t^k - \frac{\sum_{j=1}^{\ell} x_{t-j+1}^k}{\ell}\right)\right)^2$$

$$= \sum_{t=1}^{T}\left(\sum_{k=1}^{K} w_t^k x_t^k - \frac{\sum_{k=1}^K \sum_{j=1}^{\ell}(w_t^k - w_{t-j+1}^k + w_{t-j+1}^k)x_{t-j+1}^k}{\ell}\right)^2$$

$$= \sum_{t=1}^{T}\left(\left(\sum_{k=1}^{K} w_t^k x_t^k - \frac{\sum_{k=1}^K \sum_{j=1}^{\ell} w_{t-j+1}^k x_{t-j+1}^k}{\ell}\right)^2 + \left(\frac{\sum_{k=1}^K \sum_{j=1}^{\ell} \epsilon_j^k x_{t-j+1}^k}{\ell}\right)^2\right.$$

$$-2\left(\frac{\sum_{k=1}^{K}\sum_{j=1}^{\ell}\epsilon_j^k x_{t-j+1}^k}{\ell}\right)\left(\sum_{k=1}^{K} w_t^k x_t^k - \frac{\sum_{k=1}^{K}\sum_{j=1}^{\ell} w_{t-j+1}^k x_{t-j+1}^k}{\ell}\right)\Bigg)$$

$$\geq P_T(A,\boldsymbol{x}) - \sum_{t=1}^{T}\left(2M\frac{\sum_{k=1}^{K}\sum_{j=1}^{\ell}|\epsilon_j^k|M}{\ell}\right) \geq P_T(A,\boldsymbol{x}) - 2M^2\ell T\sqrt{\frac{\log K}{T-\ell}}$$

where $\epsilon_j^k = w_t^k - w_{t-j+1}^k$. The first inequality is an application of Jensen's inequality using the convexity of x^2. The third inequality follows from the fact that $\sum_{k=1}^{K}|\epsilon_j^k|$ is bounded by $j\sqrt{\frac{\log K}{T-j}}$ using our best expert assumption. □

The following theorem is the main result of this section, describing a no-regret algorithm with respect to the risk-sensitive function G_T.

Theorem 5. *Let A be a best expert algorithm that satisfies Definition 1 with instantaneous gain function $g_t^k = x_t^k - (x_t^k - AVG_\ell^*(x_1^k,...,x_{t-1}^k))^2$ for expert k at time t. Then for large enough T for any reward sequence $\boldsymbol{x}$ and any expert k we have for window size ℓ*

$$G_T(A,\boldsymbol{x}) \geq G_T(k,\boldsymbol{x}) - O\left(M^2\ell\sqrt{\frac{\log K}{T-\ell}}\right)$$

Proof: Using the best expert assumption and Lemma 1, we have

$$T\cdot G(k,\boldsymbol{x}) = \sum_{t=1}^{T} x_t^k - \sum_{t=1}^{T}(x_t^k - AVG_\ell^*(x_1^k,..,y_t^k))^2$$

$$\leq \sum_{t=1}^{T}\sum_{k'=1}^{K} w_t^{k'}x_t^{k'} - \sum_{t=1}^{T}\sum_{k'=1}^{K} w_t^{k'}(x_t^{k'} - AVG_\ell^*(x_1^{k'},..,x_t^{k'}))^2 + M\sqrt{T\log K}$$

$$\leq T\cdot G(A,\boldsymbol{x}) + 2TM^2\ell\sqrt{\frac{\log K}{T-\ell}} + M\sqrt{T\log K}$$

Dividing both sides by T yields the result. □

Corollary 1. *Let A be a best expert algorithm that satisfies Definition 1 with instantaneous reward function $g_t^k = x_t^k - (x_t^k - AVG_\ell^*(x_1^k,...,x_{t-1}^k))^2$. Then for large enough T we have for any expert k and fixed window size $\ell = O(\log T)$*

$$G(A,\boldsymbol{x}) \geq G(k,\boldsymbol{x}) - \tilde{O}\left(M^2\sqrt{\frac{\log K}{T}}\right)$$

7 Empirical Results [2]

In order to judge how well algorithms with no-regret guarantees are able to perform on real data sets with large numbers of experts, we implemented the

[2] The section of experimental results in the version of this paper published in [6] contains significant errors, and should be disregarded in favor of the corrected results presented here. Briefly, the experimental results of [6] significantly overstate the performance of the Weighted Majority and Modified Weighted Majority algorithms. All theoretical results of [6] are unaffected by these errors.

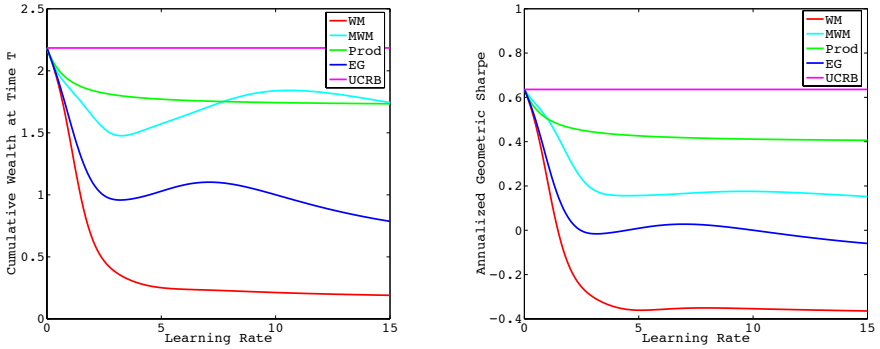

Fig. 1. Cumulative wealth (left) and annualized geometric Sharpe ratio (right) at the end of the 1632-day period for each algorithm with a selection of positive learning rates.

standard Multiplicative Weights/Weighted Majority algorithm (WM) [9, 4, 7], the modified risk-sensitive version of Weighted Majority (MWM) discussed in Section 6, Prod [5], and the exponential gradient (EG) algorithm of Hembold et al. [8]. We ran each of these algorithms with a wide variety of learning rates on a data set consisting of the closing prices on the 1632 trading days between January 4, 1999 and June 29, 2005 of the 469 S&P 500 stocks that remained in the index for the duration of the period. For the modified risk-sensitive version of Weighted Majority, we used a fixed window of size 40. Other similar window sizes yielded similar performance results.

Figure 1 shows the cumulative wealth and the annualized geometric Sharpe ratio of the four algorithms along with those of the uniform constant rebalanced (UCRB) portfolio (in which the weight is split evenly over the K experts at every time step). The x axis ranges over positive values of each algorithm's learning rate. Somewhat alarmingly, we find that all four algorithms are outperformed by UCRB on this data set. This may be attributed in part to the large number of stocks available, but could be a direct result of the mean-reverting nature of the stock market. Similar results were noted by Borodin, El-Yaniv, and Gogan [3] and by Agarwal et al.[1]

The second set of experiments was inspired by the form of the error in the initial version of this paper and by a surprising phenomenon brought to our attention by Cenk Ural who noticed that the performance of EG actually improved when he attempted using *negative* learning rates.[13] Figure 2 shows the cumulative wealth and annualized geometric Sharpe ratio of Weighted Majority, our modified risk-sensitive Weighted Majority, and EG over a range of negative learning rates, along with the uniform constant rebalanced portfolio and the best single stock (BSS) with respect to each metrics. [3] Indeed we see an

[3] Due to the form of the update rule for Prod, it is not possible to test a large range of negative learning rates while maintaining positive weights on each expert on this data set, but we suspect something similar is going on there.

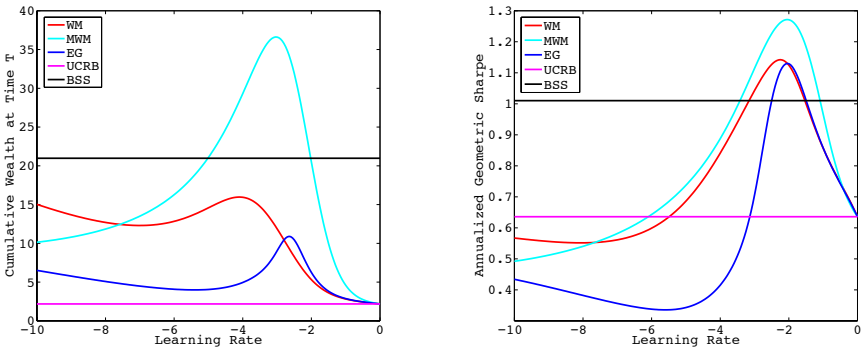

Fig. 2. Cumulative wealth (left) and annualized geometric Sharpe ratio (right) at the end of the 1632-day period for each algorithm with *negative* learning rates.

improvement in performance for all three algorithms, with our modified version of Weighted Majority yielding significantly higher cumulative wealth than the best single stock at some learning rates.

Simulations of this form raise a number of interesting questions about why it might be beneficial to decrease the weight of experts that have historically done well in favor of experts with a history of poor performance. This could be explained by the common theory that the stock market is mean-reverting; if the value of a stock climbs, it is likely to fall in the future and vice versa. It would be interesting to examine the possible theoretical guarantees and empirical performance of algorithms designed to perform well under assumptions of this kind.

References

1. A. Agarwal, E. Hazan, S. Kale, and R. E. Schapire. Algorithms for Portfolio Management based on the Newton Method, *ICML*, 2006.
2. Z. Bodie, A. Kane, and A. J. Marcus. Portfolio Performance Evaluation, Investments, 4th edition, Irwin McGraw-Hill, 1999.
3. A. Borodin, R. El-Yaniv, and V. Gogan. Can We Learn to Beat the Best Stock, *JAIR*, 21: 579–594, 2004.
4. N. Cesa-Bianchi, Y. Freund, D. Haussler, D. Helmbold, R.E. Schapire, and M.K. Warmuth. How to Use Expert Advice, *J. of the ACM*, 44(3): 427-485, 1997.
5. N. Cesa-Bianchi, Y. Mansour, and G. Stoltz. Improved Second-Order Bounds for Prediction with Expert Advice, *COLT*, 217–232, 2005.
6. E. Even-Dar, M. Kearns and J. Wortman. Risk Sensitive Online Learning. *ALT*, 2006.
7. Y. Freund, R. Schapire. Adaptive game playing using multiplicative weights. *Games and Economic Behavior*, 29:79–103, 1999.
8. D.P. Helmbold, R.E. Schapire, Y. Singer, and M.K. Warmuth. On-line portfolio selection using multiplicative updates, *Mathematical Finance*, 8(4): 325–347, 1998.

9. N. Littlestone and M. K. Warmuth. The Weighted Majority Algorithm, *Information and Computation,* 108(2): 212-261, 1994.
10. H. Markowitz. Portfolio Selection, *The Journal of Finance*, 7(1):77–91, 1952.
11. W. F. Sharpe. Mutual Fund Performance, *The Journal of Business*, Vol 39, Number 1, part 2: Supplement on Security Prices, 119-138, 1966.
12. M. K. Warmuth and D. Kuzmin. Online Variance Minimization, *COLT*, 2006.
13. C. Ural. Personal communication. 2005.

A Proof of Theorem 1

We define an *m-segment sequence* as a sequence described by expert rewards at m times, $n_1 < ... < n_m$, such that for all $i \in \{1, \ldots, m\}$, every expert's reward in the time segment $[n_{i-1}+1, n_i]$ is constant, i.e. $\forall t \in [n_{i-1}+1, n_i], \forall k \in \mathcal{K}, x_t^k = x_{n_i}^k$. where $n_0 = 0$. We say that an algorithm has a *fixed policy* in the ith segment if the weights that the algorithm places on each expert remain constant between times $n_{i-1}+1$ and n_i. The following lemma states that an algorithm with maximal Sharpe ratio at time n_i uses a fixed policy at every segment prior to i.

Lemma 2. *Let $\boldsymbol{x}$ be an m-segment reward sequence. Let A_i^r (for $i \leq m$) be the set of algorithms that have average reward r on $\boldsymbol{x}$ at time n_i. Then an algorithm $A \in A_i^r$ with minimal standard deviation has a fixed policy in every segment prior to i. The optimal Sharpe ratio at time n_i is thus attained by an algorithm that has a fixed policy in every segment prior to i.*

The intuition behind this lemma is that switching weights within a segment can only result in higher variance without enabling an algorithm to achieve an average reward any higher than it would have been able to achieve using a fixed set of weights in this segment. The proof is omitted due to lack of space.

With this lemma, we are ready to prove Theorem 1. We will consider one specific 3-segment sequence with two experts and show that there is no algorithm that can have competitive ratio bigger than 0.71 at both times n_2 and n_3 on this sequence. The three segments are of equal length. The rewards for expert 1 are .05, .01, and .05 in intervals 1, 2, and 3 respectively. The rewards for expert 2 are .011, .009, and .05. [4] The Sharpe ratio of the algorithm will be compared to the Sharpe ratio of the best expert at times n_2 and n_3. Analyzing the sequence we observe that the best expert at time n_2 is expert 2 with Sharpe ratio 10. The best expert at n_3 is expert 1 with Sharpe ratio approximately 1.95.

The intuition behind this construction is that in order for the algorithm to have a good competitive ratio at time n_2 it cannot put too much weight on expert 1 and must put significant weight on expert 2. However, putting significant weight on expert 2 prevents the algorithm from being competitive in time n_3 where it must have switched completely to expert 1 to maintain a good Sharpe ratio. The remainder of the proof formalizes this notion.

[4] Note that since the Sharpe ratio is a unitless measure, we could scale the rewards in this sequence by any positive constant factor and the proof would still hold.

Suppose first that the average reward of the algorithm on the lower bound Sharpe sequence $\boldsymbol{x}$ at time n_2 is at least .012. The reward in the second segment can be at most .01, so if the average reward at time n_2 is $.012 + z$ where z is positive constant smaller than .018, then the standard deviation of the algorithm at n_2 is at least $.002+z$. This implies that the algorithm's Sharpe ratio is at most $\frac{.012+z}{.002+z}$, which is at most 6. Comparing this to the Sharpe ratio of 10 obtained by expert 2, we see that the algorithm can have a competitive ratio no higher than 0.6, or equivalently the algorithm's regret is at least 4.

Suppose instead that the average reward of the algorithm on $\boldsymbol{x}$ at time n_2 is less than .012. Note that the Sharpe ratio of expert 1 at time n_3 is approximately $\frac{.03667}{.018} > 1.94$. In order to obtain a bound that holds for any algorithm with average reward at most .012 at time n_2, we consider the algorithm A which has reward of .012 in every time step and clearly outperforms any other algorithm.[5] The average reward of A for the third segment must be .05 as it is the reward of both experts. Now we can compute its average and standard deviation $\bar{R}_{n_3}(A, \boldsymbol{x}) \approx 2.4667$ and $\sigma_{n_3}(A, \boldsymbol{x}) \approx 1.79$. The Sharpe ratio of A is then approximately 1.38, and we find that A has a competitive ratio at time n_3 that is at most 0.71 or equivalently its regret is at least 0.55.

The lower bound sequence that we used here can be further improved to obtain a competitive ratio of .5. The improved sequence is of the form $n, 1, n$ for the first expert's rewards, and $1+1/n, 1-1/n, n$ for the second expert's rewards. As n approaches infinity, the competitive ratio of the Sharpe ratio tested on two checkpoints at n_2 and n_3 approaches .5.

B Proof of Theorem 2

Recall that we are considering a two expert scenario. Until time n, expert 1 receives a reward of 2 with probably 1/2 and a reward of 0 with probability 1/2. From n to $2n$, he always receives 1. Expert 2 always receives 1/4. Recall that we refer to the set of sequences that can be generated by this distribution as S.

In this analysis we use a form of Azuma's inequality, which we present here for sake of completeness. Note that we cannot use standard Chernoff bound since we would like to provide bounds on the behavior of adaptive algorithms.

Lemma 3 (Azuma). *Let $\zeta_0, \zeta_1, ..., \zeta_n$ be a martingale sequence such that for each i, $1 \leq i \leq n$, we have $|\zeta_i - \zeta_{i-1}| \leq c_i$ where the constant c_i may depend on i. Then for $n \geq 1$ and any $\epsilon > 0$*

$$Pr\left[|\zeta_n - \zeta_0| > \epsilon\right] \leq 2e^{-\frac{\epsilon^2}{2\sum_{i=1}^{n} c_i^2}}$$

Now we define two martingale sequences, $y_t(\boldsymbol{x})$ and $z_t(A, \boldsymbol{x})$. The first counts the difference between the number of times expert 1 receives a reward of 2 and the number of times expert 1 receives a reward of 0 on a given sequence $\boldsymbol{x} \in S$. The second counts the difference between the weights that algorithm A places on expert 1 when expert 1 receives a reward of 2 and the weights placed on

[5] Of course such an algorithm cannot exist for this sequence

expert 1 when expert 1 receives a reward of 0. We define $y_0(\boldsymbol{x}) = z_0(A, \boldsymbol{x}) = 0$ for all $\boldsymbol{x}$ and A.

$$y_{t+1}(\boldsymbol{x}) = \begin{cases} y_t(\boldsymbol{x}) + 1, & x^1_{t+1} = 2 \\ y_t(\boldsymbol{x}) - 1, & x^1_{t+1} = 0 \end{cases}, \quad z_{t+1}(A, \boldsymbol{x}) = \begin{cases} z_t(A, \boldsymbol{x}) + w^1_{t+1}, & x^1_{t+1} = 2 \\ z_t(A, \boldsymbol{x}) - w^1_{t+1}, & x^1_{t+1} = 0 \end{cases}$$

In order to simplify notation throughout the rest of this section, we will often drop the parameters and write y_t and z_t when A and $\boldsymbol{x}$ are clear from context.

Recall that $\bar{R}_t(A, \boldsymbol{x})$ is the average reward of an algorithm A on sequence $\boldsymbol{x}$ at time t. We denote the *expected* average reward at time t as $\bar{R}_t(A, D) = E_{\boldsymbol{x} \sim D}\left[\bar{R}_t(A, \boldsymbol{x})\right]$, where D is the distribution over rewards.

Next we define a set of sequences that are "close" to the balanced sequence on which the algorithm A will have a high reward, and subsequently show that for algorithms with high expected average reward this set is not empty.

Definition 2. *Let A be any algorithm and δ any positive constant. Then the set S^δ_A is the set of sequences $\boldsymbol{x} \in S$ that satisfy (1) $|y_n(\boldsymbol{x})| \leq \sqrt{2n\ln(2n)}$, (2) $|z_n(A, \boldsymbol{x})| \leq \sqrt{2n\ln(2n)}$, (3) $\bar{R}_n(A, \boldsymbol{x}) \geq 1/4 + \delta - O(1/n)$.*

Lemma 4. *Let δ be any positive constant and A be an algorithm such that $\bar{R}_n(A, D) \geq 1/4 + \delta$. Then S^δ_A is not empty.*

Proof: Since y_n and z_n are martingale sequences, we can apply Azuma's inequality to show that $\Pr[y_n \geq \sqrt{2n\ln(2n)}] < 1/n$ and $\Pr[z_n \geq \sqrt{2n\ln(2n)}] < 1/n$. Thus, since rewards are bounded by a constant value in our construction (namely 2), the contribution of sequences for which y_n or z_n are larger than $\sqrt{2n\ln(2n)}$ to the expected average reward is bounded by $O(1/n)$. This implies that if there exists an algorithm A such that $\bar{R}_n(A, D) \geq 1/4 + \delta$, then there exists a sequence $\boldsymbol{x}$ for which the $\bar{R}_n(A, \boldsymbol{x}) \geq 1/4 + \delta - O(1/n)$ and both y_n and z_n are bounded by $\sqrt{2n\ln(2n)}$. □

Now we would like to analyze the performance of an algorithm for some sequence $\boldsymbol{x}$ in S^δ_A. We first analyze the balanced sequence where $y_n = 0$ with a balanced algorithm (so $z_n = 0$), and then show how the analysis easily extends to sequences in the set S_A. In particular, we will first show that for the balanced sequence the optimal policy in terms of the objective function achieved has one fixed policy in times $[1, n]$ and another fixed policy in times $[n + 1, 2n]$. Due to lack of space the proof, which is similar but slightly more complicated than the proof of Lemma 2, is omitted.

Lemma 5. *Let $\boldsymbol{x} \in S$ be a sequence with $y_n = 0$ and let $A^{\boldsymbol{x}}_0$ be the set of algorithms for which $z_n = 0$ on $\boldsymbol{x}$. Then the optimal algorithm in $A^{\boldsymbol{x}}_0$ with respect to the objective function $MV(A, \boldsymbol{x})$ has a fixed policy in times $[1, n]$ and a fixed policy in times $[n + 1, 2n]$.*

Now that we have characterized the optimal algorithm for the balanced setting, we will analyze its performance. The next lemma (proof omitted) connects the average reward to the standard deviation on balanced sequences by using the fact that on balanced sequences algorithms behave as they are "expected."

Lemma 6. *Let $\boldsymbol{x} \in S$ be a sequence with $y_n = 0$, and let $A_0^{\boldsymbol{x}}$ be the set of algorithms with $z_n = 0$ on $\boldsymbol{x}$. For any positive constant δ, if $A \in A_0^{\boldsymbol{x}}$ and $\bar{R}_n(A, \boldsymbol{x}) = 1/4 + \delta$, then $\sigma_n(A, \boldsymbol{x}) \geq \frac{4\delta}{3}$.*

The following is a bound on the objective function at time $2n$ given high reward at time n. The proof (again omitted) uses the fact the added standard deviation is at least as large as the added average reward and thus cancels it.

Lemma 7. *Let $\boldsymbol{x}$ be any sequence and A any algorithm. If $\bar{R}_n(A, \boldsymbol{x}) = 1/4 + \delta$, then $MV_{2n}(A, \boldsymbol{x}) \leq 1/4 + \delta$ for any positive constant δ.*

Recall that the best expert at time n is expert 2 with reward $1/4$ and standard deviation 0, and the best expert at time $2n$ is expert 1 with average reward 1 and standard deviation $1/\sqrt{2}$. Using this knowledge in addition to Lemmas 6 and 7, we obtain the following proposition for the balanced sequence:

Proposition 1. *Let $\boldsymbol{x} \in S$ be a sequence with $y_n = 0$, and let $A_0^{\boldsymbol{x}}$ be the set of algorithms with $z_n = 0$ for s. If $A \in A_0^{\boldsymbol{x}}$, then A has a constant regret at either time n or time $2n$ or at both.*

We are now ready to return to the non-balanced setting in which y_n and z_n may take on values other than 0. Here we use the fact that there exists a sequence in S for which the average reward is at least $1/4 + \delta - O(1/n)$ and for which y_n and z_n are small. The next lemma shows that standard deviation of an algorithm A on sequences in S_A^δ is high at time n. The proof, which is omitted, uses the fact that such sequences and algorithm can be changed with almost no effect on average reward and standard deviation to balanced sequence, for which we know the standard deviation of any algorithm must be high.

Lemma 8. *Let δ be any positive constant, A be any algorithm, and $\boldsymbol{x}$ be a sequence in S_A^δ. Then $\sigma_n(A, \boldsymbol{x}) \geq \frac{4\delta}{3} - O\left(\sqrt{\ln(n)/n}\right)$.*

We are ready to prove the main theorem of the section.

Proof: [Theorem 2] Let δ be any positive constant. If $\bar{R}_n(A, D) < 1/4 + \delta$, then there must be a sequence $\boldsymbol{x} \in S$ with $y_n \leq \sqrt{2n \ln(2n)}$ and $\bar{R}_n(A, \boldsymbol{x}) < 1/4 + \delta$. Then the regret of A at time $2n$ will be at least $1 - 1/\sqrt{2} - 1/4 - \delta - O(1/n)$.

If, on the other hand, $\bar{R}_n(A, D) \geq 1/4 + \delta$, then by Lemma 4 there exists a sequence $\boldsymbol{x} \in S$ such that $\bar{R}_n(A, \boldsymbol{x}) \geq 1/4 + \delta - O(1/n)$. By Lemma 8, $\sigma_n(A, \boldsymbol{x}) \geq 4/3\delta - O\left(\sqrt{\ln(n)/n}\right)$, and thus the algorithm has regret at time n of at least $\delta/3 - O\left(\sqrt{\ln(n)/n}\right)$. This shows that for any δ we have that either the regret at time n is constant or the regret at time $2n$ is constant. □

C Proof of Theorem 4

The following facts about the behavior of $\ln(1 + z)$ for small z will be useful.

Lemma 9. *For any $L > 2$ and any v, y, and z such that $|v|$, $|y|$, $|v+y|$, and $|z|$ are all bounded by $1/L$ we have the following*

$$z - \frac{(3L+2)z^2}{6L} < \ln(1+z) < z - \frac{(3L-2)z^2}{6L}$$

$$\ln(1+v) + \frac{Ly}{L+1} < \ln(1+v+y) < \ln(1+v) + \frac{Ly}{L-1}$$

Similar to the analysis in [5], we bound $\ln \frac{\tilde{W}_{n+1}}{\tilde{W}_1}$ from above and below.

Lemma 10. *For the algorithm Prod with $\eta = 1/(LM) \leq 1/4$ where $L > 2$,*

$$\ln \frac{\tilde{W}_{n+1}}{\tilde{W}_1} \leq \frac{\eta L R_n(A, \boldsymbol{x})}{L-1} - \frac{\eta^2 (3L-2) n Var_n(A, \boldsymbol{x})}{6L}$$

at any time n for sequence $\boldsymbol{x}$ with the absolute value of rewards bounded by M.

Proof: Similarly to [5] we obtain,

$$\begin{aligned}\ln \frac{\tilde{W}_{n+1}}{\tilde{W}_1} &= \sum_{t=1}^{n} \ln \frac{\tilde{W}_{t+1}}{\tilde{W}_t} = \sum_{t=1}^{n} \ln \left(\sum_{k=1}^{K} \frac{\tilde{w}_t^k}{\tilde{W}_t} (1 + \eta x_t^k) \right) = \sum_{t=1}^{n} \ln(1 + \eta x_t^A) \\ &= \sum_{t=1}^{n} \ln(1 + \eta(x_t^A - \bar{R}_n(A, \boldsymbol{x}) + \bar{R}_n(A, \boldsymbol{x})))\end{aligned}$$

Now using Lemma 9 twice we obtain the proof. □

Next we bound $\ln \frac{\tilde{W}_{n+1}}{\tilde{W}_1}$ from below. The proof is based on similar arguments to the previous lemma and the observation made in [5] that $\ln \frac{\tilde{W}_{n+1}}{\tilde{W}_1} \geq \ln \left(\frac{\tilde{w}_{n+1}^k}{K} \right)$, and is thus omitted.

Lemma 11. *For the algorithm Prod with $\eta = 1/LM$ where $L > 2$, for any expert $k \in \mathcal{K}$ the following is satisfied*

$$\ln \frac{\tilde{W}_{n+1}}{\tilde{W}_1} \geq -\ln K + \frac{\eta L R_n(k, \boldsymbol{x})}{L+1} - \frac{\eta^2 (3L+2) n Var_n(k, \boldsymbol{x})}{6L}$$

at any time n for any sequence $\boldsymbol{x}$ with rewards absolute values bounded by M.

Combining the two lemmas we obtain Theorem 4.