External research project
University of Cambridge

Online Convex Optimization for Portfolio Management

Supervised by Thomas Gillam
at Cantab Capital

Marin Ballu

March 2019

Contents

1 Introduction 2

2 Theoretical foundations 3
- 2.1 Online convex optimization 3
  - 2.1.1 Definitions 3
  - 2.1.2 Various Hypothesis 4
- 2.2 The Experts problem 5
  - 2.2.1 Decision making with experts advices 5
  - 2.2.2 Prediction with experts advice 6
- 2.3 Universal portfolio management 7

3 Algorithms 8
- 3.1 Follow the Leader and variants 8
- 3.2 The Exponential Weights Method 10
  - 3.2.1 Hedge 10
  - 3.2.2 Refining Regret Bounds 12
- 3.3 Adaptive Regret 14
- 3.4 Dealing with unbounded loss 15
- 3.5 Summary 17

4 Testing 17

5 Conclusion 20

6 References 20

1 Introduction

This report serves as an overview of some algorithms, derived from the theory of online convex optimization, that are used to solve portfolio optimization problems. It was written during a PhD external project from a partnership between Cambridge Center for Analysis and the hedge fund Cantab Capital.

The material is intended for the Quants who want to have an overview of existing algorithms and hypothesis in order to enhance the performance of their own investment algorithms. This report can also interest the mathematicians who are curious about the application of the theory of online optimization in finance.

The goal of online convex optimization is to optimize a parameter sequentially in a changing environment, while knowing the results of the past chosen values and possibly additional information. Online convex optimization has been studied in several research fields including game theory, information theory and statistics. In the special case where the algorithm minimizes an error rate for an estimation, it has been of great interest in machine learning under the name of online learning [23].

In a first part we introduce the definitions of the setting of online convex optimization and the expert problem. Then, in a second part, some algorithms and their regret bounds are exposed. Finally, a small amount of testing on raw and simulated data is presented.

2 Theoretical foundations

2.1 Online convex optimization

2.1.1 Definitions

The online convex optimization setup consists of a player making choices in an environment, and receiving a convex loss based on his choices. The study of such a system is motivated by the desire to make mathematically driven decisions in a situation that is changing with time. Formally, at each timestep, the following happens:

1. The player chooses $x_t \in X$, where X is a convex subset of $\mathbb{R}^n$.
2. A loss function $f_t : X \to \mathbb{R}$ is picked in a family of positive convex functions $\mathcal{K}$ by the environment.
3. The loss $f_t(x_t)$ is revealed to the player, along with additional information about f_t.

The goal of the player is to minimize the *cumulative regret*, which is the difference between the loss that the algorithm accumulates there and the best fixed optimal loss in hindsight, defined as

$$R_T = \sum_{t=0}^{T} f_t(x_t) - \inf_{x^* \in X} \sum_{t=0}^{T} f_t(x^*).$$

We can see that for the regret to be finite, the functions f of $\mathcal{K}$ should all be lower bounded.

Sometimes, the environment can change radically, so *adaptative regret* is considered instead of cumulative regret. The *adaptative regret* is the worst cumulative regret between two timesteps $t_1 \leqslant t_2$:

$$AR_T = \sup_{0 \leqslant t_1 \leqslant t_2 \leqslant T} \left\{ \sum_{t=t_1}^{t_2} f_t(x_t) - \inf_{x^* \in X} \sum_{t=t_1}^{t_2} f_t(x^*) \right\}.$$

2.1.2 Various Hypothesis

Starting from these definitions, most results in online convex optimization are bounds on the cumulative or adaptative regret, assuming some hypothesis on three aspect of this framework:

- The type of choice the player has to make, i.e. the characterization of X.
- The way the environment picks the loss f_t. It includes the characterization of $\mathcal{K}$.
- The feedback from the environment, which forms the knowledge of the player has at time $t+1$ of the functions $(f_s)_{0 \leqslant s \leqslant t}$.

Choice of the player. The properties of the convex set $X \in \mathbb{R}^n$ is always defined jointly with the specificity of the problem. In most cases, no assumptions are made on X except convexity. When the losses are differentiable on some open set U, it can be assumed that $X \subset U$ for convenience. When choices are based on a finite number of obvious choices in a set $S \in \mathbb{R}^n$, for example in the case of experts advice, X is defined to be the convex hull of S.

Update of the loss. The update of f_t can be random or adversarial. In the first case, $(f_t)_{t \geqslant 0}$ is assumed to be a stochastic process with well-known properties, and the bounds on the regret are usually statistical bounds. However we will not focus here on statistical aspects of online learning but rather on the adversarial case. Indeed, in this case the desired bounds are worst case bounds, which means that we assume that the environment can choose a loss $f_t \in \mathcal{K}$ that is as bad as possible for the player. For the adversarial setting to make sense, the family of losses needs to satisfy a boundedness property:

$$\forall x \in X, \ \sup_{f \in \mathcal{K}} \left\{ f(x) - \inf_{x' \in X} f(x') \right\} < \infty.$$

If this property is not verified, the environment can always choose a loss as big as its wants, whatever choice the player made, so there is no good strategy for the player to limit the growth of his regret. In both the random and adversarial case, it should be understood that the player only has meaningful advantage if the possibilities of the environment are sufficiently restricted.

Feedback. We will mostly see results where the feedback to the player is the whole loss function f_t. However, when it is not the case, the knowledge of the player usually follows the *black box model*: at each step t, only local properties of f_t at the vicinity of the chosen x_t are revealed. We will see three depth of differential knowledge about f_t:

- Zeroth order black box: only $f_t(x_t)$ is revealed.

- First order black box. The value of f_t at x_t, and of its gradient $\nabla f_t(x_t)$ are revealed. If f_t is not differentiable at x_t a subgradient $y \in \partial f_t(x_t)$ is revealed, with

$$\partial f_t(x_t) := \left\{ y \in \mathbb{R}^n : \forall x \in X,\ y^T(x - x_t) \leqslant f(x) - f(x_t) \right\}.$$

  A subgradient exists at all point $x \in X$ because f_t is assumed to be convex.

- Second order black box. Assume $\mathcal{K} \subset C^2(X, \mathbb{R})$. The values $f_t(x_t)$, $\nabla f_t(x_t)$ and $\nabla^2 f_t(x_t)$ are revealed, with $\nabla^2 f_t(x_t)$ being the Hessian matrix of f_t at point x_t.

An interesting aspect of the black box model is that it is frequently used even in cases when f_t is completely revealed. Indeed, even with full feedback, the minimum of $\sum_t f_t$ on X might not be computable in polynomial time. A good strategy to compute the minimum is to use approximated versions of f_t based on their Taylor expansion at the points x_t. This is for example the motivation behind the *Online Newton Step* algorithm (Algorithm 3). Moreover, with such a strategy only a limited amount of information about f_t is stored during the algorithm. Limited storage is a very popular hypothesis in optimization and machine learning [22].

2.2 The Experts problem

2.2.1 Decision making with experts advices

In a lot of practical problems, there exists multiple different expert strategies which performance can be assessed in real time, thus the player should establish a meta strategy that combines the experts strategies in order to perform as well as possible. The study of the optimality of such a meta strategy is called the experts problem. The special case where experts provide a prediction for a given process is fundamental in machine learning theory. Indeed, suppose we have a fixed amount of different predicting algorithms that can be used for one prediction problem. For every time-step we want to combine algorithms in a way that guarantee optimal performance. Then the experts combination problem is an online convex optimization problem with the following setting.

Choice of the player. The choice of the player can be a convex combination of the choices of the experts $y_t(1), \ldots, y_t(n) \in \mathbb{R}^p$, $y_t(\text{player}) = \sum_i x_t(i) y_t(i)$. It is possible that only one expert should be chosen, in this case the player randomly chooses the expert i with probability $x_t(i)$. In both cases the choice of the player can be described as the vector $x_t = (x_t(i))_i \in X = \Delta_n$ in the *simplex*

$$\Delta_n = \left\{ x = (x(1), \ldots, x(n)) \in [0,1]^n : \sum_i x(i) = 1 \right\}.$$

There also exists a case where the player only has access to a subset of active experts, called *specialists*, at each time-step [8]. This limitation of the choice of the player also limits the performance of the algorithm.

Update of the loss. The most general case doesn't specify anything more than the convexity of the loss, and the fact that it is constructed from a concave reward function $g_t : Y \to \mathbb{R}$, with $g_t(y_t(i))$ being the reward obtained from the i-th expert. The loss is then defined as

$$f_t(x_t) = E^{i \sim x_t}[-g_t(y_t(i))] = -\sum_i x_t(i) g_t(y_t(i))$$

in the probabilistic case, and

$$f_t(x_t) = -g_t \left(\sum_i x_t(i) y_t(i) \right)$$

in the convex combination case.

Feedback. The player is usually supposed to know the performance of each expert at step $t-1$, which gives total knowledge of the loss function f_{t-1}. If the player only has knowledge of the chosen expert this problem is not called experts advice anymore but *multi-armed bandit*.

2.2.2 Prediction with experts advice

The special case where experts provide a prediction for a given process is fundamental in machine learning theory [23]. Indeed, suppose we have a fixed amount of different predicting algorithms that can be used for one prediction problem. For every time-step we want to combine algorithms in a way that guarantee optimal performance.

In the predictive case, the experts algorithms provide the predictions $y_t(1), \dots, y_t(n) \in \mathbb{R}$, and the estimation error for the i-th expert at time t be $\ell(y_t(i), y_t^*)$ such that ℓ is convex in each variables.

Choice of the player. As is the general case above, the choice of the player will be a vector in the simplex $X = \Delta_n$, whether he will apply the prediction of one expert at random or a convex combination of the predictions.

Update of the loss. The loss is the estimation error of the prediction of the player

$$f_t(x_t) = \ell(\hat{y}_t, y_t^*) = \ell \left(\sum_i x_t(i) y_t(i), y_t^* \right).$$

In this case we have

$$\mathcal{K} = \{ f : x \mapsto \ell(x^T y, y*) : \ y \in \mathbb{R}^n, \ y* \in \mathbb{R} \}.$$

The process $(y_t^*)_t$ to be predicted can either be assumed to be stochastic or adversarial with some bounds on its values. Usually, hypothesis are made on the error of the different experts $\ell(y_t(i), y_t^*)$ for first and second order bounds [6].

Feedback. The player is supposed to know $y_t(1), \dots, y_t(n), y^*_{t-1}$ at step t and knows the cost function ℓ, so he has complete knowledge of the loss function f_{t-1} in hindsight. However if the minimum of this loss function is not easily computable, the regret might become hard to compute, and the algorithm may finally rely on a black-box approach as for the algorithm Online Newton Step (Algorithm 3)[4].

2.3 Universal portfolio management

In universal portfolio management the objective is to devise a dynamic portfolio the difference of whose returns are similar to the best constant rebalanced portfolio in hindsight [20]. This model is called universal because it does not assume any underlying stochastic behavior of the stock prices. It means that such portfolio management strategies are efficient even when the market is adversarial.

A portfolio is a repartition of assets on different financial products which provide a return. It can be described by a sequence of vectors $x_t = (x_t(1), \dots, x_t(n)) \in \Delta_n$ in the simplex with n being the number of assets and $x_t(i)$ the proportion of assets that are invested in the i-th product at time t. Each asset provides return $r_t(i)$, which is the ratio of the price of stock i at time t to the price at time $t-1$. With this notion of return, the portfolio x_t yields return $x_t \cdot r_t = \sum_i x_t(i) r_t(i)$ at time t. It means that if an investor invests p_0 at time zero an uses the portfolio repartition $(x_t)_t$ until time t, his wealth at time T will be

$$p_T = p_0 \prod_{t=0}^{T} (x_t \cdot r_t) = p_0 \exp\left(\sum_{t=0}^{T} \log\left(x_t \cdot r_t\right) \right).$$

A good universal portfolio strategy is one that maximizes cumulated returns in any environment verifying some fixed hypothesis. We now see that this is exactly the general experts problem with the expert's rewards being the stock's returns $r_t(1), \dots, r_t(n)$ and the reward being the *log-return* of the portfolio $g_t(x_t \cdot r_t) = \log(x_t \cdot r_t)$.

An important aspect of the application of online convex optimization in portfolio management is that the setup described above for investment in stocks is also working for investment in different financial products, as well as portfolio management strategies themselves. For the rest of this document, we will use the term "expert" regardless of if the experts are stock or financial strategies, and use the vocabulary of online convex optimization, while every algorithm works in the specific setting of universal portfolio management. Note that it is possible to construct different universal portfolio algorithms and to even combine them by using expert combination. Such a structure can be built as a network of expert algorithms using each other, called belief Network [24].

In universal portfolio management the hypothesis one makes about the returns $(r_t)_t$ are fundamental and should never be understated for the construction of an algorithm. Indeed we have seen that if the environment is adversarial and the reward $\log(x_t \cdot r_t)$ unbounded from below, no strategy has an edge over any other strategy. In other words, in the adversarial case, one shouldn't assume that it is possible for a stock to loose more value than a fixed ratio of its price. Thus an universal portfolio management strategy cannot manage extreme risks at all, but it manages very well moderate risk: the environment can be adversarial up to a certain limit.

We will assume, in all the regret bounds presented below, that the log-returns all take value in a range of size 1: $-\delta \leqslant \log r_t(k) \leqslant 1-\delta$. We can increase the range to size M by multiplicating the bounds by M.

3 Algorithms

3.1 Follow the Leader and variants

In online convex optimization, if the loss functions $(f_t)_{t\geqslant 0}$ are completely revealed to the player, the simplest strategy is simply to take the point in the set X that minimizes cumulated loss in hindsight

$$x_{t+1} = \arg\min_{x\in X} \sum_{t'=0}^{t} f_t(x).$$

This strategy is called *Follow the Leader* [13]. Most algorithms in online convex optimization are variants of this straightforward idea. For universal portfolio management, it translates as

$$x_{t+1} = \arg\min_{x\in\Delta_n} \sum_{t'=0}^{t} \log(x\cdot r_{t'}).$$

It is not possible to compute the oracle minimum directly, however it is possible to perform any well-known convex optimization algorithm to approximate this minimum. If we have an optimization algorithm $\texttt{OptArgmin}(L_t)$ that approximates the minimizor of the cumulated loss $L_t(x) = \sum_{t'\leqslant t} f_t(x)$ then we can construct the following algorithm that approximates the *Follow the Leader* algorithm.

Algorithm 1 Follow the Leader

```
The initial strategy is $x_0 = \mathbf{1}/n$.
for $t = 1$ to $T$ do
    Play strategy $x_t$
    Gather the returns $r_t$ so that the function $L_t$ is known.
    Update $x_{t+1} = \texttt{OptArgmin}(L_t)$.
end for
```

The problem of such an algorithm is that it is unstable. For example if the loss alternates between two values, zero then one for one expert, and one then zero for a second expert, the algorithm will always choose the wrong expert because it is one time-step too late. It will thus yield the worst regret possible. However we can use a regularized version of the algorithm [14].

Algorithm 2 Follow the L^2-Regularized Leader

```
The initial strategy is $x_0 = \mathbf{1}/n$.
for $t = 1$ to $T$ do
    Play strategy $x_t$.
    Gather the returns $r_t$ so that the function $L_t$ is known.
    Update $x_{t+1} = \texttt{OptArgmin}(L_t(x) + \frac{\beta}{2}\|x\|^2)$.
end for
```

It is also possible and sometimes efficient to use L^1 regularization instead [21]. This algorithm has good regret guarantees if the the optimizer algorithm is efficient, as in the following bound [13]

$$R_T \leqslant \frac{2}{\beta}\sum_{t=0}^{T} \|\nabla_t\|_t^{*2} + \beta.$$

with

$$\nabla_t = \nabla[\log(x \cdot r_t)](x_t) = \frac{1}{x_t \cdot r_t} r_t.$$

In our particular case, we can actually precise the bound, we have $\|\nabla_t\|_t^{*2} \leqslant \frac{1}{\min_k x_t(k)^2} \leqslant C_{t,n}$, which can be a sharp bound if we assume for example that the returns are random [14], and $\beta = \sqrt{TC_{T,n}}$,

$$R_t \lesssim \sqrt{TC_{T,n}}.$$

For example one can use Newton method for optimization: let us replace the loss $\log(x \cdot r_t)$ by its second order Taylor expansion. We have

$$\log(x \cdot r_t) \approx \nabla_t^\top (x - x_t) - \frac{1}{2}[\nabla_t^\top (x - x_t)]^2.$$

The new regularized regret becomes a quadratic form plus some linear term, so its minimizer can be computed. This gives the algorithm called *Online Newton Step*.

This choice of the player can be "centered", if the player invests a fixed share for each expert in addition to the recommendation of the original algorithm, formally the player invests $(1-\alpha)p_t + \alpha\frac{1}{n}\mathbf{1}$. The interest of doing so is to guarantee that no expert will be underprioritized, it is very useful to moderate the regret in an adversarial environment. Indeed, if there is a chance that an underperforming expert has a burst of success, the centralization will guarantee some reward for the player. With this addition, there is a general regret bound obtained by Hazan et al. [4]: by setting $\alpha = \frac{n^{5/4}}{\sqrt{T\log(nT)}}$ and $\beta = \frac{1}{8n^{1/4}\sqrt{T\log(nT)}}$, we have

$$R_T \leqslant 22n^{5/4}\sqrt{T\log(nT)}.$$

Algorithm 3 Online Newton Step

```
  The initial strategy is $x_0 = \mathbf{1}/n$.
  Let $b_0 = 0$, $A_0 = I_n$.
  for $t = 0$ to $T$ do
    Play strategy $x_t$.
    Gather the returns $r_t$.
    Update $b_{t+1} = b_t + (1 + 1/\beta)\nabla_t$.
    Update $A_{t+1} = A_t + \nabla_t^T \nabla_t$
    Update
        $$x_{t+1} = \Pi_{\Delta_n}^{A_{t+1}}(A_{t+1}^{-1} b_{t+1}),$$
    with
        $$\Pi_{\Delta_n}^{A_t}(x) = \arg\min_{x' \in \Delta_n} (x - x')^\top A_t (x - x').$$
  end for
```

3.2 The Exponential Weights Method

3.2.1 Hedge

The *Aggregating Algorithm* [19], consists in combining the experts with weights that are inversely proportional to the exponential of the regret of each expert.

Algorithm 4 Aggregating algorithm

```
  The initial weights are $W_0(i) = 1$ and $x_0 = \mathbf{1}/n$.
  for $t = 0$ to $T$ do
    Play $x_t$.
    Gather the returns $r_t$.
    Update $W_{t+1}(i) = W_t(i) r_t(i)$
    Update $x_{t+1}(i) = \frac{W_{t+1}(i)}{\sum_k W_{t+1}(k)}$.
  end for
```

The advantage of this algorithm is its simplicity and flexibility, that make it a good model for making other algorithms. The algorithm can be modified by adding parameters which can then be optimized. We introduce the fixed share parameter α and the learning rate η in the following algorithm. As soon as a fixed learning rate is introduced, the algorithm is called the *Hedge algorithm* [10]. The fixed share helps reducing the conservativeness of the Aggregating algorithm. It renders the algorithm less efficient in the worst case but guarantees a better efficiency if we assume the experts randomly switch place at the rate α [3]. Indeed the exponential of an historically well performing expert trumps every other expert, thus if a low-performing expert starts having good performance, he will not be noticed until his cumulative performance is similar to the high performing one. Having a low learning rate has similar advantages, and the higher it is the faster the algorithm will focus on only one good expert near the

Algorithm 5 Hedge algorithm with fixed share

```
Let $0 \leqslant \alpha < 1$ be the fixed share.
Let $\eta > 0$ be the learning rate.
The initial weights are $W_0 = \mathbf{1}$ and $x_0 = \mathbf{1}/n$.
for $t = 0$ to $T$ do
    Play $x_t$.
    Gather the returns $r_t$.
    Update $W_{t+1}(i) = W_t(i) r_t(i)^\eta$.
    Update $x_{t+1}(i) = \frac{W_{t+1}(i)}{\sum_k W_{t+1}(k)}$.
    Update with fixed share $x_{t+1}(i) = (1-\alpha) x_{t+1}(i) + \alpha \mathbf{1}/n$.
end for
```

start of the algorithm. Note that in our special case of loss based on log-return, this algorithm is equivalent to the *Exponential Gradient algorithm* [17].

Cumulative Regret Bounds for Hedge. The regret of Hedge is bounded by

$$R_T \leqslant T\frac{\eta}{8} + \frac{\log n}{\eta}.$$

This bound is obtained by decomposing the loss at time t. Let $l_t(k) = -\log r_t$, we have Jensen inequality giving

$$-\log x_t \cdot r_t \leqslant x_t \cdot l_t$$

If we assume $0 \leqslant l_t \leqslant 1$, Hoeffding inequality, applied on the variable $e^{-\eta Y}$ where $Y = l_t(i)$ with probability $x_t(i)$, gives

$$x_t \cdot l_t \leqslant -\frac{1}{\eta} \log\left(\sum_i x_t(i) e^{-\eta l_t(i)}\right) + \frac{\eta}{8} = l_t(k) + \frac{1}{\eta}\log\left(\frac{x_{t+1}(k)}{x_t(k)}\right) + \frac{\eta}{8}$$

Then we substract the optimal loss $l_t(k)$ and add over t, and we have

$$R_T \leqslant T\frac{\eta}{8} + \frac{1}{\eta}\log(x_T(k)/x_0(k))$$

Then we remark that $x_0(k) = 1/n$ and $x_T(k) \leqslant 1$ to conclude our proof.

With $\eta = \sqrt{\frac{8\log n}{T}}$, we obtain an optimal adversarial regret bound for the Hedge algorithm

$$R_T \leqslant \sqrt{8T\log n}.$$

However we usually don't know the desired time horizon T, so to get an asymptotically efficient algorithm we can choose a decreasing learning rate $\eta_t = \sqrt{\frac{8\log n}{t}}$ and we have

$$R_T \lesssim \sqrt{T\log n}.$$

This bound cannot be improved in the adversarial case [5], because the environment can always find a counter-strategy that ensures a regret of $C\sqrt{T \log n}$. Consequently, the only reason to use another algorithm than the standard Hedge algorithm is that other stronger assumptions are made on the environment. Hedge is a good baseline to compare to in practice because it has low complexity and optimal theoretical bound. We will now see sharper regret bounds with stronger hypothesis.

3.2.2 Refining Regret Bounds

Second order bounds. The Hedge algorithm, where $\eta_t = \sqrt{\frac{8 \log n}{t}}$, attains its minimax bound very frequently in practice. If for example the losses $l_t(i)$ of each expert follows an iid distribution with mean $\bar{l}(k)$, then the Hedge algorithm already returns its worst performance $R_T \geqslant \Omega(\sqrt{T \log n})$ [11]. To palliate this problem, sharper bounds involving a certain notion of variance of the loss might be used. We call these *second order regret bounds* [6], of the form

$$R_T \lesssim \sqrt{V_T \log n} + \log T.$$

The variance here has different definitions in different examples. We can for example choose

$$V_T = \sum_{t=0}^{T} (x_t \cdot l_t - l_t(k))^2$$

the square excess loss. This variance is often supposed to be small $V_T \ll T$. We can remark that taking $\eta_t = \sqrt{(\log n)/V_T}$ in Hedge simply gives the desired regret bound. The bounds give good results in the case of independent losses [11], and in the case where the experts report their confidence [6]. The variation of Hedge that attains this bound is as follows.

Algorithm 6 Second Order Hedge

```
Let $\eta > 0$ be the learning rate.
The initial weights are $W_0 = \mathbf{1}$ and $x_0 = \mathbf{1}/n$.
for $t = 0$ to $T$ do
    Play $x_t$.
    Gather the returns $r_t$.
    Update
        $$W_{t+1}(i) = W_t(i) \exp\left(\eta \log r_t(i) - 4\eta^2 \left(\log r_t(i) - \frac{1}{t}\sum_{s=0}^{t-1} \log r_s(i)\right)^2\right).$$
    Update $x_{t+1}(i) = \frac{W_{t+1}(i)}{\sum_k W_{t+1}(k)}$.
end for
```

Remark that the updates on the weights here are equivalent to the following update: $W_{t+1} = e^{-\eta L_t - 4\eta^2 V_t}$ where $V_t = \frac{1}{t}\sum_{s=0}^{t-1}(l_s - \frac{L_{t-1}}{t})^2$.

Quantile Regret Bounds. The regret bounds can also be refined by expressing the minimum regret on only a subset K of good experts instead of all the experts. We define the *prior* on all the experts π as a probability distribution on $\{1, \ldots, n\}$, then a *quantile regret bound* with prior π is an inequality of the form

$$\min_{k \in K} R_T \lesssim \sqrt{T \log\left(\frac{1}{\pi(K)}\right)},$$

verified for every subset of experts K. This bound shows the performance of the algorithm if multiple experts are good. For example, if a lot of experts are used, the performance of Hedge would not be as good as if only a small subset was used, however it is possible construct an algorithm that is less impaired by the addition of similar experts [7] and performs even with an infinite set of expert. A Hedge-based algorithm that achieve the bound follows, called the *Normal Hedge algorithm* [7].

Algorithm 7 Normal Hedge

```
Let $\phi(R, \eta) = e^{\eta R^2 1_{R>0}}$ be the potential. Let $\pi$ be a prior distribution on the set of experts.
The initial weights are $W_0(i) = \pi(i)$ and $x_0 = \mathbf{1}/n$. Set the regret $R_0(i) = 0$ for every expert.
for $t = 0$ to $T$ do
    Play $x_t$.
    Gather the returns $r_t$.
    Update cumulative regret $R_t(k) = R_{t-1} + \log\left(\frac{r_t(k)}{x_t \cdot r_t}\right)$.
    Update $\eta_t$ that solves the equation $\sum_i \pi(i)\phi(R_t(k), \eta_t) = e$ (use a line search).
    Update
        $$W_{t+1}(i) = W_t(i)\frac{\partial \phi}{\partial R}(R_t(i), \eta_t) = W_{t+1} 2\eta_t R_t(i) e^{\eta R_t(i)^2} 1_{R_t(i)>0}.$$
    Update $x_{t+1}(i) = \frac{W_{t+1}(i)}{\sum_k W_{t+1}(k)}$.
end for
```

SQUINT. Koolen and Van Erven [18] provided an algorithm that combines the two approaches, and yields the following regret bound for every subset K of the experts:

$$R_T(K) \lesssim \sqrt{V_T(K)(C_\eta(K) - \log \pi(K))},$$

with $R_T(K) = \mathbb{E}_{\pi(k|K)} R_T(k)$, $V_T(K) = \mathbb{E}_{\pi(k|K)} V_T(k)$, and $C_\pi(K)$ depending on the prior on the learning rate η, taking for example the value $C_\eta(K) = \log V_T(K)$ for the uniform prior, or $C_\eta(K) = \log\log V_T(K)$ for $\eta \sim \frac{d\eta}{\eta(\log\eta)^2}$. The algorithm *SQUINT* is described as follows.

Algorithm 8 SQUINT

```
Let $\pi$ be the prior distribution on the set of experts, and $\gamma$ be the prior on the learning rate $\eta \in [0, 1/2]$.
The initial weights are $W_0(i) = \pi(i)$ and $x_0 = \mathbf{1}/n$. Set the regret $R_0(i) = 0$ for every expert, as well as the loss $L_0 = 0$ and the variance $V_0(0) = 0$.
for $t = 0$ to $T$ do
    Play $x_t$.
    Gather the returns $r_t$.
    Update cumulative loss $L_t(i) = L_{t-1}(i) - \log r_t(i)$.
    Update cumulative regret $R_t(i) = R_{t-1}(i) + \log\left(\frac{r_t(i)}{x_t \cdot r_t}\right)$.
    Update cumulative variance $V_t(i) = \frac{t-1}{t}\left(V_{t-1}(i) + (l_t(i) - \frac{L_{t-1}(i)}{t})^2\right)$.
    Update
        $$W_{t+1}(i) = \pi(i)\mathbb{E}_{\gamma(\eta)}\left[e^{\eta R_T(i) - \eta^2 V_T(i)}\right].$$
    Update $x_{t+1}(i) = \frac{W_{t+1}(i)}{\sum_k W_{t+1}(k)}$.
end for
```

3.3 Adaptive Regret

We show here an algorithm showing optimal adaptive regret bounds. In order to seek an adaptive algorithm, a modified version of Hedge can be constructed, which assumes the pool of experts evolves with time [16, 2, 15]. A method to build an experts combination

Algorithm 9 Follow the Leading History (FLTH)

```
Let $\eta > 0$ be the learning rate.
At each time, there is a set of experts $S_t$. The initial set is $S_1 = \{1\}$.
The initial weights are $W_0(i) = 1$ and $x_0 = \mathbf{1}/n$.
for $t = 0$ to $T$ do
    Play $x_t$.
    Gather the returns $r_t$.
    Update $W_{t+1}(i) = W_t(i)e^{\eta r_t(i)}$ for $i \in S_t$ and $W_{t+1}(j) = \mathbf{1}$ for $j \in S_{t+1} \cap S_t^c$.
    Update $S_t \to S_{t+1}$.
    Update $x_{t+1}(i) = \frac{W_{t+1}(i)}{\sum_{k \in S_{t+1}} W_{t+1}(k)}$ for $i \in S_{t+1}$.
end for
```

algorithm that provides good adaptive regret bounds involves applying the FTLH as a meta-algorithm to a pool of experts combination algorithms that start at different times. For example, if the player possess an online algorithm $\mathcal{A}$ that provides good cumulative regret, he will call $\mathcal{A}_0$ at time $t = 0$, then continue $\mathcal{A}_0, \ldots, \mathcal{A}_{t-1}$ at time t while starting a copy of $\mathcal{A}$ again from scratch by calling it $\mathcal{A}_t$. A certain history of lenght T_0 can be conserved by choosing $S_t = \{t - T_0 + 1, \ldots, t\}$. If the algorithm $\mathcal{A}$ guarantees cumulative regret $R_T(\mathcal{A})$, then the adaptative algorithm constructed with FLTH guarantees the adaptative regret bound

$$AR_T \leqslant R_T(\mathcal{A}) + O(\sqrt{T} \log T),$$

which gives, if one chooses a Newton step for $\mathcal{A}$,

$$AR_T \lesssim (M + n^{5/4})\sqrt{T}(\sqrt{\log n} + \log T).$$

If one uses Hedge as the original algorithm, we have the optimal bound

$$AR_T \lesssim \sqrt{T \log n} \log T.$$

3.4 Dealing with unbounded loss

Every regret bound shown so far assumes that the log-returns are in a range of the form $-\delta \leqslant \log r_t(k) \leqslant 1 - \delta$. However, in most practical cases, the return does not show such behavior. To still find solutions for those cases, we could assume that the range is wider, i.e. of the form $-\delta M \leqslant \log r_t(k) \leqslant (1 - \delta)M$. With this hypothesis, all the regret bounds of the algorithms shown previously are multiplied by M, if we replace in the algorithms the value of l_t by l_t/M. If the value of M is not known a priori, it can be updated in the algorithm as M_t with the empirical knowledge of $\max_{i,t} l_t(i)$. The problem with such a strategy is that if M is very big, then the algorithm will come to be very unnefficient. This is why another approach is jointly used when the loss varies in a wide range: randomization. The idea is that some random variables will be incorporated in the algorithm to guarantee a statistical regret bound. A simple version of randomization is the algorithm *Follow the Perturbed Leader*.

With the random perturbations following an exponential law and the perturbation rates well chosen, this algorithm has been shown to be *asymptotically consistent* [25]:

$$\limsup_{T \to \infty} \frac{\mathbb{E} R_T}{T} \leqslant 0.$$

It means that, in average and after a long time, the algorithm performs better than the best expert in hindsight.

As we know, we cannot compute the minimum in our special case, however the strategy performed in Online Newton Step is still valid, thus we may apply the perturbation scheme to obtain a perturbed version of Online Newton Step.

This algorithm has not been studied in the literature but will not provide much worse regret bounds in the case of bounded return than the original ONS, and it will still be asymptotically consistent.

Algorithm 10 Follow the Perturbed Leader

```
  Let $\mu_t > 0$ be a sequence of perturbation rates. Let $Y_t(1), \ldots, Y_t(n)$ be a sequence of
  iid random variables
  Set the weights $x_0 = \mathbf{1}/n$ and the cumulated maximum losses $M_0 = 0$.
  for $t = 1$ to $T$ do
    Play strategy $x_t$.
    Gather the returns $r_t$ so that the loss function $L_t$ is known.
    Update $M_t = M_{t-1} + |-\log \min_i r_t(i)|$.
    Update $x_{t+1} = \texttt{OptArgmin}(L_t(x) - \frac{\mu_t Y_t \cdot x}{M_t})$.
  end for
```

Algorithm 11 Perturbed Online Newton Step

```
  Let $\mu_t > 0$ be a sequence of perturbation rates. Let $Y_t(1), \ldots, Y_t(n)$ be a sequence of
  iid random variables. Let $\beta > 0$ be the regularization factor.
  Set the weights $x_0 = \mathbf{1}/n$ and the cumulated maximum losses $M_0 = 0$.
  Let $b_0 = 0$, $A_0 = I_n$.
  for $t = 0$ to $T$ do
    Play strategy $x_t$.
    Gather the returns $r_t$.
    Update $M_t = M_{t-1} + |-\log \min_i r_t(i)|$.
    Update $b_{t+1} = b_t + (1 + 1/\beta)\nabla_t + \frac{\mu_t Y_t}{M_t}$.
    Update $A_{t+1} = A_t + \nabla_t^T \nabla_t$
    Update
        $$x_{t+1} = \Pi_{\Delta_n}^{A_{t+1}}(A_{t+1}^{-1} b_{t+1}),$$
    with
        $$\Pi_{\Delta_n}^{A_t}(x) = \arg\min_{x' \in \Delta_n} (x - x')^\top A_t (x - x').$$
  end for
```

3.5 Summary

The following table summarizes the algorithms and bounds that have been presented in this document.

Algorithm	Regret Bound	Setting of efficiency
Follow the Regularised Leader	$R_T \lesssim \sqrt{TC_{T,n}}$	Adversarial
Online Newton Step	$R_T \lesssim n^{5/4}\sqrt{T}\log nT$	Adversarial, good in practice
Hedge	$R_T \lesssim \sqrt{T\log n}$	Adversarial, theoretically optimal
Second Order Hedge	$R_T \lesssim \sqrt{V_T \log n} + \log T$	Random Losses (better if iid)
Normal Hedge	$\min_{k\in K} R_T(k) \lesssim \sqrt{T(-\log \pi(K))}$	Lots of similar experts
SQUINT	$R_T(K) \lesssim \sqrt{V_T(K)(C_\eta(T) - \log \pi(K))}$	Lots of similar experts and random losses
Follow the Leading History	$AR_T \lesssim R_T(A) + \sqrt{T}\log T$	Evolutive set of experts, changing environment
Follow the Perturbed Leader	$\limsup \frac{\mathbb{E}R_T}{R} \leqslant 0$	Unbounded losses, high volatility

Table 1: Recapitulation of the presented algorithms

4 Testing

Raw data might be good for assessing the efficiency of an algorithm in practice, so we used a portfolio that only invests daily in oil and natural gas respectively from 1997 to 2019. The practical efficiency of the Online Newton Step for portfolio optimisation have been extensively studied by Hazan et al. [4]. The simple Hedge algorithm yields remarkable results as seen in the following figure.

The variants of Hedge that we have presented, such as Second Order Hedge, Normal Hedge and SQUINT, provide almost identical results to Hedge. It could be explained by the fact that in this market, the volatility is high enough that it could be considered adversarial, thus the simple Hedge algorithm is optimal. An adaptive Hedge based on Follow the Leading History, yields slighlty better returns than the classical Hedge. However, it is not significative enough to see it as a jump in performance.

For the simulations we construct artificial stocks using an autoregressive model. This is a mean of generating a stochastic process by specifying that the update at time t depends linearly on the processes previous values and some iid noise. This gives the following chart.

In this simulation, the performance of Hedge and its derivates were similar, and did better than the second best stock. The best performance was attained by the adaptive algorithm FTLH.

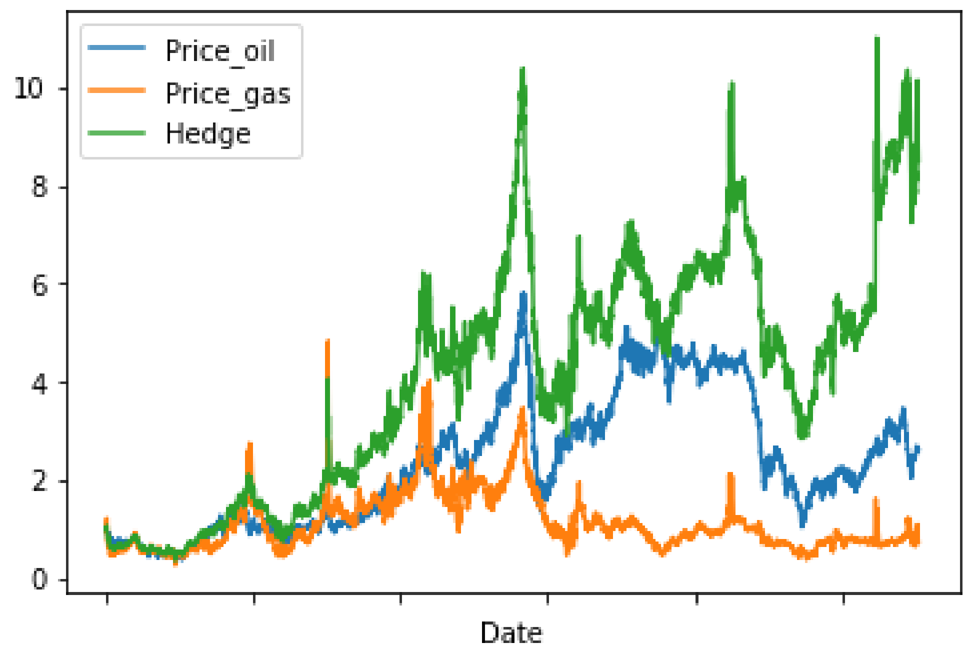

Figure 1: Performance of Hedge in combining oil and natural gas stocks

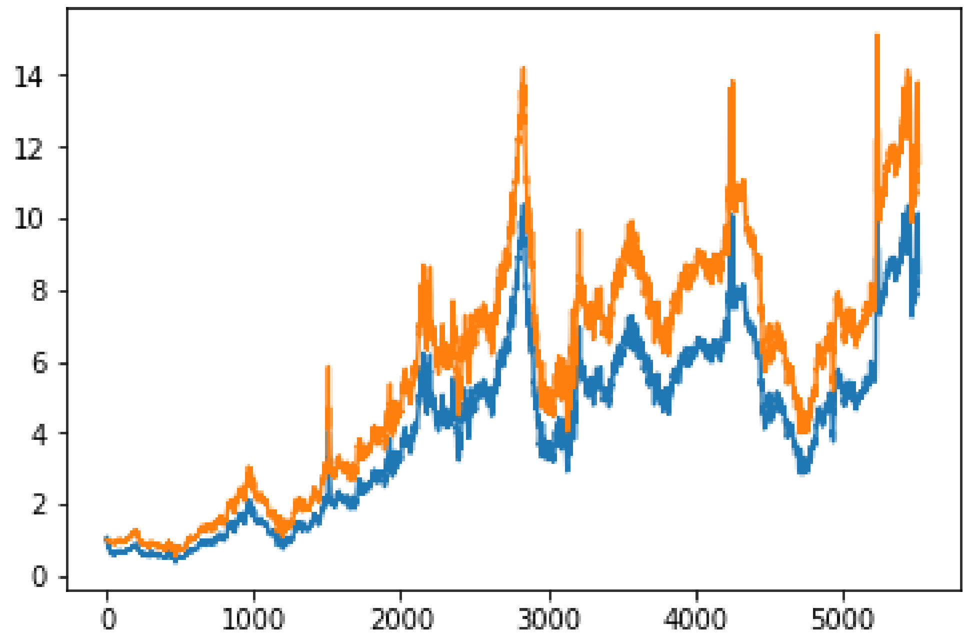

Figure 2: Performance of Follow the Leading History compared to Hedge

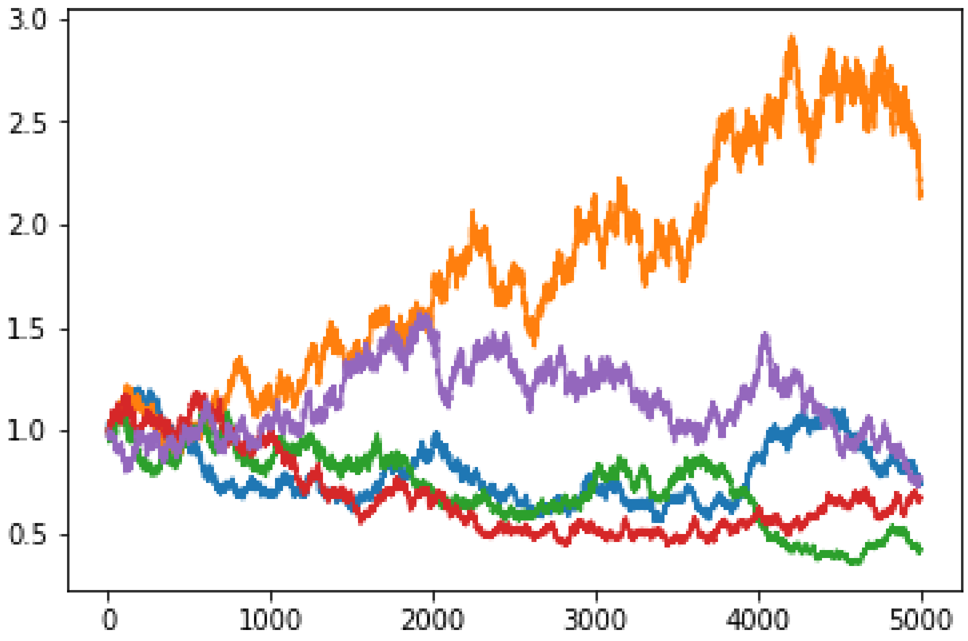

Figure 3: Simulation using autoregressive processes

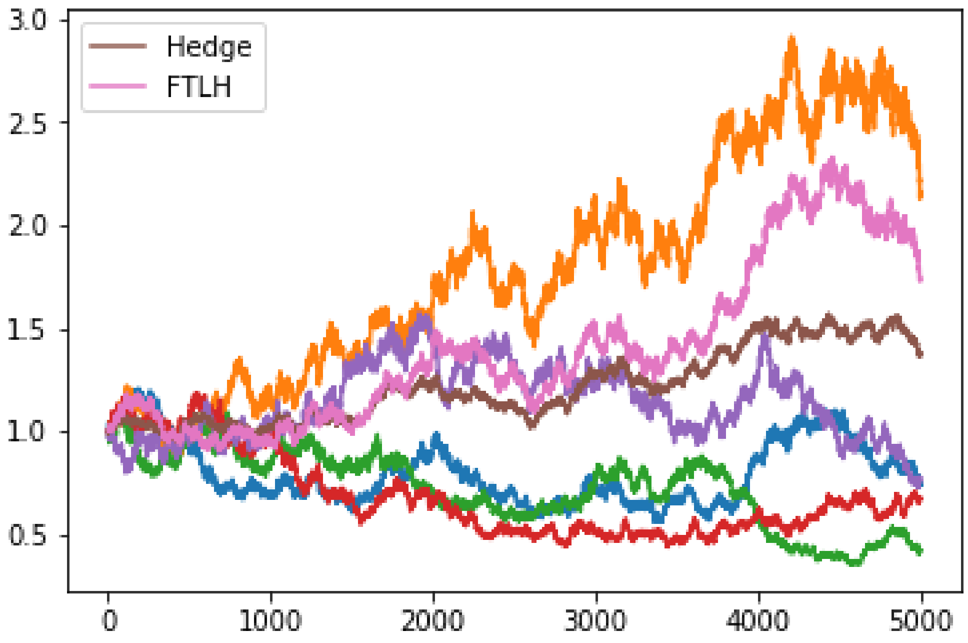

Figure 4: Performance of Hedge and FTLH in the simulation

5 Conclusion

A lot of algorithm for experts combination exist, and each person needing them in practice can fine-tune them for their use cases. The list that we have presented is far from exhaustive and there is still active research in the field of Online Convex Optimization, whether it is for industrial application [9], for quantum computing [1], for machine learning [12], or for Finance. A very interesting aspect of the experts method is that it can be applied as a meta strategy on other good financial strategies coming from other theoretical backgrounds, and benefit to the overall score.

I would just like to thank my project supervisor Thomas Gillam for his assistance and patience throughout the duration of my project. I would enjoy discussing more This is a fascinating topic to be working on and I hope to learn much more in the future.

6 References

[1] Scott Aaronson, Xinyi Chen, Elad Hazan, Satyen Kale, and Ashwin Nayak. Online learning of quantum states. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 8962–8972. Curran Associates, Inc., 2018.

[2] Dmitry Adamskiy, Wouter M. Koolen, Alexey Chernov, and Vladimir Vovk. A closer look at adaptive regret. In Nader H. Bshouty, Gilles Stoltz, Nicolas Vayatis, and Thomas Zeugmann, editors, *Algorithmic Learning Theory*, pages 290–304, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[3] Dmitry Adamskiy, Wouter M. Koolen, Alexey Chernov, and Vladimir Vovk. A closer look at adaptive regret. *Journal of Machine Learning Research*, 17(23):1–21, 2016.

[4] Amit Agarwal, Elad Hazan, Satyen Kale, and Robert E. Schapire. Algorithms for portfolio management based on the newton method. In *ICML*, 2006.

[5] Nicolo Cesa-Bianchi and Gabor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, USA, 2006.

[6] Nicolo Cesa-Bianchi, Yishay Mansour, and Gilles Stoltz. Improved second-order bounds for prediction with expert advice. LNAI 3559, pages 217–232. Springer, 2005.

[7] Kamalika Chaudhuri, Yoav Freund, and Daniel Hsu. A parameter-free hedging algorithm. 03 2009.

[8] Alexey V. Chernov and Vladimir Vovk. Prediction with expert evaluators' advice. *CoRR*, abs/0902.4127, 2009.

[9] Raphal Deswarte, Véronique Gervais, Gilles Stoltz, and Sébastien Da Veiga. Sequential model aggregation for production forecasting. working paper or preprint, November 2018.

[10] Yoav Freund and Robert E. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In Paul Vitányi, editor, *Computational Learning Theory*, pages 23–37, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg.

[11] Pierre Gaillard, Gilles Stoltz, and Tim Van Erven. A second-order bound with excess losses. *Journal of Machine Learning Research*, 35, 02 2014.

[12] Udaya Ghai, Elad Hazan, and Yoram Singer. Exponentiated gradient meets gradient descent. *CoRR*, abs/1902.01903, 2019.

[13] Elad Hazan. Introduction to online convex optimization. *Foundations and Trends in Optimization*, 2(3-4):157–325, 2016.

[14] Elad Hazan and Satyen Kale. Extracting certainty from uncertainty: regret bounded byvariation incosts. *Machine Learning*, 80(2):165–188, Sep 2010.

[15] Elad Hazan and C Seshadhri. Adaptive algorithms for online decision problems. *Electronic Colloquium on Computational Complexity (ECCC)*, 14, 01 2007.

[16] Elad Hazan and C. Seshadhri. Efficient learning algorithms for changing environments. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 393–400, New York, NY, USA, 2009. ACM.

[17] Jyrki Kivinen and Manfred K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Inf. Comput.*, 132(1):1–63, January 1997.

[18] Wouter M. Koolen and Tim van Erven. Second-order quantile methods for experts and combinatorial games. *CoRR*, abs/1502.08009, 2015.

[19] Alexander Korotin, Vladimir V'yugin, and Evgeny Burnaev. Aggregating strategies for long-term forecasting. 03 2018.

[20] Thomas M. Cover. Universal portfolios. *Mathematical Finance*, 1, 01 1997.

[21] H. Brendan McMahan. Follow-the-regularized-leader and mirror descent: Equivalence theorems and l1 regularization. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2011.

[22] Nicol N. Schraudolph, Jin Yu, and Simon Gnter. A stochastic quasi-newton method for online convex optimization. In Marina Meila and Xiaotong Shen, editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2 of *Proceedings of Machine Learning Research*, pages 436–443, San Juan, Puerto Rico, 21–24 Mar 2007. PMLR.

[23] Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2012.

[24] Joel Veness, Tor Lattimore, Avishkar Bhoopchand, Agnieszka Grabska-Barwinska, Christopher Mattern, and Peter Toth. Online learning with gated linear networks. *CoRR*, abs/1712.01897, 2017.

[25] Vladimir V. V'yugin. Online learning in case of unbounded losses using the follow perturbed leader algorithm. *CoRR*, abs/1008.4232, 2010.