Online Prediction of Non-Stationary Time Series[*]

Research Report: Fall Semester 2019

Saad Christophe Gaillard Pierre

February, 2020

[*]Semester project course at ENS Paris

Contents

Abstract

This report is a description of my semester project with M. Gaillard. Its aim is to forecast nonstationary time series using online learning. We first present our motivation and ARROWS, an algorithm of the literature, which we will use as benchmark. We then present some online learning techniques along with some algorithms used for online predictions. Finally, we present our algorithm Fixed Experts which we test against ARROWS and we give some experimental results.

1 Introduction

In this section, we give an introduction to the problem we will treat in this report along with existing works and contributions.

1.1 Motivation

We consider here the problem of online forecasting of nonstationary time series.
A time series is a sequence of signals taken at successive equally spaced points in time. Given some time series, predicting the next values according to the previous ones can be very useful in the field of data analysis. For example, financial time series varies a lot in distribution due to constant changes in market conditions, using online forecasting techniques could be really useful to adapt to these unexpected variations.

More technically, in this project, we focus on a class of problems called nonparametric regression. We try to forecast a sequence $\theta_1, \theta_2, ..., \theta_T$ with $\theta_i \in \mathbb{R}^d$.
At each time step t, we try to forecast θ_t according to previous noised observations, we then receive a feedback $y_t = \theta_t + Z_t$ where Z_t is a zero-mean, independent subgaussian noise of variance σ^2.

1.2 Introduction to online stochastic optimization

We explain now the idea behind online stochastic optimization.

Online stochastic optimization is defined in a game-theoretic framework, it can be seen as a structured repeated game where:

1. At each iteration t, an online player chooses an element $\hat{\theta}_t$ from a usually convex bounded set of decisions $\kappa \in \mathbb{R}^d$ without knowing the outcome of his choice.
2. After commiting his decision, the adversary chooses a convex loss function $\ell_t : \kappa \mapsto \mathbb{R}$ from a bounded family of loss functions $\mathscr{L}$.
3. The player then suffer loss $\ell_t(\hat{\theta}_t)$ and adds this loss to his total sum of losses $L = \sum_{i=1}^{T} \ell_i(\hat{\theta}_i)$

Our goal is to minimize the sum of losses over a number T of iterations.

A performace measure for an algorithm is its **dynamic regret** defined in Equation 1. It is considered as "good" if it grows sublinearly as a function of T, this implies that on average the algorithm is performing as the best strategy we can have in hindsight.

$$R_T = \sup_{\{\ell_1,...,\ell_T\} \in \mathscr{L}^n} \left(\sum_{t=1}^{T} \ell_t(\hat{\theta}_t) - \min_{\theta \in \kappa} \sum_{t=1}^{T} \ell_t(\theta) \right) = o(T) \tag{1}$$

Another stronger dynamic regret is defined in Equation 2. We compare an algorithm to the best decision made at every iteration.

$$\tilde{R}_T = \sup_{\{\ell_1,...,\ell_T\} \in \mathscr{L}^n} \left(\sum_{t=1}^{T} \ell_t(\hat{\theta}_t) - \sum_{t=1}^{T} \min_{\theta \in \kappa} \ell_t(\theta) \right) \tag{2}$$

To evaluate the performance of an algorithm, we look at an upper bound of its worst case regret.

1.3 Existing Work

We take as benchmark ARROWS, an algorithm of the literature [1], described in Algorithm 2.

Its strategy to forecast the next data point θ_t is to compute the average of previous feedbacks y_t in the current bin. A bin starts at $t_h < t$ and end at $t-1$ and t_h is updated following a restart rule.

ARROWS rely on the following **assumptions**

1. The time horizon for the online learner is known to be T
2. The parametrer σ^2 of subgaussian noice is known
3. The total variation of the sequence $M = \sum_{i=2}^{T} ||\theta_{i+1} - \theta_i||$ is upper bounded by known positive C_T
4. Only the squared loss $\ell_t = (\hat{\theta}_t - \theta_t)^2$ is considered
5. $|\theta_1| \leq U$
6. $\theta_1, \theta_2, ..., \theta_T \in \mathbb{R}$

Theorem 1. *With high probability, ARROWS achieves an optimal strong dynamic regret of*

$$\tilde{R}_T = O(T^{1/3} C_T^{2/3} \sigma^{4/3} + U^2 + C_T^2 + \sigma^2)$$

Proof [1]

1.4 Contributions

We present a new algorithm to minimize Equation 2. The idea behind our algorithm is to aggregate many predictions computed using with different time horizons in order to adapt to non-stationarity. The latter are generated using an Online Convex Optimization algorithm, the Online Newton Step.

Our goal is to garentee a similar regret bound as ARROWS but with less assumptions.
No prior knowledge of the total variation of the time series and of the variance of the noise required, higher dimension time series and a generalization for a wider family of loss function, the exp-concave class.

2 Fixed Experts algorithm

If we could restart optimally an Online Newton Step algorithm, we would achieve optimal predictions. But since we have no prior knowledge of the time series, we can't predict the exact restart time.
We therefore start at each time $1 \leq k < t$ an Online Newton Step algorithm $ONS(k)$ with exp-concave loss function (defined below). These algorithms run in parallel and we aggregate the results using the prediction with expert advice model.

Assumptions

1. The loss functions $\ell_t : \kappa \subset \mathbb{R}^d \to \mathbb{R}$ are λ-expconcave
2. The gradient of the loss is bounded by a constant G, $\|\nabla \ell_t\|_\infty \leq G$
3. κ is a convex subset of $\mathbb{R}^d$ with diameter bounded by D (i.e., $\sup_{\theta,\theta' \in \Theta} \|\theta - \theta'\|_1$)

2.1 Exp-concave functions

For the rest of this report, we consider the following class of convex loss functions, which is wider than the strongly convex class.

Definition 1. *A convex function $f : \mathbb{R}^n \mapsto \mathbb{R}$ is defined to be λ-exp-concave over $\kappa \subseteq \mathbb{R}^n$ if the function $g : \kappa \mapsto \mathbb{R}$ with $g(x) = e^{-\lambda f(x)}$ is concave.*
The larger the value of λ, the more stringent the assumption of exp-concavity is.

For example, the squared loss function on interval $[-B, B]$ is $\frac{1}{2B}$-exp-concave.

2.2 Online Newton Step (ONS)

Online newton step is the Online Convex Optimization algorithm we use for making predictions. $ONS(k)$ starts running at time k to the end of the time series and at each time t, it generates a prediction θ_t^k.
Online newton step is an **second order** method which will take into consideration the second order derivative. But since the Hessian is too expensive to compute, this OCO algorithm will be based on the **quasi-newton method** which uses an approximation of the hessian. Algorithm 1 describes how it operates.

Algorithm 1 $ONS(k)$

```
1: Input convex set $\kappa \subset \mathbb{R}^d$, $T$, $\theta_k^k \in \kappa$, step size $\eta$, an initial matrix $A_0 = \frac{1}{\eta^2 D^2} I$
2: for $t = k$ to $T$
3:     Predict $\theta_t^k$
4:     Observe $\ell_t(\theta_t^k)$ and $\nabla \ell_t(\theta_t^k)$
5:     Update $A_t = A_{t-1} + \nabla \ell_t(\theta_t^k) \nabla \ell_t(\theta_t^k)^\top$
6:     Project and Update:    $\theta_{t+1}^k = \Pi_\kappa^{A_t}(\theta_t^k - \frac{1}{\eta} A^{-1} \nabla l_t(\theta_t^k))$    $\Pi_\kappa^A(y) := \underset{x \in \kappa}{\arg\min}\ (y-x)^\top A(y-x)$
7: end for
```

Theorem 2. *For λ-exp-concave loss functions ℓ_t, ONS with $\eta = \frac{1}{2}\min\{\frac{1}{4GD}, \lambda\}$ grantees a regret bound of*

$$R_T \leq \frac{1}{2\eta} + \frac{d \log(T)}{2\eta}$$

where d is the dimension of θ_t.
Proof [3, Thm 4.4]

2.3 Prediction with expert advice

At each time t, our algorithm seeks the best combination of $ONS(k)$ predictions $\theta_t^k \quad 1 \leq k < t$.
We present here a technique widely used in online learning, the prediction with expert advice model.

Let us imagine a game where an online player needs to make a prediction $\hat{\theta}_t$ and he has the assistance of K advisers. The player has to choose at iteration t the best advice.
Each expert makes at each iteration a prediction $\theta_t^k \in \kappa$.
A natural forecasting strategy would be a weighted average of experts' predictions. We give to each expert a weight w_t^k which represents the quality of his advice.
The algorithm then makes prediction $\hat{\theta}_t = \sum_{k \in K} w_t^k \theta_t^k$ and $\hat{\theta}_t \in \kappa$ since $\hat{\theta}_t$ is a convex combination of $\theta_t^k \in \kappa, 1 \leq k \leq K$

See Algorithm 3

2.4 Update methods

We give in this sections an example of experts predictions aggregation method, the exponentially weighted forecaster.

Exponentially weighted forecasters are of the form

$$w_{t+1}^k = \frac{w_t^k e^{-\eta \ell(\theta_t^k)}}{\sum_{i=1}^K w_t^i e^{-\eta \ell(\theta_t^i)}}$$

The weight of an expert decreases exponentially with respect to the total sum of his losses.

Remark 1 (Gradient trick)**.** *If ℓ is differentiable, the gradient trick is a technique where we approximate the loss $\ell(x)$ by its first order approximation $\nabla \ell(x).x$. The intuition behind this technique is that $\nabla \ell(x).x$ is big if the expert's advice x points in the direction of a large increase of the loss function. We would like to attribute a lower weight to this expert.*

An extension of this forecaster is BOA (with gradient trick), the update method we will consider. The algorithm updates the weight of expert k with learning rate $\eta_{k,t}$ at time t as follows:

$$w_{t+1}^k = \frac{\eta_{k,t} e^{\eta_{k,t} \nabla \ell_t(\hat{\theta}_t)^T (\hat{\theta}_t - \theta_t^k) - \eta_{k,t}^2 (\nabla \ell_t(\hat{\theta}_t)^T (\hat{\theta}_t - \theta_t^k))^2} w_t^k}{\sum_{k \in E} \eta_{k,t} e^{\eta_{k,t} \nabla \ell_t(\hat{\theta}_t)^T (\hat{\theta}_t - \theta_t^k) - \eta_{k,t}^2 (\nabla \ell_t(\hat{\theta}_t)^T (\hat{\theta}_t - \theta_t^k))^2} w_t^k}$$

Theorem 3. *(Theorem 1 of [2]) If losses $\ell_t(\theta_t^k) \in [0,1]$ and with learning rates $\eta_k \le 0.5$, BOA garentees:*

$$\sum_{t=1}^T \ell_t(\hat{\theta}_t) - \ell_t(\theta_t^k) \le \frac{\log K}{\eta_k} + \eta_k \sum_{t=1}^T (\ell_t(\hat{\theta}_t) - \ell_t(\theta_t^k))^2$$

Proof in [2]

2.5 Description

We seek to forecast non-stationary sequences of length T

- We take as set of experts:
$$E = \{k \mid 1 \le k < T\}, \quad |E| = K = T - 1$$
- At each time t, all experts make a prediction θ_t^k and a confidence $I_t^k \in \{0,1\}$ with
$$I_t^k = \begin{cases} 1 & \text{if } k \le t \\ 0 & \text{otherwise} \end{cases}$$
- At time t, θ_t^k is obtained by running an $ONS(k)$
- We observe for each expert a loss $\ell_t(\theta_t^k)$ and gradient $\nabla \ell_t(\theta_t^k)$ as well as the loss of the algorithm $\ell_t(\hat{\theta}_t)$ and its gradient $\nabla \ell_t(\hat{\theta}_t)$
- At each time t, the algorithm makes prediction
$$\hat{\theta}_t = \sum_{k \in E} p_t^k \theta_t^k$$
with $p_t^k = \frac{w_t^k I_t^k}{\sum_{k \in E} w_t^k I_t^k} \in \Delta_{|E|}$

Algorithm 5 gives a description of the algorithm.

2.6 Regret Bound

We try to prove a strong dynamic reget bound for Fixed Experts. We believe that

$$\tilde{R}_T = O(M \log(T))$$

with $M = \sum_{t=1}^{T-1} \|\theta_t - \theta_{t+1}\|_1$ is the total variation of the sequence.
From Theorem 1, ARROWS garentees $O(\sigma^{4/3} M^{2/3} T^{1/3})$. However, it only works for the squared loss function and the total variation of the time series M along with the variance of the noise σ^2 have to be known for the restart rule.
We will try to bound the regret of our algorithm in 4 steps.

1. By Theorem 2 we bound the dynamic regret of each $ONS(k)$. In other words, we bound the loss of each expert compared to the best decision we can have in hindsight.
2. With Theorem 3, we bound the loss of the algorithm $\hat{\theta}_t$ with respect to each expert.
3. We combine the results to obtain a dynamic regret bound of the algorithm. We bound the loss of the algorithm compared to the best decision we have in hindsight.
4. We transform Step 3 to obtain a strong dynamic regret bound by introducing the total variation of the sequence M. This step still has to be proved.

We now start our proof.
Defining $\alpha := \frac{1}{2}\min\{\frac{1}{4GD}, \lambda\}$, by exp-concavity we have from Lemma 4.3 of [3] for all $\theta, \theta' \in \kappa$

$$\ell_t(\theta) - \ell_t(\theta') \le \nabla \ell_t(\theta)^\top (\theta - \theta') - \frac{\alpha}{2}\left(\nabla \ell_t(\theta)^\top (\theta - \theta')\right)^2. \tag{3}$$

1. From Theorem 2, we bound the experts $ONS(k)$ regret for λ-expconcave losses:

$$\forall \theta \ \forall k \in E, \forall t \ge k \quad \sum_{i=k}^{t} \ell_i(\theta_i^k) \le \sum_{i=k}^{t} \ell_i(\theta) + \frac{d \log(t-k+1)+1}{2\alpha} \tag{4}$$

2. Applying BOA with learning rates $\eta_k := 2GD\alpha$ to combine the experts θ_t^k for $k \in E$ with losses $f_t(\theta) \in [0,1]$ (to be defined later), from Theorem 3 we have

$$\forall k \in E \quad \forall t \ge k \quad \sum_{i=k}^{t} f_i(\hat{\theta}_i) - \sum_{i=k}^{t} f_i(\theta_i^k) \le \frac{\log(t-1)}{\alpha GD} + \alpha GD \sum_{i=k}^{t} (f_i(\hat{\theta}_i) - f_i(\theta_i^k))^2. \tag{5}$$

where $t-1$ is the number of experts that were active before time t. Applying BOA with losses

$$f_i : \theta \mapsto \frac{\nabla \ell_i(\hat{\theta}_i)^\top (\theta - \hat{\theta}_i)}{2GD} + \frac{1}{2} \in [0,1],$$

we have

$$\forall k \in E \quad \forall t \ge k \quad \sum_{i=k}^{t} \nabla \ell_i(\hat{\theta}_i)^\top \left(\hat{\theta}_i - \theta_i^k\right) \le \frac{2\log(t-1)}{\alpha} + \frac{\alpha}{2} \sum_{i=k}^{t} (\nabla \ell_i(\hat{\theta}_i)^\top (\hat{\theta}_i - \theta_i^k))^2.$$

Now, using the exp-concavity of the losses (see Inequality (3)), it entails

$$\forall k \in E \quad \forall t \ge k \quad \sum_{i=k}^{t} \ell_i(\hat{\theta}_i) - \ell_i(\theta_i^k) \le \frac{2\log(t-1)}{\alpha}.$$

3. Combining with (4), we finally get for all $\theta \in \kappa$ and all $1 \le t_1 \le t_2 \le T$

$$\sum_{t=t_1}^{t_2} \ell_t(\hat{\theta}_t) - \sum_{t=t_1}^{t_2} \ell_t(\theta) \le \frac{(d+4)\log(t_2+1)+1}{2\alpha}$$

We manage to bound the regret of our algorithm by a term in $O(\log T)$

$$\forall \theta \quad R_T = \sum_{t=1}^{T} \ell_t(\hat{\theta}_t) - \sum_{t=1}^{T} \ell_t(\theta) \le \frac{(d+4)\log(T+1)+1}{2\alpha}$$

4. Still need to prove a strong dynamic regret bound, for all $\theta_1, \ldots, \theta_T \in \kappa$

$$\sum_{t=1}^{T} \ell_t(\hat{\theta}_t) - \ell_t(\theta_t) \le \frac{\square M(d+4)\log(t_2+1)+1}{2\alpha},$$

where $M = \sum_{t=1}^{T-1} \|\theta_t - \theta_{t+1}\|_1$.

3 Experiments

We implement and test both ARROWS and Fixed Experts. We use in both algorithms the squared loss and consider the root mean squared error as a measure of comparison.
We implement Fixed Experts with Online Gradient Descent described in Algorithm 4 instead of Online Newton Step and BOA as weights update method.

3.1 Sanity Check

We first check the behavior of our algorithm on a non-stationary time series generated by constant chunks of different values. This setting is more favorable to ARROWS since he uses an instant restart rule. We take $Z_t = 0$ for this experiment.

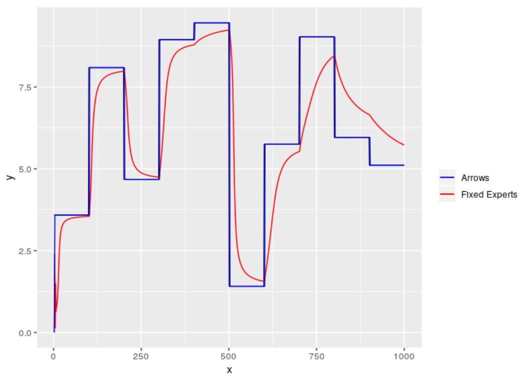

Figure 1: Time series predictions

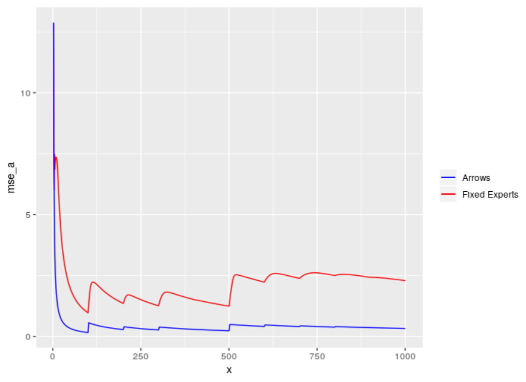

Figure 2: Root mean squared error

We can see in Figure 2 that the cumulative loss of Fixed Experts is greater than the cumulative loss of ARROWS. However, with an aggressive learning rate, Fixed Experts can reach the same performances. But we prefer to observe the behavior of the algorithm without prior knowledge of the time series.

3.2 Abrupt change

We test our algorithm against ARROWS on a non-stationary time series starting with an autoregressive moving average (ARMA) time series and ending with a linear function of time.

- $\theta_t = \sum_{i=1}^{5} \alpha_i \theta_{t-i} + \sum_{i=1}^{2} \beta_i \epsilon_{t-i} + \epsilon_t$ with $\alpha = (0.6, -0.5, 0.4, -0.4, 0.3), \beta = (0.3, -0.2), \epsilon_t \sim \mathcal{N}(0, 0.1)$
- $\theta_t = ct + c$, where c is a constant, the last value of the first generated chunk.

We take the gaussian noise $Z_t \sim \mathcal{N}(0,1)$.
In this setting, the upper bound of the total-variation of the time series C_T used by ARROWS is underestimated due to the second part of the time series and is not representative of the whole time series.

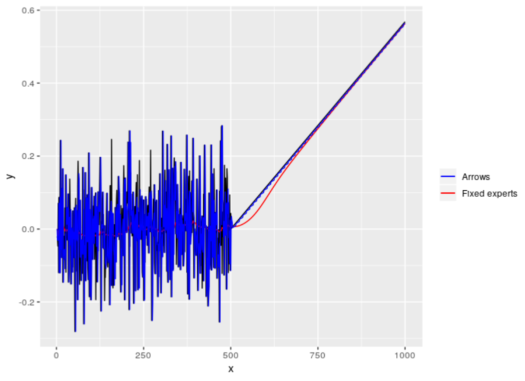

Figure 3: Time series predictions

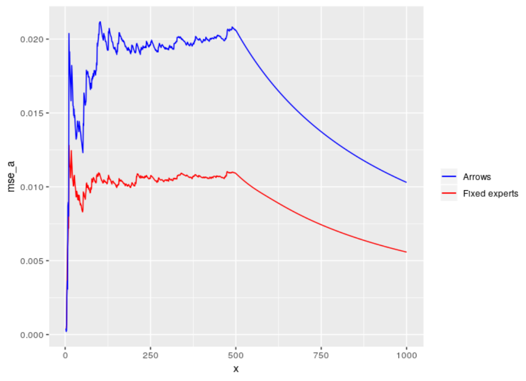

Figure 4: Root mean squared error

We can see in Figure 2 that the cumulative loss of Fixed Experts is less than the cumulative loss of ARROWS for the first part of the time series but converges slower when the values are linear as a function of the time.

3.3 Non-stationary time series

We finally test our algorithm on a non-stationary time series generated by different autoregressive time series $\theta_t = \sum_{i=1}^{h} \alpha_i \theta_{t-i} + \epsilon_t$ with $\epsilon_t \sim \mathcal{N}(0,1)$ and different parameters and horizons.

- $h = 10,\ \alpha_i = 1/11 \quad 1 \leq i \leq h$
- $h = 5,\ \alpha_i = 1/6 \quad 1 \leq i \leq h$
- $h = 15,\ \alpha_i = 1/16 \quad 1 \leq i \leq h$
- $h = 2,\ \alpha_i = 1/3 \quad 1 \leq i \leq h$

We add a gaussian noise $Z_t \sim \mathcal{N}(0,1)$

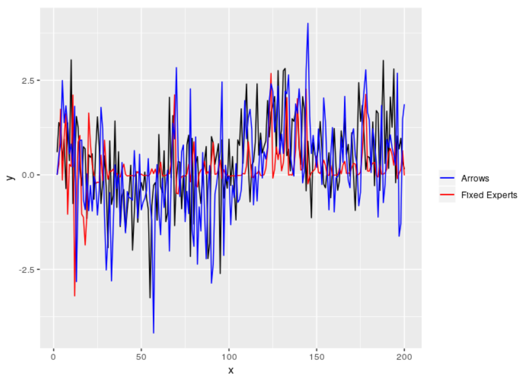

Figure 5: Time series predictions

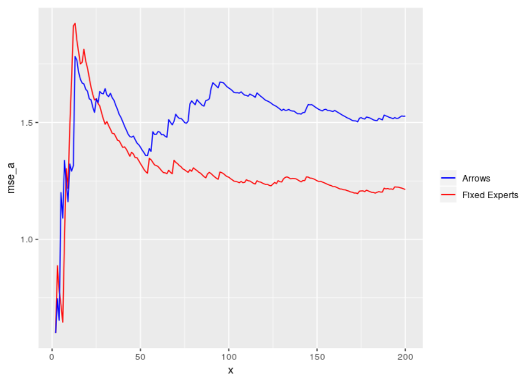

Figure 6: Root mean squared error

This time series is the most representative of real life data. Fixed Experts outperforms ARROWS and reacts better in slight changes in data distribution.

A ARROWS

- t_h : start time of the current bin, t : current time point
- $\bar{y}_{t_h:t}$: average of the y values for time steps t_h to t
- $pad_0(y_{t_h},...,y_t)$: vector $(y_{t_h} - \bar{y}_{t_h:t}, ..., y_t - \bar{y}_{t_h:t})^T$ zero-padded at the end till its length is a power of 2.
- $Y(x)$ where x is a sequence of values, denotes the element-wise soft thresholding of the sequence with threshold $\alpha\sqrt{\beta \log(T)}$
- H denotes the orthogonal discrete Haar wavelet transform matrix of proper dimensions
- Let $Hx = \alpha = [\alpha_1, ..., \alpha_k]^T$ where k being a power of 2 is the length of x. Then the vector $[\alpha_2, ..., \alpha_k]^T$ can be viewed as a concatenation of $log_2(k)$ contiguous blocks represented by $\alpha[l]$, $l = 0, ..., log_2(k)-1$. Each block $\alpha[l]$ at level l contains 2^l coefficients.

Algorithm 2 ARROWS (Adaptive Restarting Rule for Online averaging using Wavelet Shrinkage)

```
1:  Input Observed $y$ values, time horizon $T$, $\delta \in (0,1]$, total variation bound $C_T$ , a hyper-parameter $\beta > 6$
2:  Initialize $t_h = 1$, $newBin = 1$, $y_0 = 0$
3:  for $t = 1$ to $T$
4:      Predict $\bar{x}_t^{t_h} \leftarrow \begin{cases} y_{t-1} & newBin = 1 \\ \bar{y}_{t_h:t-1} & \text{otherwise.} \end{cases}$
5:      Set $newBin = 0$
6:      Observe $y_t$ and suffer loss $(x_t^{t_h} - \theta_t)^2$
7:      Set $\hat{y} = pad_0(y_{t_h}, ..., y_t)$ and $k$ the padded length
8:      Set $\hat{\alpha}(t_h : t) = Y(H\hat{y})$
9:      Restart rule If $\frac{1}{\sqrt{k}} \sum_{l=0}^{log_2(k)-1} 2^{l/2} ||\hat{\alpha}(t_h : t)[l]||_1 > T^{-\frac{1}{3}} C_T^{\frac{1}{3}} \sigma^{\frac{2}{3}}$ then
10:         Set $newBin = 1$
11:         Set $t_h = t + 1$
```

B Prediction with expert advice

We give here the algorithm for the prediction with expert advice model.

Algorithm 3 Learning from exprerts advices

```
1: Initialization $\mathbf{w_1} = (\frac{1}{K}, ..., \frac{1}{K}) \in \mathbb{R}^{|K|}$,  and Total Loss $L = 0$
2: for $t = 1$ to $T$
3:     Predict $\hat{\theta}_t = \sum_{k \in K} w_t^k \theta_t^k$
4:     Suffer Loss $\ell_t(\hat{\theta}_t)$, $L = L + \ell_t(\hat{\theta}_t)$
5:     Update $\mathbf{w_{t+1}}$
6: return $L$
```

C Online Gradient Descent (OGD)

Another Online Convex Optimization algorithm is Online Gradient Descent, which we use in our experiments.

Online gradient descent, is an iterative optimization method of the **first order** which consists of iteratively moving the current point in the direction of the function's gradient to reach its extrema.

Algorithm 3 describes how it operates.

Algorithm 4 *OGD*(k)

```
1: **Input** convex set $\kappa$, $T$, $\theta_k^k \in \kappa$, step sizes $\{\eta_t\}$
2: **for** $t = k$ **to** $T$
3:     **Predict** $\theta_t^k$
4:     **Observe** $\ell_t(\theta_t^k)$ and $\nabla \ell_t(\theta_t^k)$
5:     **Project** and **Update:**    $\theta_{t+1}^k = \Pi_\kappa(\theta_t^k - \eta_t \nabla \ell_t(\theta_t^k))$    $\Pi_\kappa(y) := \underset{x \in \kappa}{\arg\min} \, ||y - x||_2$
6: **end for**
```

D Fixed Experts algorithm

Algorithm 5 gives a description of the algorithm with weights update method BOA and experts' predictions update with online gradient descent.

Algorithm 5 Fixed Experts

```
1: **Input** $\gamma_1, ..., \gamma_K > 0$,  $\eta_1, ..., \eta_K > 0$, $A_0 =$
2: **Initialization** $w_1 = (\frac{1}{K}, ..., \frac{1}{K}) \in \mathbb{R}^K$, $\theta_0^k = 0 \; \forall k \in E$
3: **For** $t = 1$ **to** $T$
4:     **Foreach** $1 \le k < T$
5:         **Set** $I_t^k = \begin{cases} 1 & \text{if } k \le t \\ 0 & \text{otherwise} \end{cases}$
6:         **Set** $p_t^k = \frac{w_t^k I_t^k}{\sum_{k \in E} w_t^k I_t^k}$
7:     **Predict** $\hat{\theta}_t = \sum_{k \in E} p_t^k \theta_t^k$
8:     **Observe** $\ell_t(\hat{\theta}_t)$, $\ell_t(\theta_t^k)$ and $\nabla \ell_t(\hat{\theta}_t)$, $\nabla \ell_t(\theta_t^k)$
9:     **Update BOA** $w_{t+1}^k = \frac{\eta_k e^{\eta_k \nabla \ell_t(\hat{\theta}_t)^T (\hat{\theta}_t - \theta_t^k) - \eta_k^2 (\nabla \ell_t(\hat{\theta}_t)^T (\hat{\theta}_t - \theta_t^k))^2 I_t^k} w_t^k}{\sum_{k \in E} \eta_k e^{\eta_k \nabla \ell_t(\hat{\theta}_t)^T (\hat{\theta}_t - \theta_t^k) - \eta_k^2 (\nabla \ell_t(\hat{\theta}_t)^T (\hat{\theta}_t - \theta_t^k))^2 I_t^k} w_t^k}$
10:    **Update Expert Predictions (OGD)**
               $$\theta_{t+1}^k = \theta_t^k - \gamma_k \nabla l_t^k(\theta_t^k) \quad \forall k \; k \le t$$
```

References

[1] Dheeraj Baby and Yu-Xiang Wang. *Online Forecasting of Total-Variation-bounded Sequences.* 2019. arXiv: 1906.03364 [cs.LG].

[2] Pierre Gaillard, Gilles Stoltz, and Tim Van Erven. *A Second-order Bound with Excess Losses.* 2014. arXiv: 1402.2044 [stat.ML].

[3] Elad Hazan. *Introduction to Online Convex Optimization.* 2019. arXiv: 1909.05207 [cs.LG].