Aggregation of sleeping predictors to forecast electricity consumption

Marie Devaine [1] [2] [3], Yannig Goude [2], and Gilles Stoltz [1] [4]

[1]École Normale Supérieure, Paris, France
[2]EDF R&D, Clamart, France
[3]Université Paris-Sud, Orsay, France
[4]HEC Paris, Jouy-en-Josas, France

Contents

Chapter 1

Introduction and context

1.1 Description of the framework

Data

The data set consists in hourly observations of the Slovaquian comsumption of electricity. The units are Megawatts and the period of reference is from 01/01/2005 to 12/31/2007. There are 35 base forecasters (experts). The experts do not necessarily output a prediction at each step, they express themselves only at certain moments. (Yet at each step at least one expert makes a prediction.) The framework is therefore called prediction with "sleeping experts".

Notations

Time rounds are indexed by $t = 1, 2, \ldots, n$. Experts are indexed by $i = 1, \ldots, N$. The active experts at round t are given by a subset $E_t \subset \{1, \ldots, N\}$. The observation at round t is denoted by y_t. The prediction of expert i at round t exists if and only if $i \in E_t$ and we denote it by $f_{i,t}$. For practical reasons, we set the prediction of expert i to zero when $i \notin E_t$. We can thus define the vector of the predictions of the experts at time t as $\mathbf{f}_t = (f_{1,t}, \ldots, f_{N,t})$.

At each round the master forecaster outputs a vector of weights (an element of $\mathbb{R}^N$), which he uses to form a linear prediction (to be compared to y_t). Two prototypical cases arise.

- Constrained prediction: for each t, the vector used for the prediction is a probability distribution over E_t. In this case we denote the vector of weights by $\mathbf{p}_t \in \mathcal{X}_{E_t}$. More precisely, $\mathbf{p}_t \in \mathbb{R}^N$ belongs to $\mathcal{X}_{E_t}$ if $p_{i,t} = 0$ for $i \notin E_t$, for all i, we have $p_{i,t} \geqslant 0$, and $\sum_{i \in E_t} p_{i,t} = 1$. (Note that in the classical framework $\mathcal{X}$ is simply the simplex over $\{1 \ldots N\}$.) In this constrained case, the master forecaster forms the prediction

$$\widehat{y}_t = \sum_{i \in E_t} p_{i,t} f_{i,t} = \mathbf{p}_t \cdot \mathbf{f}_t \ .$$

- Unconstrained prediction: the vector used for prediction at time t is possibly any vector $\mathbf{u}_t$ of $\mathbb{R}^N$. In this unconstrained case the master forecaster forms the prediction

$$\widehat{y}_t = \sum_{i \in E_t} u_{i,t} f_{i,t} = \mathbf{u}_t \cdot \mathbf{f}_t \ .$$

  Note that the second equality holds because of the convention that $f_{i,t} = 0$ for $i \notin E_t$.

An idea to deal with this "sleeping experts" framework may be to go back to the classical framework by considering a partition of the data set that depends on the E_t. Let K be the number of values taken by the E_t on the data (for $t \in \{1, \ldots, n\}$), we denote by $U_1, \ldots, U_K$ the corresponding values. In addition, we denote by k_t the index corresponding at the step t, so that $E_t = U_{k_t}$.

Criteria to assess the quality of a master forecaster

We use a root mean squared error criterion (RMSE). Formally, the RMSE of a sequence $\mathbf{v}_1^n = (\mathbf{v}_1, \dots, \mathbf{v}_n)$ of aggregation vectors is defined by

$$\mathrm{RMSE}(\mathbf{v}_1^n) = \sqrt{\frac{1}{n}\sum_{t=1}^{n}\left(\sum_{i\in E_t} v_{i,t} f_{i,t} - y_t\right)^2}.$$

We give below some lower bounds beyond which it is difficult to go.

1. The following bounds have counterparts in the classical framework,

$$B_{\text{oracle}} = \min_{(j_1,\dots,j_n)\in E_1\times\dots\times E_n} \mathrm{RMSE}\big((\delta_{j_1},\dots,\delta_{j_n})\big) = \min_{(j_1,\dots,j_n)\in E_1\times\dots\times E_n} \sqrt{\frac{1}{n}\sum_{t=1}^{n}(f_{j_t,t}-y_t)^2}$$

   or

$$B_{\mathbb{R}^N} = \inf_{\mathbf{v}\in\mathbb{R}^N} \mathrm{RMSE}\big((\mathbf{v},\dots,\mathbf{v})\big) = \inf_{\mathbf{v}\in\mathbb{R}^N} \sqrt{\frac{1}{n}\sum_{t=1}^{n}(\mathbf{v}\cdot\mathbf{f}_t - y_t)^2}.$$

   The second oracle has a counterpart in terms of a given probability distribution $\mathbf{p} \in \mathcal{X}$ (it is legally defined thanks to the convention that $f_{i,t} = 0$ for $i \notin E_t$). However, the $\mathbf{p}\cdot\mathbf{f}_t$ will be in general too biased. This is, by the way, already the case even with a general $\mathbf{v} \in \mathbb{R}^N$. The performance table below will show that the oracle $B_{\mathbb{R}^N}$ is not a desirable target to achieve.

2. A first family of oracles designed for the setting of sleeping experts is in terms of the partition of time depending on the the values of the E_t, first in terms of any linear combination,

$$B_{\text{ part}\mathbb{R}^N} = \inf_{(\mathbf{v}_1,\dots,\mathbf{v}_K)\in(\mathbb{R}^N)^K} \mathrm{RMSE}\big((\mathbf{v}_1,\dots,\mathbf{v}_K)\big) = \inf_{(\mathbf{v}_1,\dots,\mathbf{v}_K)\in(\mathbb{R}^N)^K} \sqrt{\frac{1}{n}\sum_{t=1}^{n}(\mathbf{v}_{k_t}\cdot\mathbf{f}_t - y_t)^2},$$

   then only with probability distributions,

$$B_{\text{ part}\mathcal{X}} = \min_{(\mathbf{q}_1,\dots,\mathbf{q}_K)\in\mathcal{X}_{U_1}\times\dots\times\mathcal{X}_{U_K}} \mathrm{RMSE}\big((\mathbf{q}_1,\dots,\mathbf{q}_K)\big) = \min_{(\mathbf{q}_1,\dots,\mathbf{q}_K)\in\mathcal{X}_{U_1}\times\dots\times\mathcal{X}_{U_K}} \sqrt{\frac{1}{n}\sum_{t=1}^{n}(\mathbf{q}_{k_t}\cdot\mathbf{f}_t - y_t)^2},$$

   and finally only with Dirac distributions,

$$B_{\text{part}} = \min_{(j_1,\dots,j_K)\in U_1\times\dots\times U_K} \mathrm{RMSE}\big((\delta_{j_1},\dots,\delta_{j_K})\big) = \min_{(j_1,\dots,j_K)\in U_1\times\dots\times U_K} \sqrt{\frac{1}{n}\sum_{t=1}^{n}\left(f_{j_{k_t},t}-y_t\right)^2}.$$

3. We can also take into account some renormalizations to deal with some experts being sleeping. In this respect, for $\mathbf{q} \in \mathcal{X}$ we denote by $\mathbf{q}(E_t) = \sum_{i \in E_t} q_i$ a renormalization factor and let $\mathbf{q}^{E_t} \in \mathcal{X}_{E_t}$ be the conditional distribution of $\mathbf{q}$ subject to E_t. It is defined from $\mathbf{q}$ by only keeping the components indexed by E_t and dividing them by $\mathbf{q}(E_t)$. By convention, $q^{E_t} = (0, \dots, 0)$ when $\mathbf{q}(E_t) = 0$.

   a) A first version of the oracle is

$$B^{(a)}_{\text{renorm}} = \min_{\mathbf{q} \in \mathcal{X}} \sqrt{\frac{1}{\sum_{t=1}^n \mathbb{I}_{\{q(E_t) \neq 0\}}} \sum_{t=1}^n \mathbb{I}_{\{q(E_t) \neq 0\}} \left(\mathbf{q}^{E_t} \cdot \mathbf{f}_t - y_t\right)^2}$$

   which implies a natural average at each round of active forecasters when taking $\mathbf{q}$ as the uniform distribution $(1/N, \dots, 1/N)$,

$$B^{(a)}_{\text{ave}} = \sqrt{\frac{1}{n} \sum_{t=1}^n \left(\frac{\sum_{j \in E_t} f_{j,t}}{|E_t|} - y_t \right)^2}.$$

   b) A second (more continuous) version is given by a RMSE criterion with non-equal weights to all losses,

$$B^{(b)}_{\text{renorm}} = \min_{\mathbf{q} \in \mathcal{X}} \sqrt{\frac{1}{\sum_{t=1}^n q\left(E_t\right)} \sum_{t=1}^n \mathbf{q}\left(E_t\right) \left(\mathbf{q}^{E_t} \cdot \mathbf{f}_t - y_t\right)^2}$$

   and the average at each round of active forecasters corresponding:

$$B^{(b)}_{\text{ave}} = \sqrt{\frac{1}{\sum_{t=1}^n |E_t|} \sum_{t=1}^n |E_t| \left(\frac{\sum_{j \in E_t} f_{j,t}}{|E_t|} - y_t \right)^2}.$$

   The versions a) and b) give the same bound when we take the minimum over Dirac distributions only,

$$B_{\text{best exp}} = \min_{j=1,\dots,N} \text{RMSE}((\delta_j, \dots, \delta_j)) = \min_{j=1,\dots,N} \sqrt{\frac{1}{\sum_{t=1}^n \mathbb{I}_{\{j \in E_t\}}} \sum_{t=1}^n \mathbb{I}_{\{i \in E_t\}} \left(f_{j,t} - y_t\right)^2}.$$

4. Instead of renormalizing on the restriction of $\mathbf{q} \in \mathcal{X}$ indexed by active experts, we may consider the Euclidian projection onto $\mathcal{X}_{E_t}$, which we denote by $\Pi_{\mathcal{X}_{E_t}}$. For $\mathbf{q} \in \mathcal{X}$, we denote by $\mathbf{q}_{|E_t}$ its restriction to the set E_t of active experts, defined as $\mathbf{q}_{|E_t} = \left(q_{i_1}, \dots, q_{i_{|E_t|}}\right)$, with $\left\{i_1, \dots, i_{|E_t|}\right\} = E_t$ and $i_1 < i_2 < \dots < i_{|E_t|}$. Finally, we define the projection P_t as follows: for all $\mathbf{q} \in \mathcal{X}$,

$$P_t(\mathbf{q}) = \Pi_{\mathcal{X}_{E_t}}\left(\mathbf{q}_{|E_t}\right).$$

   The corresponding oracle is

$$B_{\text{project}} = \min_{\mathbf{q} \in \mathcal{X}} \sqrt{\frac{1}{n} \sum_{t=1}^n \left(P_t(\mathbf{q}) \cdot \mathbf{f}_t - y_t\right)^2}.$$

   No simple reduction is obtained in this case when we take the minimum over Dirac distributions only, since $P_t(\delta_j) = \delta_j$ when $j \in E_t$ but $P_t(\delta_j)$ is the uniform distribution of $\mathcal{X}_{E_t}$ when $j \notin E_t$.

Notations used in the description of the algorithms

We denote by $\ell(\cdot,\cdot)$ the square loss $\ell(x,y) = (x-y)^2$. We need in the proofs a uniform bound on the losses; it exists when the observations and the predictions are bounded by B for instance, which is the case in practice. In this case, as the data are non negative, we can take B^2 as uniform bound on our losses. (The only difficulty is to exhibit a value for B in advance.)

For each round t, we denote by

$$\ell_t(\mathbf{v}) = \ell\left(\sum_{i\in E_t} v_i f_{i,t}, y_t\right)$$

the instantaneous loss of a vector $\mathbf{v} \in \mathbb{R}^N$ (possibly a probability distribution on E_t). When $\mathbf{v} = \delta_i$ is a Dirac mass on an active expert, i.e., $i \in E_t$, we simply write

$$\ell_{i,t} = \ell_t(\delta_i) = \ell(f_{i,t}, y_t).$$

The loss of the master forecaster at round t equals

$$\widehat{\ell}_t = \ell(\widehat{y}_t, y_t)$$

with the notations above.

All these quantities have cumulative counterparts. The cumulative loss of the forecaster is referred to as

$$\widehat{L}_n = \sum_{t=1}^{n} \widehat{\ell}_t \ .$$

The cumulative losses of the experts are less easy to define in the framework of sleeping experts. We consider

1. for $\mathbf{v} \in \mathbb{R}^N$,
$$L_n(\mathbf{v}) = \sum_{t=1}^{n} \ell_t(\mathbf{v}) \ ;$$

2. for $\mathbf{v}_1^K = (\mathbf{v}_1, \ldots, \mathbf{v}_K) \in \left(\mathbb{R}^N\right)^K$ (where $\mathbf{v}_1^K$ can possibly be a sequence of probability distributions on $U_1 \times \ldots \times U_K$)
$$L_n\left(\mathbf{v}_1^K\right) = \sum_{t=1}^{n} \ell_t\left(\mathbf{v}_{k_t}\right) \ ;$$

3. for a distribution $\mathbf{q} \in \mathcal{X}$
$$L'_n(\mathbf{q}) = \sum_{t=1}^{n} \mathbf{q}(E_t)\ell_t(\mathbf{q}^{E_t}) \ , \qquad L'_{i,n} = \sum_{t=1}^{n} \mathbb{I}_{\{i\in E_t\}}\ell_{i,t} \ ,$$
where the second definition follows from the first one by taking the probability distribution $\mathbf{q} = \delta_i$;

4. for a distribution $\mathbf{q} \in \mathcal{X}$
$$L''_n(\mathbf{q}) = \sum_{t=1}^{n} \ell_t(P_t(\mathbf{q})) \ .$$

Minimization of RMSE via the minimization of regret

We attempt to minimize the RMSE of the master forecaster by ensuring that it has a small regret, where the regret is defined

1. either as
$$\sup_{\mathbf{v}\in\mathbb{R}^N} R_n(\mathbf{v}) \qquad \text{where} \qquad R_n(\mathbf{v}) = \widehat{L}_n - L_n(\mathbf{v}) \; ;$$
2. or
$$\sup_{\mathbf{v}_1^K\in(\mathbb{R}^N)^K} R_n\left(\mathbf{v}_1^K\right) \qquad \text{where} \qquad R_n\left(\mathbf{v}_1^K\right) = \widehat{L}_n - L_n\left(\mathbf{v}_1^K\right) = \sum_{t=1}^{n} \ell_t\left(\mathbf{v}_{k_t}\right) \; ;$$
3. or
$$\sup_{\mathbf{q}\in\mathcal{X}} R'_n(\mathbf{q}) \qquad \text{where} \qquad R'_n(\mathbf{q}) = \sum_{t=1}^{n} \mathbf{q}\left(E_t\right)\widehat{\ell}_t - L'_n(\mathbf{q}) \; ;$$
in the case we restrict our attention in the previous definition to Dirac distributions only, we get
$$\max_{j=1,\dots,N} R'_n\left(\delta_j\right) \overset{\text{not.}}{=} \max_{j=1,\dots,N} R'_{j,n} \qquad \text{where} \qquad R'_{j,n} = \sum_{t=1}^{n} \mathbb{I}_{\{j\in E_t\}}\left(\widehat{\ell}_t - \ell_{j,t}\right) ;$$
4. or
$$\sup_{\mathbf{q}\in\mathcal{X}} R''_n(\mathbf{q}) \qquad \text{where} \qquad R''_n(\mathbf{q}) = \widehat{L}_n - L''_n(\mathbf{q}) \; .$$

Numerical values

For practical purposes we do not use the whole data set to build our forecasts. As predictions are made for the next 24 hours, we split our data set into 24 fixed-hour subsets and run 24 algorithms in parallel. In our examples and simulations, we chose the subset corresponding to noon. We denote by n the total number of prediction steps and by n_{12} the number of prediction in the noon data subset. (We index by a subscript 12 the quantities that refer to this subset only.) M is some typical order of magnitude for y_t.

n	n_{12}	M	N	K	K_{12}
26277	1095	700	35	74	74

Some standard RMSE are summarized for the noon data subset below. As argued above, $B_{\mathbb{R}^N}$ is high because of the bias induced by sleeping experts. (When a highly weighted expert is missing, the bias is important and the corresponding instantaneous loss is large.) We must therefore use the information of sleeping/active experts to get lower bounds smaller that this one.

B_{oracle}	$B_{\mathbb{R}^N}$	$B_{\text{part}\mathbb{R}^N}$	$B_{\text{part}\mathcal{X}}$	B_{part}	$B^{(a)}_{\text{renorm}}$
9.4	40.7	16.4	24.5	29.1	29.7

$B^{(a)}_{\text{ave}}$	$B^{(b)}_{\text{renorm}}$	$B^{(b)}_{\text{ave}}$	B_{project}	$B_{\text{best exp}}$
31.3	29.2	30.7	29.2	30.4

Global performance of the experts (results are very similar for each hour data set) is drawn in Figure 1.1. In Figure 1.2, we plot the global performance of each expert with respect to his percentage of activity.

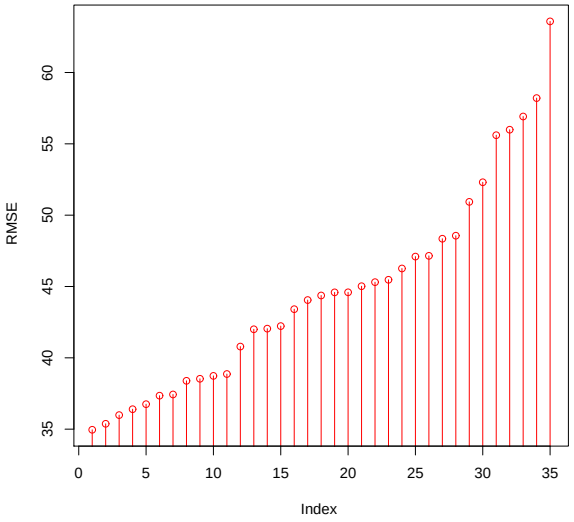

Figure 1.1: Performance of experts

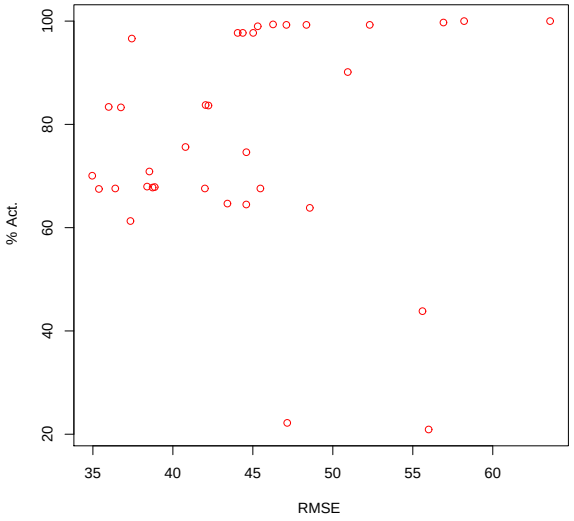

Figure 1.2: RMSE vs percentage of activity

Chapter 2

Base forecasters

2.1 Exponentially Weighted Average Beta-Forecaster

2.1.1 References

This forecaster is described in [BM05].

2.1.2 Theoretical bound

For β such that $\log \frac{1}{\beta}$ is of the order of $\frac{1}{B}\sqrt{\frac{\log N}{n}}$ we have

$$\max_{i\in\{1,\dots,N\}} R'_{i,n} \leqslant \mathcal{O}\left(B\sqrt{n\log N}\right) .$$

2.1.3 Interpretation and/or comments

This is variant of the exponentially weighted average forecaster in the context of sleeping experts. It is arguably less intuitive that the one presented in Section 2.2 but we can prove in a straightforward way a theoretical bound on its performance.

The parameter β used here should be thought of as $e^{-\eta}$ with $\eta > 0$. The weigths obtained with this algorithm are thus of the same form as the weights given by the algorithm of Section 2.2. However, the difference is that it is not the same quantity (the same regret) that is used in the exponential weighting, even though in both cases the aim is to control the original regret, $\max_i R'_{i,n}$.

2.1.4 Statement and implementation

The parameter β belongs to $]0,1[$. We introduce the β–regret: for $i \in \{1,\dots,N\}$ and $n \geqslant 1$,

$$R'_{\beta,i,n} = \sum_{t=1}^{n} \mathbb{I}_{\{i\in E_t\}}\left(\beta\widehat{\ell}_t - \ell_{i,t}\right).$$

Note that $R'_{i,n}$ is a β–regret for $\beta = 1$ (which is however a forbidden value for β in this section).

For $t \geqslant 1$, the convex weights p_t are defined as

$$p_{i,t} = \frac{\mathbb{I}_{\{i\in E_t\}}\beta^{-R'_{\beta,i,t-1}}}{\sum_{j\in E_t}\beta^{-R'_{\beta,j,t-1}}} .$$

By convention, $R'_{\beta,i,0} = 0$ for all $i = 1,\dots,N$.

The forecaster is implemented as follows.

Parameters: $\beta \in]0,1[$

Initialization: $R'_{\beta,i,0} = 0$ for all $i = 1, \dots, N$ and

$$p_{i,1} = \begin{cases} \dfrac{1}{|E_1|} & \text{if } i \in E_1; \\ 0 & \text{if } i \notin E_1. \end{cases}$$

For each round $t = 1, 2, \dots, n$,

(1) predict with $\widehat{y}_t = \mathbf{p}_t \cdot \mathbf{f}_t$;

(2) observe y_t and compute the regrets

$$R'_{\beta,i,t} = R'_{\beta,i,t-1} + \mathbb{I}_{\{i \in E_t\}} \left(\beta \widehat{\ell}_t - \ell_{i,t} \right)$$

for all $i = 1, \dots, N$;

(3) compute $\mathbf{p}_{t+1}$ as

$$p_{i,t+1} = \frac{\mathbb{I}_{\{i \in E_{t+1}\}} \beta^{-R'_{\beta,i,t}}}{\sum_{j \in E_{t+1}} \beta^{-R'_{\beta,j,t}}}$$

for all $i = 1, \dots, N$.

2.1.5 Proof of the theoretical bound

Without loss of generality, we can assume that $B = 1$. (If this is not the case, one simply considers the $\ell_{i,t}/B$ instead of the $\ell_{i,t}$.)

We introduce first some notations: for $i \in \{1, \dots, N\}$ and $t \geqslant 1$, we denote

$$w'_{i,t} = \beta^{-R'_{\beta,i,t-1}}, \quad w_{i,t} = \mathbb{I}_{\{i \in E_t\}} w'_{i,t}, \quad W_t = \sum_{i=1}^{N} w_{i,t} \quad \text{and} \quad W'_t = \sum_{i=1}^{N} w'_{i,t} \, .$$

In particular, we have

$$p_{i,t} = \frac{w_{i,t}}{W_t} \, .$$

By convexity of ℓ in its first argument, we have

$$\widehat{\ell}_t \leqslant \sum_{i=1}^{N} p_{i,t} \ell_{i,t} = \sum_{i=1}^{N} \frac{w_{i,t} \ell_{i,t}}{W_t} \qquad \text{and thus,} \qquad W_t \widehat{\ell}_t - \sum_{i=1}^{N} w_{i,t} \ell_{i,t} \leqslant 0 \, ,$$

an inequality we will need later.

We now bound W'_t by N for all t. We have $W'_0 = N$ and now show that the sequence $(W'_t)_{t \geqslant 0}$ decreases. We use that for $\beta \in]0,1[$ and $x \in [0,1]$, we have $\beta^x \leqslant 1 - (1-\beta)x$

and $\beta^{-x} \leqslant 1 + (1-\beta)x/\beta$ to get:

$$\begin{aligned}
W'_{t+1} &= \sum_{i=1}^{N} w'_{i,t} \beta^{-\mathbb{I}_{\{i \in E_t\}}\left(\beta \widehat{\ell}_t - \ell_{i,t}\right)} \\
&\leqslant \sum_{i=1}^{N} w'_{i,t} (1 - (1-\beta)\ell_{i,t}\mathbb{I}_{\{i \in E_t\}}) \left(1 + (1-\beta)\beta \widehat{\ell}_t \mathbb{I}_{\{i \in E_t\}}/\beta\right) \\
&\leqslant \sum_{i=1}^{N} w'_{i,t} \left(1 - (1-\beta)\mathbb{I}_{\{i \in E_t\}}\left(\ell_{i,t} - \widehat{\ell}_t\right)\right) \\
&\leqslant \sum_{i=1}^{N} w'_{i,t} + (1-\beta) \underbrace{\left(W_t \widehat{\ell}_t - \sum_{i=1}^{N} w_{i,t}\ell_{i,t}\right)}_{\leqslant 0} \\
&\leqslant W'_t \, .
\end{aligned}$$

In particular, we have that $w'_{i,t+1} \leqslant N$ for all $t \geqslant 0$. Since $w'_{i,t+1} = \beta^{-R'_{\beta\, i,t}}$, we have

$$R'_{\beta,i,t} \leqslant \frac{\log N}{\log \frac{1}{\beta}} \, ,$$

which rewrites as

$$\sum_{t=1}^{n} \mathbb{I}_{\{i \in E_t\}} \widehat{\ell}_t \leqslant \frac{\sum_{t=1}^{n} \mathbb{I}_{\{i \in E_t\}} \ell_{i,t} + \frac{\log N}{\log \frac{1}{\beta}}}{\beta} \leqslant \sum_{t=1}^{n} \mathbb{I}_{\{t \in E_t\}} \ell_{i,t} + \left(n\left(\frac{1}{\beta} - 1\right) + \frac{\log N}{\log \frac{1}{\beta}}\right).$$

Optimizing in β (by taking the parametrization $\beta = e^{-\eta}$ and optimizing in η) we get the claimed bound.

2.1.6 Performance

$1-\beta$	$1-$ 1e-3	5e-1	1e-4	1e-5	1e-6	$1-\beta^\star =$ 4e-6
RMSE	31.6	31.7	32.7	30.9	30.8	30.5

Forward note: We obtain the same performance as with the algorithm of Section 2.2. Indeed, setting $\beta = e^{-\eta}$, the two algorithms can be seen equivalent for sufficiently small values of η. These values are the one which we are interested in and for those, β is close to 1. Thus, the table above is the same as the one which will be given in Section 2.2 for small values of η (for which $1-\beta \approx \eta$).

2.1.7 Graphical evolution of the weights

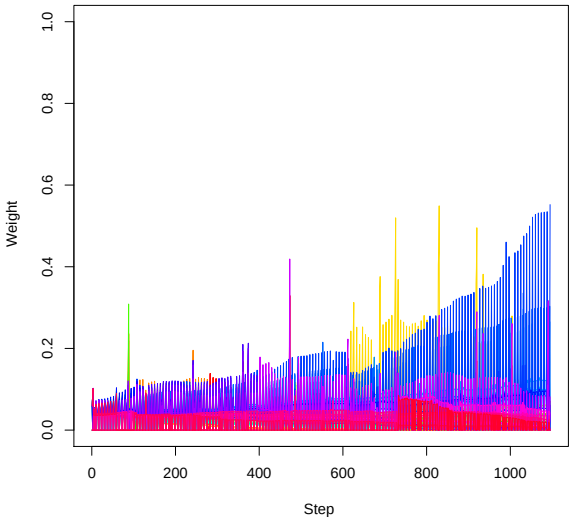

Figure 2.1: Evolution of the weights for $\beta = 1 - 4e\text{-}6$

2.2 Exponentially Weighted Average Forecaster

2.2.1 References

This is an adaptation of the forecaster of Section 2.1. We designed it and there is no previous occurrence of it in the literature.

2.2.2 Theoretical bound

For η of the order of $\frac{1}{B}\sqrt{\frac{\log N}{n}}$, we have

$$\max_{i\in\{1,\dots,N\}} R'_{i,n} \leqslant \mathcal{O}\left(B\sqrt{n\log N}\right) .$$

2.2.3 Interpretation and/or comments

We claim that the forecaster introduced below is more natural than the one of Section 2.1 since it is based on the regret and not on a variant of it. Note that with respect to the standard (i.e., non-sleeping) version of the exponentially weighted average forecaster, it cannot be defined only in terms of the cumulative losses of the experts. Here, unlike the standard case, the loss of the master forecaster in the numerator does not cancel out with corresponding terms in the denominator of the expression defining the weights.

2.2.4 Statement and implementation

The parameter η belongs to $]0,+\infty[$. For $t \geqslant 1$, the convex weights $\mathbf{p}_t$ are defined as

$$p_{i,t} = \frac{\mathbb{I}_{\{i\in E_t\}}\, e^{\eta R'_{i,t-1}}}{\sum_{j\in E_t} e^{\eta R'_{j,t-1}}} .$$

By convention $R'_{i,0} = 0$ for all $i = 1,\dots,N$.

The forecaster is implemented as follows.

Parameters: $\eta > 0$

Initialization: $R'_{i,0} = 0$ for all $i = 1, \dots, N$ and

$$p_{i,1} = \begin{cases} \dfrac{1}{|E_1|} & \text{if } i \in E_1; \\ 0 & \text{if } i \notin E_1. \end{cases}$$

For each round $t = 1, 2, \dots, n$

(1) predict with $\widehat{y}_t = \mathbf{p}_t \cdot \mathbf{f}_t$;

(2) observe y_t and compute the regrets

$$R'_{i,t} = R'_{i,t-1} + \mathbb{I}_{\{i \in E_t\}} \left(\widehat{\ell}_t - \ell_{i,t} \right)$$

for all $i = 1, \dots, N$;

(3) compute $\mathbf{p}_{t+1}$ as

$$p_{i,t+1} = \frac{\mathbb{I}_{\{i \in E_{t+1}\}} \, e^{\eta R'_{i,t}}}{\sum_{j \in E_{t+1}} e^{\eta R'_{j,t}}}$$

for all $i = 1, \dots, N$.

2.2.5 Proof of theoretical bound

We do not provide a direct proof but derive rather the bound from the one of Section 2.1. In particular we use that taking $\beta = e^{-\eta} \in]0, 1[$,

$$e^{\eta R'_{i,n}} = \beta^{-R'_{\beta,i,n}} \times e^{\eta(1-\beta)\sum_{t=1}^{n} \widehat{\ell}_t \mathbb{I}_{\{i \in E_t\}}} = w'_{i,n} \times e^{\eta(1-\beta)\sum_{t=1}^{n} \widehat{\ell}_t \mathbb{I}_{\{i \in E_t\}}}$$

where we used the notations of Section 2.1.5 for the two equalities. We know from Section 2.1.5 that $w'_{i,n} \leqslant N$ for all $i \in \{1 \dots, n\}$, so that

$$e^{\eta R'_{i,n}} \leqslant N \, e^{\eta(1-\beta)\sum_{t=1}^{n} \widehat{\ell}_t \mathbb{I}_{\{i \in E_t\}}} \, .$$

Using in addition $e^{-\eta} \geqslant 1 - \eta$, we thus get for $\eta > 0$

$$\eta R'_{i,n} \leqslant \log N + \eta(1 - e^{-\eta}) \sum_{t=1}^{n} \widehat{\ell}_t \mathbb{I}_{\{i \in E_t\}} \leqslant \log N + \eta^2 \sum_{t=1}^{n} \widehat{\ell}_t \mathbb{I}_{\{i \in E_t\}} \, .$$

Rearranging and bounding the sum in a crude but uniform manner by n,

$$R'_{i,n} \leqslant \frac{\log N}{\eta} + \eta B^2 n \, .$$

The claimed bound is obtained by taking $\eta = \sqrt{\log N / (B^2 n)}$,

$$R'_{i,n} \leqslant 2B\sqrt{n \log N} \, .$$

2.2.6 Performance

η	1e-8	1e-7	1e-6	1e-5	1e-4	$\eta^\star = 4e\text{-}6$
RMSE	31.3	31.2	30.8	30.9	32.7	30.5

2.2.7 Graphical evolution of the weights

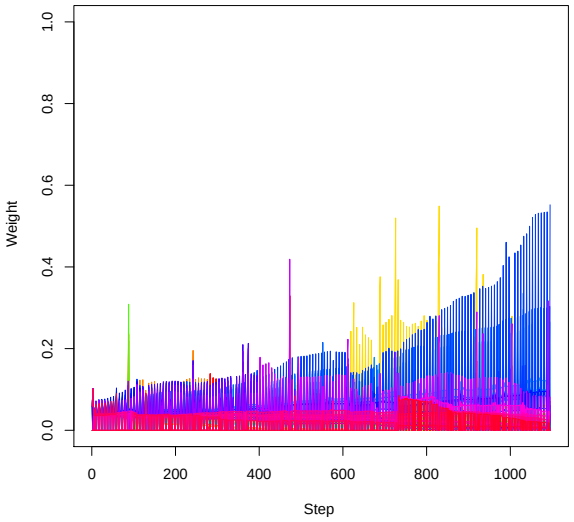

Figure 2.2: Evolution of the weights for $\eta = 4e\text{-}6$

2.3 Exponential Gradient Forecaster

2.3.1 References

This forecaster follows from the one of Section 2.2 thanks to the consideration of the gradients of the losses, see [CBL06, Section 2.5].

2.3.2 Theoretical bound

For η of the order of $\frac{1}{C}\sqrt{\frac{\log N}{n}}$, we have

$$\max_{i\in\{1,\dots,N\}} R'_{i,n} \leqslant \mathcal{O}\left(C\sqrt{n\log N}\right) \ ,$$

where C is a constant such that $-C \leqslant \left(\nabla\widehat{\ell}_t\right)_i \leqslant C$ for all $i = 1,\dots,N$ and $t = 1,\dots,n$. We can take $C = 2B^2$.

2.3.3 Interpretation and/or comments

The forecaster below is the exponentiated gradient version of the forecaster of Section 2.2, in which the cumulative regrets appearing in the exponent in the definition of the weights are replaced by some gradient-based upper bound.
We only adapt here the forecaster of Section 2.2. We also considered a gradient version of the algorithm of Section 2.1 but again, it obtains about the same performance as the one presented in this section. This is why we do not write a dedicated section for it.

2.3.4 Statement and implementation

The parameter η belongs to $]0,+\infty[$. We use $\nabla\widehat{\ell}_t$ to denote the gradient of the convex function $\mathbf{v}\mapsto\ell_t(\mathbf{v})$ taken in $\mathbf{p}_t$, that is,

$$\nabla\widehat{\ell}_t = 2(\widehat{y}_t - y_t)\mathbf{f}_t \ .$$

For $t \geqslant 1$, the vector of weights $\mathbf{p}_t$ is defined component-wise as

$$\begin{aligned} p_{i,t} &= \frac{\mathbb{I}_{\{i\in E_t\}}\exp\left(\eta\sum_{s=1}^{t-1}\mathbb{I}_{\{i\in E_s\}}\nabla\widehat{\ell}_s\cdot(\mathbf{p}_t-\delta_i)\right)}{\sum_{j\in E_t}\exp\left(\eta\sum_{s=1}^{t-1}\mathbb{I}_{\{j\in E_s\}}\nabla\widehat{\ell}_s\cdot(\mathbf{p}_t-\delta_j)\right)} \\ &= \frac{\mathbb{I}_{\{i\in E_t\}}\exp\left(2\eta\sum_{s=1}^{t-1}\mathbb{I}_{\{i\in E_s\}}(\widehat{y}_s-y_s)(\widehat{y}_s-f_{i,s})\right)}{\sum_{j\in E_t}\exp\left(2\eta\sum_{s=1}^{t-1}\mathbb{I}_{\{j\in E_s\}}(\widehat{y}_s-y_s)(\widehat{y}_s-f_{j,s})\right)} \ . \end{aligned}$$

By convention an empty sum is nul.

The forecaster is implemented as follows.

Parameters: learning rate $\eta > 0$

Initialization: $\widetilde{R}_{i,0} = 0$ for all $i = 1, \dots, N$ and

$$p_{i,1} = \begin{cases} \dfrac{1}{|E_1|} & \text{if } i \in E_1 \\ 0 & \text{if } i \notin E_1 \end{cases}$$

For each round $t = 1, 2, \dots, n$

(1) predict with $\widehat{y}_t = \mathbf{p}_t \cdot \mathbf{f}_t$;

(2) observe y_t and compute the pseudo-regrets

$$\widetilde{R}_{i,t} = \widetilde{R}_{i,t-1} + 2\mathbb{I}_{\{i \in E_t\}}(\widehat{y}_t - y_t)(\widehat{y}_t - f_{i,t})$$

for all $i = 1, \dots, N$;

(3) compute $\mathbf{p}_{t+1}$ as

$$p_{i,t+1} = \frac{\mathbb{I}_{\{i \in E_{t+1}\}} e^{\eta \widetilde{R}_{i,t}}}{\sum_{j \in E_{t+1}} e^{\eta \widetilde{R}_{j,t}}}$$

for all $i = 1, \dots, N$.

2.3.5 Proof of the theoretical bound

Since $\mathbf{v} \mapsto \ell_t(\mathbf{v})$ is convex and differentiable, we can upper bound the instantenous regrets as

$$\ell(\widehat{y}_t, y_t) - \ell(f_{i,t}, y_t) = \ell_t(\mathbf{p}_t) - \ell_t(\delta_i) \leqslant \nabla \widehat{\ell}_t \cdot (\mathbf{p}_t - \delta_i) = \widetilde{\ell}(\widehat{y}_t, y_t) - \widetilde{\ell}(f_{i,t}, y_t) \ ,$$

where we denoted, for all probability distributions $\mathbf{q} \in \mathcal{X}_{E_t}$, the linearized losses by

$$\widetilde{\ell}(\mathbf{q} \cdot \mathbf{f}_t, y_t) = \nabla \widehat{\ell}_t \cdot \mathbf{q} \ .$$

Summing up the bound, we get

$$R'_{i,n} = \sum \mathbb{I}_{\{i \in E_t\}} \left(\widehat{\ell}_t - \ell_{i,t} \right) \leqslant \sum_{t=1}^{n} \mathbb{I}_{\{i \in E_t\}} \left(\widetilde{\ell}(\widehat{y}_t, y_t) - \widetilde{\ell}(f_{i,t}, y_t) \right) \leqslant \frac{\log N}{\eta} + nC\eta$$

for all $i = 1, \dots, N$, where C is some bound on the $\widetilde{\ell}$, for instance, $C = 2B^2$. We used here the result of Section 2.2.5, up to the replacement of the losses by their linearized counterparts. We can do so because we only use in this proof and in the present one that the losses are nonnegative functions convex in their first arguments.

2.3.6 Performance

η	1e-7	1e-6	1e-5	1e-4	1e-3	$\eta^\star = 1.1\text{e-}4$
RMSE	31.3	30.9	29.8	28.2	33.5	28.2

2.3.7 Graphical evolution of the weights

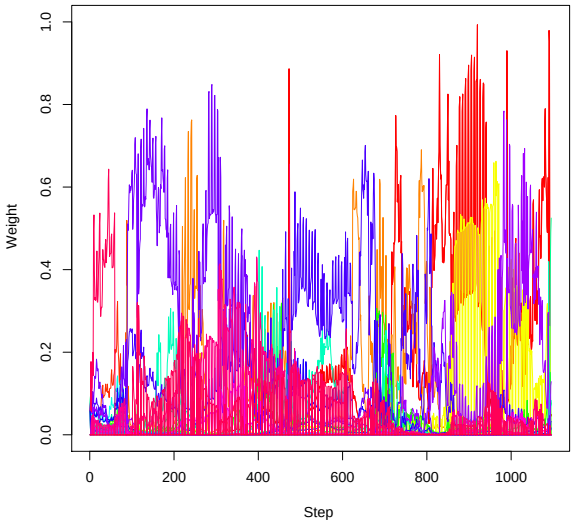

Figure 2.3: Evolution of the weights for $\eta = 1.1\text{e-}4$

2.4 Exponential Gradient Forecaster Partition

2.4.1 References

This forecaster is an adaptation from the EG forecaster studied in [CB99].

2.4.2 Theoretical bound

For η of the order of $\frac{1}{B^2K}\sqrt{\frac{\log N}{n}}$, we have

$$\sup_{\mathbf{q}_1^K \in \mathcal{X}_{U_1} \times \ldots \times \mathcal{X}_{U_K}} R_n\left(\mathbf{q}_1^K\right) \leqslant \mathcal{O}\left(CK\sqrt{n \log N}\right)$$

where $\mathbf{q}_1^K$ is a shorthand notation for $(\mathbf{q}_1, \ldots, \mathbf{q}_K)$ and C is a constant such that $-C \leqslant \left(\nabla \widehat{\ell}_t\right)_i \leqslant C$ for all $i = 1, \ldots, N$ and $t = 1, \ldots, n$. We can take $C = 2B^2$.

2.4.3 Interpretation and/or comments

This forecaster is obtained by running in parallel K classic exponential gradient algorithms on each subset U_k for $k \in \{1 \ldots, K\}$. The bound is obtain by summing up the base bounds of [CB99] on each subset. Note that if K is large, then the proposed bound is poor.

2.4.4 Statement and implementation

The parameter η belongs to $]0, +\infty[$.

For $t \geqslant 2$, we introduce the notation $U^t = U_{k_t} \cap \{1, \ldots, t-1\}$, which refers to the set of past rounds with the same sleeping configuration as the current one, i.e., the rounds indexed by $s \leqslant t-1$ and such that $E_s = U_{k_t}$.

For $t \geqslant 1$, the weight vector $\mathbf{p}_t$ is defined as

$$\begin{aligned} p_{i,t} &= \frac{\mathbb{I}_{\{i \in E_t\}} \exp\left(\eta \sum_{s=1}^{t-1} \mathbb{I}_{\{i \in E_s\}} \nabla \widehat{\ell}_s \cdot (\mathbf{p}_t - \delta_i)\right)}{\sum_{j \in E_t} \exp\left(\eta \sum_{s=1}^{t-1} \mathbb{I}_{\{j \in E_s\}} \nabla \widehat{\ell}_s \cdot (\mathbf{p}_t - \delta_j)\right)} \\ &= \frac{\mathbb{I}_{\{i \in E_t\}} \exp\left(2\eta \sum_{s \in U^t} (\widehat{y}_s - y_s)(\widehat{y}_s - f_{i,s})\right)}{\sum_{j \in E_t} \exp\left(2\eta \sum_{s \in U^t} (\widehat{y}_s - y_s)(\widehat{y}_s - f_{j,s})\right)} . \end{aligned}$$

By convention an empty sum is nul. The definition is legal because if $i \in E_t$ then by definition of U^t, one also has $i \in E_s$ for all $s \in U^t$.

This forecaster is implemented as follows.

Parameters: learning rate $\eta > 0$

For each round $t = 1, \dots, n$,

(1) if $U^t = \emptyset$

$$p_{i,t} = \begin{cases} \dfrac{1}{|E_t|} & \text{if } i \in E_t; \\ 0 & \text{if } i \notin E_t. \end{cases}$$

Otherwise,

$$p_{i,t} = \frac{\mathbb{I}_{\{i \in E_t\}} \exp\left(2\eta \sum_{s \in U^t} (\widehat{y}_s - y_s)(\widehat{y}_s - f_{i,s})\right)}{\sum_{j \in E_t} \exp\left(2\eta \sum_{s \in U^t} (\widehat{y}_s - y_s)(\widehat{y}_s - f_{j,s})\right)} \,,$$

for $i = 1, \dots, N$.

(2) predict with $\widehat{y}_t = \mathbf{p}_t \cdot \mathbf{f}_t$.

2.4.5 Performance

η	1e-6	1e-5	1e-4	1e-3	$\eta^\star =$ 7.4e-5
RMSE	31.3	30.8	29.9	37.1	29.8

2.4.6 Graphical evolution of the weights

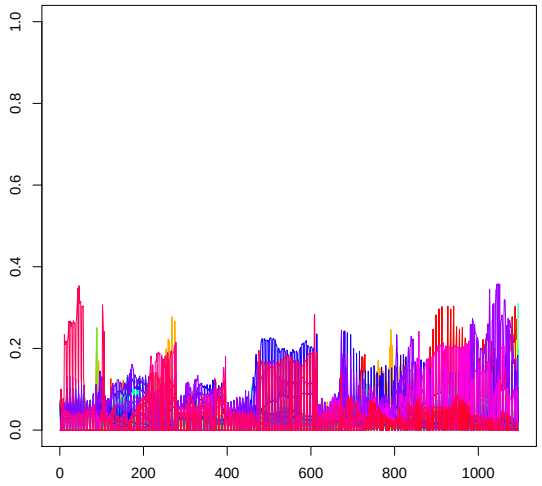

Figure 2.4: Evolution of the weights for η = 1.1e-4

2.5 Renormalized Exponential Gradient Forecaster

2.5.1 References

This forecaster is described in [FSS97].

2.5.2 Theoretical bound

For η of the order of $\frac{1}{C}\sqrt{\frac{\log N}{n}}$, we have

$$\sup_{q \in \mathcal{X}} R_n'(q) \leqslant \mathcal{O}\Big(C\sqrt{n \log N}\Big) \ ,$$

where C is a constant such that $-C \leqslant \left(\nabla \widehat{\ell}_t\right)_i \leqslant C$ for all $i = 1, \dots, N$ and $t = 1, \dots, n$. We can take $C = 2B^2$.

2.5.3 Interpretation and/or comments

This forecaster is another adaptation of the gradient-based forecasters to the setting of sleeping experts. The notion of regret minimized is not the same as in Section 2.3. However, their respective performance is similar. This is despite the fact that the two algorithms are quite different, as the graphical evolutions of the weights reveal.

2.5.4 Statement and implementation

The parameter η belongs to $]0, +\infty[$. This forecaster is implemented as follows.

Parameters: learning rate η
Initialization: $p_{i,1} = 1/N$ for $i = 1, \dots, N$
For each round $t = 1, 2, \dots, n$

(1) predict with $\widehat{y}_t = \mathbf{p}_t^{E_t} \cdot \mathbf{f}_t$

(2) observe y_t and compute $\mathbf{p}_{t+1}$ as follows:

$$p_{i,t+1} = \begin{cases} p_{i,t} e^{-2\eta f_{i,t}\left(\widehat{y}_t - y_t\right)} \dfrac{\sum_{j \in E_t} p_{j,t}}{\sum_{k \in E_t} p_{k,t} e^{-2\eta f_{k,t}\left(\widehat{y}_t - y_t\right)}} & \text{if } i \in E_t, \\ p_{i,t} & \text{if } i \notin E_t. \end{cases}$$

2.5.5 Proof of the theoretical bound

Since $\mathbf{v} \mapsto \ell_t(\mathbf{v})$ is convex and differentiable, we can upper bound the instantenous regrets as

$$\ell(\widehat{y}_t, y_t) - \ell\left(\mathbf{q}^{E_t}\right) = \ell_t\left(\mathbf{p}_t^{E_t}\right) - \ell\left(\mathbf{q}^{E_t}\right) \leqslant \nabla \widehat{\ell}_t \cdot \left(\mathbf{p}_t^{E_t} - \mathbf{q}^{E_t}\right) \ .$$

We thus have

$$R'_n(\mathbf{q}) \leqslant \sum_{t=1}^{n} \mathbf{q}(E_t)\, \nabla \widehat{\ell}_t \cdot \left(\mathbf{p}_t^{E_t} - \mathbf{q}^{E_t}\right) .$$

We now prove the following bound,

$$\nabla \widehat{\ell}_t \cdot \left(\mathbf{p}_t^{E_t} - \mathbf{q}^{E_t}\right) \leqslant \frac{\eta C^2}{2} + \frac{1}{\eta}\left(\mathcal{K}\left(\mathbf{q}^{E_t}, \mathbf{p}_t^{E_t}\right) - \mathcal{K}\left(\mathbf{q}^{E_t}, \mathbf{p}_{t+1}^{E_t}\right)\right)$$

where we denoted by $\mathcal{K}(\,\cdot\,,\,\cdot\,)$ the Kullback-Leibler divergence between two distributions. We recall that the latter is defined, for two probability distributions $\mathbf{p}$ and $\mathbf{q}$ over a set with R elements as

$$\mathcal{K}(\mathbf{q}, \mathbf{p}) = \sum_{i=1}^{R} q_i \log\left(\frac{q_i}{p_i}\right) .$$

The claimed bound actually follows from an application of the general lemma stated below together with the fact that by definition of the forecaster,

$$p_{i,t+1}^{E_t} = \frac{p_{i,t}^{E_t}\, e^{-\eta \gamma_{i,t}}}{\sum_{j=1}^{N} p_{i,t}^{E_t}\, e^{-\eta \gamma_{j,t}}} ,$$

for all $t \in \{1, \dots, n\}$ and $i = 1, \dots, N$, where we set

$$\gamma_{i,t} = 2(\widehat{y}_t - y_t) f_{i,t} = \left(\nabla \widehat{\ell}_t\right)_i .$$

Lemme 1. *Let $\mathbf{q}$, $\mathbf{p}$ be two probability distributions over a set with R elements, $\gamma \in \mathbb{R}^R$ be any R–dimensional real vector. Define a distribution $\mathbf{p}'$ as follows: for $i = 1, \dots, R$,*

$$p'_i = \frac{p_i\, e^{-\eta \gamma_i}}{\sum_{j=1}^{R} p_j\, e^{-\eta \gamma_j}} .$$

Then, denoting by D a bound such that $-D \leqslant \gamma_i \leqslant D$ for $i = 1, \dots, R$, one has

$$\eta\, \mathbf{p} \cdot \gamma - \eta \sum_{i=1}^{R} q_i \gamma_i - \frac{\eta^2 D^2}{2} \leqslant \mathcal{K}(\mathbf{q}, \mathbf{p}) - \mathcal{K}(\mathbf{q}, \mathbf{p}') . \tag{2.1}$$

Proof. We start with a chain of equalities,

$$\begin{aligned}
\mathcal{K}(\mathbf{q}, \mathbf{p}) - \mathcal{K}(\mathbf{q}, \mathbf{p}') &= \sum_{i=1}^{R} q_i \log\left(\frac{p'_i}{p_i}\right) \\
&= \sum_{i=1}^{R} q_i \log\left(\frac{p_i e^{-\eta \gamma_i}}{\sum_{j=1}^{R} p_j\, e^{-\eta \gamma_j}} \cdot \frac{1}{p_i}\right) \\
&= \sum_{i=1}^{R} -q_i \log\left(\sum_{j=1}^{R} p_j\, e^{-\eta \gamma_j}\right) + \sum_{i=1}^{N} q_i \log\left(e^{-\eta \gamma_i}\right) \\
&= -\sum_{i=1}^{R} q_i \log\left(\sum_{j=1}^{R} p_j\, e^{-\eta \gamma_j}\right) - \sum_{i=1}^{R} q_i \eta \gamma_i .
\end{aligned}$$

We have, first resorting to Hoeffding's lemma for each $i = 1, \dots, R$ and then using that $\sum_{i=1}^{R} q_i = 1$,

$$\sum_{i=1}^{R} q_i \log\left(\sum_{j=1}^{R} p_j \, e^{-\eta \gamma_j} p_j\right) \leqslant \sum_{i=1}^{R} q_i \left(-\eta \sum_{j=1}^{R} \gamma_j p_j + \frac{\eta^2 D^2}{2}\right) = -\eta \, \gamma \cdot \mathbf{p} + \frac{\eta^2 D^2}{2} \, .$$

□

We now get back to the main proof. So far, we have

$$R'_n(\mathbf{q}) \leqslant \sum_{t=1}^{n} \mathbf{q}(E_t) \left(\frac{\eta C^2}{2} + \frac{1}{\eta}\left(\mathcal{K}\left(\mathbf{q}^{E_t}, \mathbf{p}_t^{E_t}\right) - \mathcal{K}\left(\mathbf{q}^{E_t}, \mathbf{p}_{t+1}^{E_t}\right)\right)\right) .$$

In our case, as we have that $p_{i,t} = p_{i,t+1}$ if $i \notin E_t$ and therefore $\mathbf{p}_{t+1}(E_t) = \mathbf{p}_t(E_t)$, we derive

$$\begin{aligned} \mathbf{q}(E_t)\left(\mathcal{K}\left(\mathbf{q}^{E_t}, \mathbf{p}_t^{E_t}\right) - \mathcal{K}\left(\mathbf{q}^{E_t}, \mathbf{p}_{t+1}^{E_t}\right)\right) &= \sum_{i \in E_t} q_i \log\left(\frac{p_{i,t+1}}{p_{i,t}}\right) \\ &= \sum_{i=1}^{N} q_i \log\left(\frac{p_{i,t+1}}{p_{i,t}}\right) = \mathcal{K}(\mathbf{q}, \mathbf{p}_t) - \mathcal{K}(\mathbf{q}, \mathbf{p}_{t+1}) \, . \quad (2.2) \end{aligned}$$

Substituting, we get

$$R'_n(\mathbf{q}) \leqslant \sum_{t=1}^{n} \mathbf{q}(E_t) \left(\frac{\eta C^2}{2} + \frac{1}{\eta}(\mathcal{K}(\mathbf{q}, \mathbf{p}_t) - \mathcal{K}(\mathbf{q}, \mathbf{p}_{t+1}))\right) .$$

A telescoping summing has appeared and we are left with

$$R'_n(\mathbf{q}) \leqslant \frac{\eta n C^2}{2} + \frac{\mathcal{K}(\mathbf{q}, \mathbf{p}_1)}{\eta} \leqslant \frac{\eta n C^2}{2} + \frac{\log N}{\eta}$$

where the last equality proceeds from the choice of $\mathbf{p}_1$ as the uniform distribution. Optimizing in η we obtain the claimed bound.

2.5.6 Performance

η	1e-7	1e-6	1e-5	1e-4	1e-3	$\eta^\star =$ 1e-4
RMSE	31.3	30.9	29.8	28.2	34.7	28.2

2.5.7 Graphical evolution of the weights

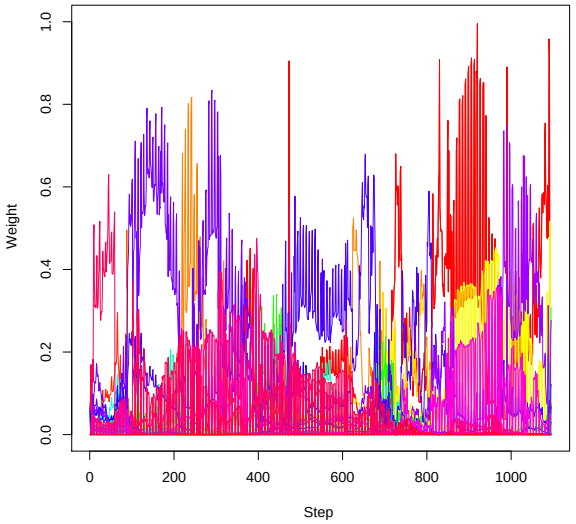

Figure 2.5: Evolution of the weights for $\eta =$ 1e-4

2.6 Zinkevich's forecasters

2.6.1 References

These forecasters are an adaptation of the ones proposed in [Zin03].

2.6.2 First forecaster: lazy projection

Theoretical bound

The aim of this forecaster is to minimize the regret defined in terms of the $R_n''(\mathbf{q})$, that is, to achieve a performance nearly as good as the oracle B_{project}. We do not provide any theoretical bound in the sleeping case for the time being.

Interpretation and/or comments

The forecaster below uses the standard version of the lazy Zinkevich's forecaster in the framework of sleeping experts. The original forecaster proceeds by projecting on the simplex of size N the weights obtained by a so-called gradient descent. Here, in the sleeping expert context, at each round the projection is performed on the subset of the weights indexed by the active experts.

Statement and implementation

The parameter η belongs to $]0, +\infty[$.

We first define a sequence of intermediate weights $\mathbf{w}_t$ as $\mathbf{w}_1 = (1/N, \dots, 1/N)$ and $\mathbf{w}_{t+1} = \mathbf{w}_t - \eta \mathbb{I}_t * \nabla \widehat{\ell}_t$, where $\mathbb{I}_t = \left(\mathbb{I}_{\{1 \in E_t\}}, \dots, \mathbb{I}_{\{N \in E_t\}}\right)$ and for two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^N$, $\mathbf{u} * \mathbf{v}$ is the term by term product $(u_1 v_1, \dots, u_N v_N)$. The final convex weights $\mathbf{p}_t$ output at round t are then given by

$$\mathbf{p}_t = P_t(\mathbf{w}_t)$$

where we use again the notations of Section 1.1.

We briefly recall how to project a real m-tuple $\mathbf{x} = (x_1, \dots, x_m) \in \mathbb{R}^m$ onto $\mathcal{X}_m$, the simplex of dimension m: let a be the unique real number such that

$$\sum_{i=1}^{m} (y_i + a)_+ = 1 \ .$$

Then the Euclidian projection of $\mathbf{x}$ onto $\mathcal{X}_m$ is given by

$$\Pi_{\mathcal{X}_m}(\mathbf{x}) = \left((x_1 + a)_+, \dots, (x_m + a)_+\right) \ .$$

The forecaster is implemented as follows.

Parameters: $\eta > 0$
Initialization:

$$p_{i,1} = \begin{cases} \dfrac{1}{|E_1|} & \text{if } i \in E_1; \\ 0 & \text{if } i \notin E_1. \end{cases}$$

For each round $t = 1, 2, \ldots, n$,

(1) predict with $\widehat{y}_t = \mathbf{p}_t \cdot \mathbf{f}_t$;

(2) obtain y_t and perform the update

$$\mathbf{w}_{t+1} = \mathbf{w}_t - 2\eta \, (\widehat{y}_t - y_t) \, (\mathbb{I}_t * \mathbf{f}_t) \; ;$$

(3) compute $\mathbf{p}_{t+1} = P_{t+1}(\mathbf{w}_{t+1})$.

Performance

η	1e-9	1e-8	1e-7	1e-6	$\eta^\star =$ 2e-8
RMSE	31.7	31.4	34.4	39.1	31.3

Graphical evolution of the weights

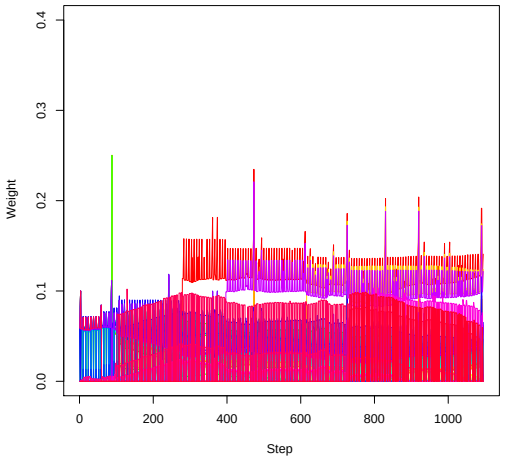

Figure 2.6: Evolution of the weights of the lazy version for $\eta =$ 2e-8.

2.6.3 Second forecaster: by plug-in and projection

Interpretation and/or comments

This second version is less intuitive but it achieves a better practical performance. In view of its statement, the oracle its performance should be compared to is given by

$$\min_{\mathbf{q}\in\mathcal{X}} \sum_{t=1}^{n} \ell_t\left(\mathbf{q}^t\right)$$

where the sequence of the $\mathbf{q}^t$ is defined iteratively by $\mathbf{q}^1 = P_1(\mathbf{q})$ and $\mathbf{q}^{s+1} = P_{s+1}(\mathbf{q}^s)$ for $s \geqslant 1$.

Theoretical bound

One can prove a $O(\sqrt{n})$ bound on the regret

$$\widehat{L}_n - \min_{\mathbf{q}\in\mathcal{X}} \sum_{t=1}^{n} \ell_t\left(\mathbf{q}^t\right) \ .$$

Statement and implementation

The forecaster is implemented as follows.

Parameters: $\eta > 0$

Initialization:

$$p_{i,1} = \begin{cases} \dfrac{1}{|E_1|} & \text{if } i \in E_1; \\ 0 & \text{if } i \notin E_1. \end{cases}$$

For each round $t = 1, 2, \ldots, n$,

(1) predict with $\widehat{y}_t = \mathbf{p}_t \cdot \mathbf{f}_t$;

(2) compute $\mathbf{p}_{t+1}$ as

$$\mathbf{p}_{t+1} = P_{t+1}\Big(\mathbf{p}_t - 2\eta\,(\widehat{y}_t - y_t)(\mathbb{I}_t * \mathbf{f}_t)\Big) \ .$$

Performance

η	1e-7	1e-6	1e-5	1e-4	$\eta^\star = $ 1e-5
RMSE	39.1	30.9	28.2	36.4	28.2

Graphical evolution of the weights

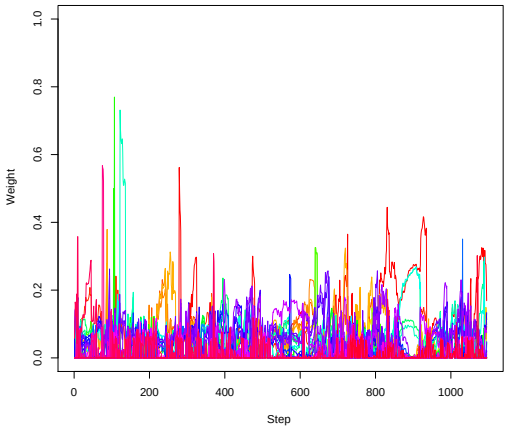

Figure 2.7: Evolution of the weights of the plug-in version for $\eta =$ 1e-5.

2.7 Ridge Regression Forecaster

2.7.1 References

This is an adaptation of [CBL06, Section 11.7].

2.7.2 Theoretical bound

We have

$$R_n(\mathbf{v}) \leqslant \frac{\lambda}{2}\|\mathbf{v}\|^2 + \left(\sum_{i=1}^{n} \log\left(1 + \frac{\mu_i}{\lambda}\right)\right) \max_{1 \leqslant t \leqslant n} \ell_t(\mathbf{v})$$

where $\mu_1, \dots, \mu_N$ are the eigenvalues of $\sum_{t=1}^{n} \mathbf{f}_t \mathbf{f}_t^T$. See [CBL06, Section 11.7] for the proof.

2.7.3 Interpretation and/or comments

The forecaster below uses the standard version of the ridge regression forecaster in the framework of sleeping experts. In order to ensure that this forecaster is well defined at each round, predictions of inactive experts are set to zero as explained in the introduction. Because of that, the predictions of the forecaster are strongly biased.

In fact, this forecaster is designed to come close to the performance given by $B_{\mathbb{R}^N}$. It actually achieves this goal but the latter is not ambitious and quite irrelevant, as the introduction underlines.

2.7.4 Statement and implementation

The parameter λ belongs to $]0, +\infty[$. We take $\mathbf{u}_1$ as the uniform distribution on E_1. For $t \geqslant 2$, the vector of weights $\mathbf{u}_t$ is defined as

$$\mathbf{u}_t = \operatorname*{argmin}_{\mathbf{v} \in \mathbb{R}^N} \left\{ \lambda \|\mathbf{v}\|^2 + \sum_{s=1}^{t-1} (\mathbf{v} \cdot \mathbf{f}_s - y_s)^2 \right\} .$$

The computation of this least-square estimate is given by $\mathbf{u}_t = A_t^{-1} \mathbf{b}_t$, where for $t \geqslant 2$,

$$A_t = \lambda I + \sum_{s=1}^{t-1} \mathbf{f}_s \mathbf{f}_s^T \qquad \text{and} \qquad \mathbf{b}_t = \sum_{s=1}^{t-1} y_s \mathbf{f}_s .$$

Simple manipulations lead to the recursive update

$$\mathbf{u}_{t+1} = \mathbf{u}_t - A_{t+1}^{-1} \left(\mathbf{u}_t^T \mathbf{f}_{t+1} - y_{t+1}\right) \mathbf{f}_t .$$

The forecaster is implemented as follows.

Parameters: penalization factor λ
Initialization: $A_1 = \lambda I$ and $\mathbf{u}_1 = (1/N, \dots, 1/N)$
For each round $t = 1, \dots, n$

(1) predict with $\widehat{y}_t = \mathbf{u}_t \cdot \mathbf{f}_t$;

(2) observe y_t and update A_{t+1}

$$A_{t+1} = A_t + \mathbf{f}_t \mathbf{f}_t^T;$$

(3) compute $\mathbf{u}_{t+1}$ as

$$\mathbf{u}_{t+1} = \mathbf{u}_t - A_{t+1}^{-1} \left(\mathbf{u}_t^T \mathbf{f}_{t+1} - y_{t+1} \right) \mathbf{f}_t \ .$$

2.7.5 Performance

λ	1e-3	1	1e+3	1e+6	$\lambda^\star = 4e+5$
RMSE	52.8	51.5	49.1	42.1	41.8

2.7.6 Graphical evolution of the weights

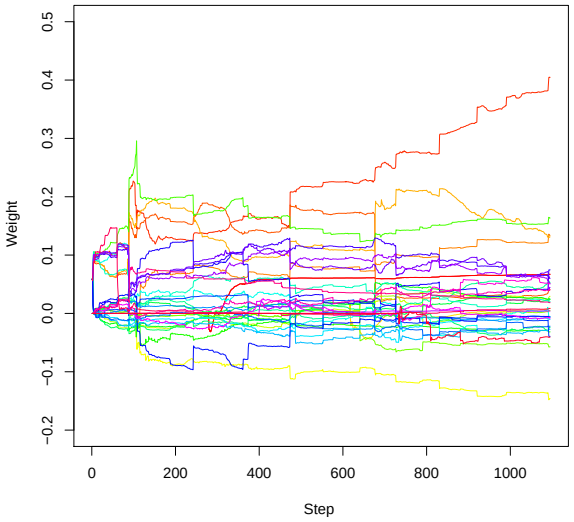

Figure 2.8: Evolution of the weights for $\lambda = 4e{+}5$

2.8 Ridge Regression Partition Forecaster

2.8.1 References

This is an adaptation of [CBL06, Section 11.7].

2.8.2 Theoretical bound

We proceed as in Section 2.4 and derive the global bound as a sum over the K different sub-regimes U_k, for $k = 1, \dots, K$:

$$R_n(\mathbf{v}_1^K) \leqslant \sum_{k=1}^{K} \left(\frac{\lambda}{2} \|\mathbf{v}_k\|^2 + \left(\sum_{i=1}^{n} \log\left(1 + \frac{\mu_i^k}{\lambda}\right) \right) \max_{t \in U_k} \ell_t(\mathbf{v}_k) \right)$$

for all $\mathbf{v}_1^K = (\mathbf{v}_1, \dots, \mathbf{v}_K) \in \left(\mathbb{R}^N\right)^K$, where $\mu_1^k, \dots, \mu_N^k$ are the eigenvalues of $\sum_{t \in U_k} \mathbf{f}_t \mathbf{f}_t^T$ for each $k = 1, \dots, K$.

2.8.3 Interpretation and/or comments

This forecaster is another adaptation of the ridge regression forecaster to the setting of sleeping experts. It simply performs K simultaneous instances of ridge regression to avoid any bias issues (see the issues encountered with the adaptation of Section 2.7).

Nevertheless, this results in poor results on this data set, because K is large and the first steps of instance of the ridge regression forecaster have a large loss. We try to improve these results by cleaning the data set, e.g., by removing data corresponding to rounds t when $|E_t| = 1$ (we remove 40 such t). The results are then slightly better but still very poor.

2.8.4 Statement and implementation

The parameter λ belongs to $]0, +\infty[$.

We introduce the sets

$$\mathbb{R}^N_{E^t} = \left\{ \mathbf{v} \in \mathbb{R}^N : \quad \forall i \notin E_t,\ v_i = 0 \right\}$$

and $U^t = U_{k_t} \cap \{1, \dots, t-1\}$.

For $t \geqslant 1$,

- if $U^t = \emptyset$, the vector of weights $\mathbf{u}_t$ is defined as the uniform distribution on E_t;
- otherwise $\mathbf{u}_t$ is defined as

$$\mathbf{u}_t = \operatorname*{argmin}_{\mathbf{v} \in \mathbb{R}^N_{E_t}} \left\{ \lambda \|\mathbf{v}\|^2 + \sum_{s \in U^t} (\mathbf{v} \cdot \mathbf{f}_s - y_s)^2 \right\} .$$

Note that it is equivalent to consider that we perform K simultaneous ridge regressions (on each U_k, for $k = 1, \dots, K$). We thus have at step t that if $U^t \neq \emptyset$, then $\mathbf{u}_t = A_t^{-1} \mathbf{b}_t$,

$$A_t = \lambda I + \sum_{s \in U^t} \mathbf{f}_s \mathbf{f}_s^T \qquad \text{and} \qquad \mathbf{b}_t = \sum_{s \in U^t} y_s \mathbf{f}_s \ .$$

This forecaster is implemented as follows.

Parameters: penalization factor λ

For each round $t = 1, \dots, n$,

(1) if $U^t = \emptyset$, define $\mathbf{u}_t$ as

$$u_{i,t} = \begin{cases} \dfrac{1}{|E_t|} & \text{if } i \in E_t; \\ 0 & \text{if } i \notin E_t; \end{cases}$$

(2) otherwise, when $U^t \neq \emptyset$, compute A_t as

$$A_t = \lambda I + \sum_{s \in U^t} \mathbf{f}_s \mathbf{f}_s^T$$

and $\mathbf{u}_t$ as

$$(u_{i,t})_{i \in E_t} = A_t^{-1} \sum_{s \in U^t} y_s \mathbf{f}_s$$

and $u_{i,t} = 0$ for $i \notin E_t$;

(3) predict with $\widehat{y}_t = \mathbf{u}_t \cdot \mathbf{f}_t$;

(4) observe y_t.

2.8.5 Performance

The performance is shown for the initial (not cleaned) data set. It is slightly better for cleaned data but remains poor (the best RMSE equaling 137).

λ	1e-3	1	1e+3	1e+6	1e+9	$\lambda^\star = 1.1$e+5
RMSE	192.7	148.39	143.4	142.6	142.6	142.5

2.8.6 Graphical evolution of the weights

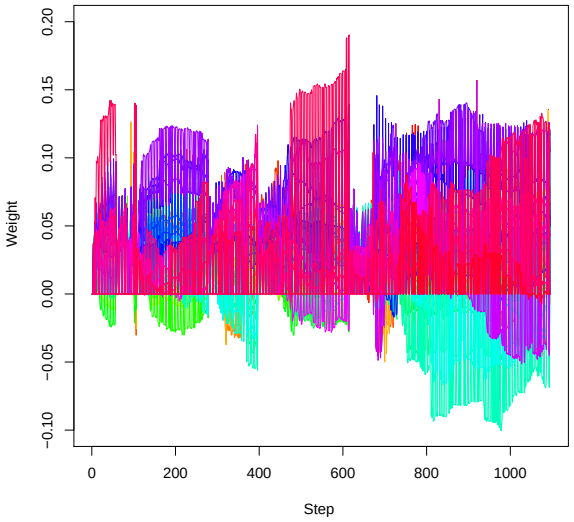

Figure 2.9: Evolution of the weights for λ =1.1e+5

2.9 Normalized Ridge Forecaster

2.9.1 References

This is a new forecaster!

2.9.2 Theoretical bound

We have no theoretical bound yet.

2.9.3 Interpretation and/or comments

The forecaster results from an adaptation of the forecaster of Section 2.7 (the least-squares formulation) in the spirit of the one of Section 2.5 (the renormalization factors in front of the squares).

Its aim is to get close to the oracle 28.8 that is the linear (weights in $\mathbb{R}^N$) counterpart of $B_{\text{renorm}}^{(b)}$.

2.9.4 Statement and implementation

The parameter λ belongs to $]0, +\infty[$. We take $\mathbf{u}_1$ as the uniform distribution over E_1. For $t \geqslant 2$, we compute $\mathbf{u}_t \in \mathbb{R}^N$ as indicated below and then aggregate the forecasts of the experts with ${\mathbf{u}_t}^{E_t}$, where for all $\mathbf{v} \in \mathbb{R}^N$, we define, similarly to the notations used in Section 1.1,

$$\mathbf{v}(E_t) = \sum_{i=1}^{N} \mathbb{I}_{\{i \in E_t\}} \nu_{i,t} \qquad \text{and} \qquad \mathbf{v}^{E_t} = \begin{cases} \dfrac{\nu_{i,t}}{\mathbf{v}(E_t)} & \text{if } i \in E_t, \\ 0 & \text{if } i \notin E_t, \end{cases}$$

and $\mathbf{u}_t$ is defined as

$$\mathbf{u}_t = \operatorname*{argmin}_{\mathbf{v} \in \mathbb{R}^N} \left\{ \lambda \|\mathbf{v}\|^2 + \sum_{s=1}^{t-1} \mathbf{v}(E_s) \left(\mathbf{v}^{E_s} \cdot \mathbf{f}_s - y_s \right)^2 \right\} .$$

Note that we do not provide any efficient recursive computation anymore.

The forecaster is implemented as follows.

Parameters: penalization factor λ

Initialization: $\mathbf{u}_1 = (1/N, \dots, 1/N)$

For each round $t = 1, \dots, n$

(1) predict with $\widehat{y}_t = \mathbf{u}_t{}^{E_t} \cdot \mathbf{f}_t$;

(2) observe y_t and compute $\mathbf{u}_{t+1}$ as

$$\mathbf{u}_{t+1} = \underset{\mathbf{v}\in\mathbb{R}^N}{\operatorname{argmin}} \left\{ \lambda \|\mathbf{v}\|^2 + \sum_{s=1}^{t} \mathbf{v}(E_s)(\mathbf{v}^{E_s} \cdot \mathbf{f}_s - y_s)^2 \right\} .$$

2.9.5 Performance

The performance of this forecaster is better than the one of ridge regression (Section 2.7) but remains poor (worse than the one of the best expert). This might be due to a lack of precision in the implementation: since no closed form expression is available for the $\mathbf{u}_t$, numerical minimizations (implemented in the software `R`) are performed at each step.

λ	1e+4	1e+5	1e+6	1e+7	$\lambda^\star$ = 1e+6
RMSE	31.5	43.12	31.3	34.0	31.3

2.9.6 Graphical evolution of the weights

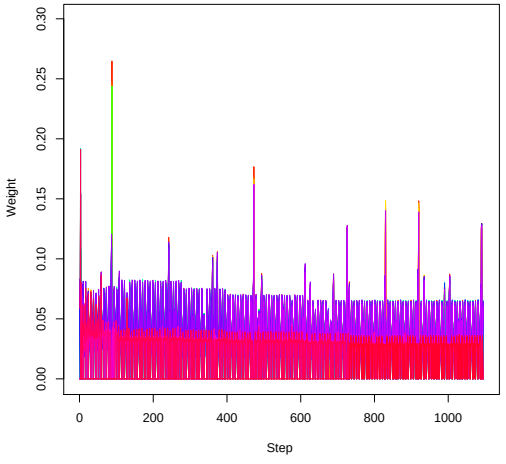

Figure 2.10: Evolution of the weights for λ = 1e+6.

2.10 Fixed-Share Forecaster

2.10.1 References

This is an adaptation of a forecaster described, e.g., in [CBL06, Section 5.2].

2.10.2 Theoretical bound

We take a larger comparison class, whose elements are indexed by sequences $(i_1, \dots, i_n)$ of elements of $\{1, \dots, N\}$ and are called compound experts. They predict at round t as expert i_t. In the context of sleeping experts we thus only consider sequences such that that for all t, one has $i_t \in E_t$; a sequence satisfying this condition will be referred to as an admissible sequence. The tracking regret for a such sequence is defined as

$$R_n((i_1, \dots, i_n)) = \sum_{t=1}^{n} \widehat{\ell}_t - \ell_{i_t,t} \, .$$

The size

$$\mathrm{size}(i_1, \dots, i_n) = \sum_{t=2}^{n} \mathbb{I}_{\{i_{t-1} \neq i_t\}}$$

of a sequence counts how many switches occur in the sequence.

The following theoretical bound holds for the tracking regret of the forecaster presented in this section:

$$R_n((i_1, \dots, i_n)) \leqslant \frac{m+1}{\eta} \log N + \frac{1}{\eta} \log \frac{1}{(\alpha/N)^m (1-\alpha)^{n-m-1}} + \frac{\eta}{8} n B^4$$

for all admissible sequences $(i_1, \dots, i_n)$ of size smaller than m. Thus, taking $\alpha = m/(n-1)$ and

$$\eta = \frac{1}{B^2} \sqrt{\frac{8}{n} \left((m+1) \log N + (n-1) H\left(\frac{m}{n-1}\right) \right)} \, ,$$

we get

$$R_n((i_1, \dots, i_n)) \leqslant B^2 \sqrt{\frac{n}{2} \left((m+1) \log N + (n-1) H\left(\frac{m}{n-1}\right) \right)} \, ,$$

where H denotes the binary entropy function: $H(x) = -x \log x - (1-x) \log(1-x)$ for $x \in [0, 1]$.

2.10.3 Interpretation and/or comments

At each step, this forecaster performs two updates. The first one is similar to the one performed by the exponentially weighted average forecaster of Section 2.2. The second one mixes the previous weights to ensure that all base forecasters get a sufficient weight, which allows them to recover quickly higher weights in case they start outputting good predictions.

Note that for $\alpha = 0$, we do not recover the forecaster of Section 2.2. This is because we deal in a different way with the fact that some experts may be inactive.

2.10.4 Statement and implementation

The parameter η belongs to $]0, +\infty[$, while α belongs to $[0, 1]$.

For $t \geqslant 1$, the convex weights $\mathbf{p}_t$ are defined in two steps: the loss update (the same as in Section 2.2) and the share update, which allows our forecaster to be more reactive to breaking points (when the index of the best expert changes).

The forecaster is implemented as follows.

Parameters: $\eta > 0$ and $0 \leqslant \alpha \leqslant 1$

Initialization: $\mathbf{w}_0 = (1/N, \ldots, 1/N)$

For each round $t = 1, 2, \ldots, n$,

(1) predict with the weights

$$\mathbf{p}_t = \frac{\mathbf{w}_{t-1}}{\sum_{j=1}^{N} w_{j,t-1}} \; ;$$

(2) Loss update: observe y_t and update for each $i = 1, \ldots, N$,

$$v_{i,t} = \begin{cases} w_{i,t-1}\, e^{\eta\left(\widehat{\ell}_t - \ell_{i,t}\right)} & \text{if } i \in E_t, \\ \text{undefined} & \text{if } i \notin E_t; \end{cases}$$

(3) Share update: let $w_{i,t} = 0$ if $i \notin E_{t+1}$, and

$$w_{i,t} = \frac{1}{|E_{t+1}|} \sum_{j \in E_t \setminus E_{t+1}} v_{j,t} + \frac{\alpha}{|E_{t+1}|} \sum_{j \in E_t \cap E_{t+1}} v_{j,t} + (1-\alpha)\, \mathbb{I}_{\{i \in E_t \cap E_{t+1}\}} v_{i,t}$$

if $i \in E_{t+1}$ (with the convention that an empty sum is null).

Note: in (2), it is equivalent to put $\widehat{\ell}_t$ or to omit it.

2.10.5 Proof of the theoretical bound

The proof is given by a straightforward adaptation of the proof in the non-sleeping case proposed in [CBL06, Section 5.2]. Indeed, the version proposed here corresponds to a (fake) prior weight assignment to the set of admissible compound experts given by a certain Markovian process. We give here the relation that defines the transition probability:

$$w'_0(i_1, \ldots, i_{t+1}) = w'_0(i_1, \ldots, i_t)\; \mathbb{I}_{\{i_{t+1} \in E_{t+1}\}} \left[(1-\alpha) \mathbb{I}_{\{i_{t+1} = i_t\}} + \frac{\alpha}{|E_{t+1}|} + \mathbb{I}_{\{i_t \notin E_{t+1}\}} \frac{1-\alpha}{|E_{t+1}|} \right] .$$

That is, the transition function from i_t to i_{t+1} is given, at round $t+1$, by

$$\mathrm{Tr}_{t+1}(i_t \to i_{t+1}) = \mathbb{I}_{\{i_{t+1} \in E_{t+1}\}} \left[(1-\alpha) \mathbb{I}_{\{i_{t+1} = i_t\}} + \frac{\alpha}{|E_{t+1}|} + \mathbb{I}_{\{i_t \notin E_{t+1}\}} \frac{1-\alpha}{|E_{t+1}|} \right] .$$

Rewriting the proof of [CBL06, Theorem 5.1] with these prior weights we obtain how compute efficiently $w_{i,t}$ thanks to the vector $\mathbf{v}_t$. In particular, the last equality of the

proof yields

$$w_{i,t} = \sum_{j \in E_t} v_{j,t} \operatorname{Tr}_{t+1}(j \to i) = \sum_{j \in E_t} v_{j,t} \mathbb{I}_{\{i \in E_{t+1}\}} \left[(1-\alpha)\mathbb{I}_{\{i=j\}} + \frac{\alpha}{|E_{t+1}|} + \mathbb{I}_{\{j \notin E_{t+1}\}} \frac{1-\alpha}{|E_{t+1}|} \right] ,$$

from which follows the expression stated in our implementation.

As the prior weights w'_0 are larger than the ones w_0 proposed by the original version of [CBL06, Section 5.2] that puts weights on all compound experts, this entails the proposed bound.

2.10.6 Performance

This forecaster has a good performance (better than the one of the exponentiated gradient forecaster of Section 2.3). This might be because this forecaster uses two parameters; however, in practice, tuning two parameters online is often more delicate than simply calibrating one. We will propose several solutions to this issue in Section 3.4. The best performance (RMSE of 27.0) is achieved for $(\eta^\star, \alpha^\star) = (2\text{e-}3, 0.2)$.

	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.1$	$\alpha = 0.2$
$\eta =$ 1e-6	35.9	33.4	32.6	32.0
$\eta =$ 1e-5	33.1	32.1	31.7	31.5
$\eta =$ 1e-4	31.2	29.3	29.1	29.5
$\eta =$ 1e-3	29.3	28.1	27.5	27.2
$\eta =$ 1e-2	28.4	28.0	27.9	27.8

2.10.7 Graphical evolution of the weights

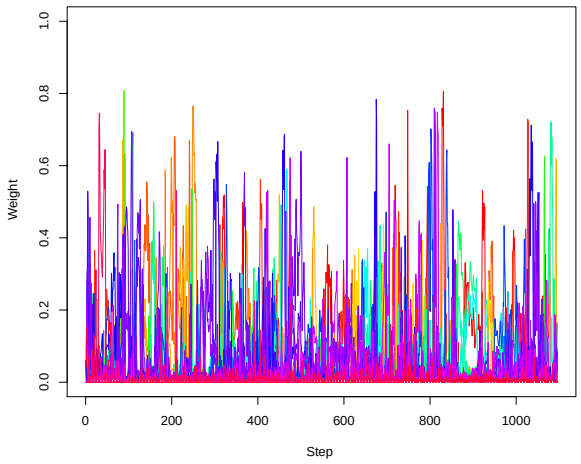

Figure 2.11: Evolution of the weights for $(\eta^\star, \alpha^\star) = (2\text{e-}3, 0.2)$.

2.11 Exponentiated-Gradient Fixed-Share Forecaster

2.11.1 References

It follows from the one of Section 2.10 thanks to the consideration of the gradients of the losses, see [CBL06, Section 2.5].

2.11.2 Theoretical bound

Recall that $\nabla\widehat{\ell}_t = 2(\widehat{y}_t - y_t)\mathbf{f}_t$.
We denote by C a constant such that $-C \leqslant \left(\nabla\widehat{\ell}_t\right)_i \leqslant C$ for all $i = 1, \dots, N$ and $t = 1, \dots, n$. We can take $C = 2B^2$, so that the range is $4B^2$.

The bound of Section 2.10 holds in particular for the forecaster described here, up to the replacement of B^2 by the new range $4\,B^2$.

2.11.3 Interpretation and/or comments

The forecaster below is the exponentiated gradient version of the forecaster of Section 2.10, in which the cumulative regret of interest is first upper-bounded by some gradient-based sum.

2.11.4 Statement and implementation

The parameter η belongs to $]0, +\infty[$, while α belongs to $[0, 1]$. The forecaster is implemented as follows.

Parameters: $\eta > 0$ and $0 \leqslant \alpha \leqslant 1$
Initialization: $\mathbf{w}_0 = (1/N, \dots, 1/N)$
For each round $t = 1, 2, \dots, n$,

(1) predict with the weights

$$\mathbf{p}_t = \frac{\mathbf{w}_{t-1}}{\sum_{j=1}^{N} w_{j,t-1}} \; ;$$

(2) Loss update: observe y_t and update

$$v_{i,t} = \begin{cases} w_{i,t-1}\, e^{-2\eta\left(\widehat{y}_t - y_t\right)\left(\widehat{y}_t - f_{i,t}\right)} & \text{if } i \in E_t, \\ \text{undefined} & \text{if } i \notin E_t; \end{cases}$$

(3) Share update: let $w_{i,t} = 0$ if $i \notin E_{t+1}$, and

$$w_{i,t} = \frac{1}{|E_{t+1}|} \sum_{j \in E_t \setminus E_{t+1}} v_{j,t} + \frac{\alpha}{|E_{t+1}|} \sum_{j \in E_t \cap E_{t+1}} v_{j,t} + (1-\alpha)\, \mathbb{I}_{\{i \in E_t \cap E_{t+1}\}} v_{i,t}$$

if $i \in E_{t+1}$ (with the convention that an empty sum is null).

2.11.5 Performance

This forecaster has a relatively good performance (better than the one of the exponentiated gradient forecaster of Section 2.3), but surprisingly enough it does not beat the exponentially weighted average version of the fixed-share forecaster presented in Section 2.10.

	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.1$	$\alpha = 0.2$
$\eta =$ 1e-5	33.0	32.0	31.7	31.5
$\eta =$ 1e-4	28.5	28.0	28.3	28.9
$\eta =$ 1e-3	29.9	29.5	29.3	29.2
$\eta =$ 1e-2	28.9	28.7	28.4	28.5
$\eta =$ 1e-1	29.0	28.7	28.6	28.5

The best performance is achieved for the pair $(\eta^\star, \alpha^\star) =$ (2e-4, 0.07) and the value of the corresponding RMSE is 27.2.

2.11.6 Graphical evolution of the weights

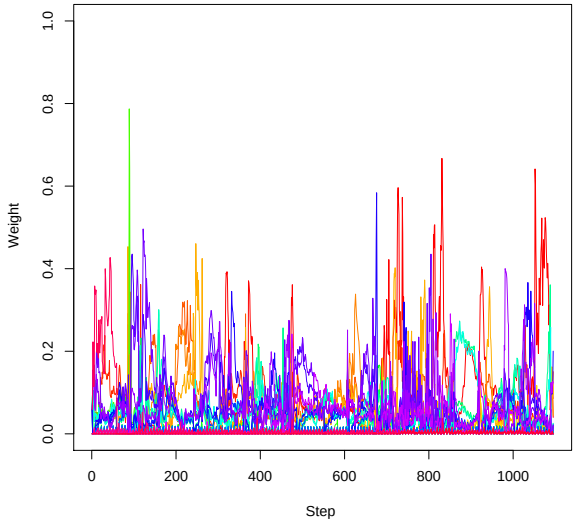

Figure 2.12: Evolution of the weights for $(\eta^\star, \alpha^\star) = (\text{2e-4}, 0.07)$.

2.12 Another sleeping adaptation of the fixed-share forecaster

2.12.1 References

This is another adaptation of the original fixed-share forecaster described, e.g., in [CBL06, Section 5.2]. As the forecaster of Section 2.10, this forecaster has a counterpart in terms of exponentiated gradient.

2.12.2 Interpretation and/or comments

We do not provide any theoretical result for these forecasters yet. They achieve a good performance especially in the exponentiated gradient case.

Moreover, if we take $\alpha = 0$ we recover exactly the forecaster of Section 2.2 (respectively, Section 2.3 in the exponentiated gradient case), which is not the case for the forecasters of Sections 2.10 and 2.11. However, the weights obtained in this case are not exactly the same because of computational issues: the loss update for the fixed-share forecasters is made multiplicatively whereas it is made additively in the case of the forecasters of Sections 2.10 and 2.11.

2.12.3 Statement and implementation

We present here only the implementation of the variant of the exponentially weighted average fixed-share forecaster. The exponentiated gradient version can be obtained by replacing the losses by the gradients of the losses.

The parameter η belongs to $]0, +\infty[$, while α belongs to $[0, 1]$ (when $\alpha = 0$, we recover exactly the forecaster of Section 2.2). The forecaster is implemented as follows.

Parameters: $\eta > 0$ and $0 \leqslant \alpha \leqslant 1$

Initialization: $\mathbf{w}_0 = (1/N, \ldots, 1/N)$

For each round $t = 1, 2, \ldots, n$,

(1) predict with the weights

$$p_{i,t} = \frac{\mathbb{I}_{\{i \in E_t\}} w_{i,t-1}}{\sum_{j=1}^{N} \mathbb{I}_{\{i \in E_t\}} w_{j,t-1}} \qquad \text{for each } i = 1, \ldots, N;$$

(2) Loss update: observe y_t and update for each $i = 1, \ldots, N$,

$$v_{i,t} = \begin{cases} w_{i,t-1}\, e^{-\eta\left(\widehat{\ell}_t - \ell_{i,t}\right)} & \text{if } i \in E_t, \\ v_{i,t-1} & \text{if } i \notin E_t; \end{cases}$$

(3) Share update: let

$$w_{i,t} = \begin{cases} \alpha \dfrac{W_t}{|E_{t+1}|} + (1-\alpha) v_{i,t} & \text{if } i \in E_{t+1}, \\ 0 & \text{if } i \notin E_{t+1}, \end{cases}$$

where $W_t = \sum_{i=1}^{N} \mathbb{I}_{\{i \in E_{t+1}\}} v_{i,t}$.

2.12.4 Performance

The two forecasters have a good performance. The one of the exponentially weighted average version is similar to the one of the forecaster of Section 2.10. The best performance is achieved for $(\eta^\star, \alpha^\star) = (1\text{e-}3, 0.2)$.

	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.1$	$\alpha = 0.2$
$\eta =$ 1e-5	30.6	30.6	30.7	30.9
$\eta =$ 1e-4	30.7	29.8	29.6	29.7
$\eta =$ 1e-3	28.7	27.7	27.3	27.0
$\eta =$ 1e-2	27.9	27.6	27.5	27.4

The exponentiated gradient version has a performance better than the one of the forecaster of Section 2.11. The best performance (RMSE of 26.5) is achieved for $(\eta^\star, \alpha^\star) =$ (2e-4, 0.05).

	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.1$	$\alpha = 0.2$
$\eta =$ 1e-5	29.9	30.4	30.6	30.9
$\eta =$ 1e-4	27.0	27.1	27.6	28.5
$\eta =$ 1e-3	29.2	29.0	29.0	28.8
$\eta =$ 1e-2	28.3	28.4	28.2	28.3
$\eta =$ 1e-1	28.9	28.6	28.6	28.2

2.12.5 Graphical evolution of the weights

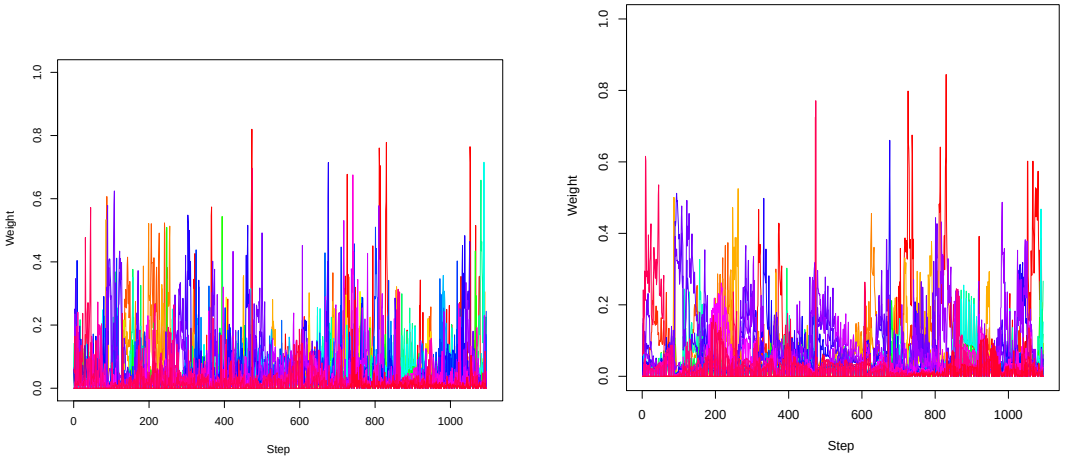

Figure 2.13: Evolution of the weights of the two variants of the fixed-share forecasters considered here: the exponentially weighted average version (left) and the exponentiated gradient version (right).

Chapter 3

Tricks

3.1 On-line calibration of one parameter

3.1.1 References

This adaptive calibration method has been introduced in [GMS08].

3.1.2 Theoretical Bound

No theoretical bound yet. The aim is to achieve a performance nearly as good as the one of the considered prediction method tuned with the best parameter in hindsight.

3.1.3 Interpretation and/or comments

This method provides a generic adaptive implementation of all forecasters that depend only on a (possibly vector-valued) tuning parameter. It automatically calibrates this parameter by choosing the parameter value that minimizes the past cumulative loss of the considered forecaster. Several optimization methods can be used. We choose a grid-based optimization procedure, but continuous methods, though necessarily inaccurate and more difficult to control, could be more effective.

3.1.4 Statement and implementation

Consider a forecaster depending on a parameter λ and whose weight vector at step t is denoted by $\mathbf{v}_t = \mathbf{v}_t^{(\lambda)}$. The parameter $\lambda \in \Lambda$ is called a tuning parameter of the forecaster and our aim is to choose it in an automatic way. The parameter space is, for instance, $\Lambda =]0, +\infty[$ for ridge regression, exponential weighted average, and exponential gradient forecasters.

In addition we assume that $\mathbf{v}_1^{(\lambda)} = \mathbf{v}_1^\star$ does not depend on λ, which is the case for all methods studied so far.

The proposed calibration method tunes the forecaster automatically by minimizing of the empirical loss. The latter is thus called ELM calibrated forecaster thereafter. It chooses $\mathbf{v}_1 = \mathbf{v}_1^\star$ and, for $t \geqslant 2$,

$$\mathbf{v}_t = \mathbf{v}_t^{\left(\widehat{\lambda}_t\right)} \qquad \text{where} \qquad \widehat{\lambda}_t \in \operatorname*{argmin}_{\lambda \in \widetilde{\Lambda}} \sum_{s=1}^{t-1} \left(\mathbf{v}_s^{(\lambda)} \cdot \mathbf{f}_s - y_s\right)^2 .$$

We chose a finite logarithmically-scaled parameter grid $\widetilde{\Lambda} \subset \Lambda$ to perform an approximation of the minimization above.

This is implemented as follows.

Parameters: a grid $\widetilde{\Lambda}$
Initialization: $\mathbf{v}_1 = \mathbf{v}_1^\star$
For each round $t = 1, 2, \ldots, n$,

(1) predict with $\widehat{y}_t = \mathbf{v}_t \cdot \mathbf{f}_t$;

(2) observe y_t and compute $\widehat{\lambda}_{t+1}$ as

$$\widehat{\lambda}_{t+1} \in \operatorname*{argmin}_{\lambda \in \widetilde{\Lambda}} \sum_{s=1}^{t} \left(\mathbf{v}_s^{(\lambda)} \cdot \mathbf{f}_s - y_s \right)^2 ;$$

(3) compute $\mathbf{v}_{t+1}$ as

$$\mathbf{v}_{t+1} = \mathbf{v}_{t+1}^{\left(\widehat{\lambda}_{t+1}\right)} .$$

3.1.5 Performance

We have tested the calibration via empirical loss minimization (ELM) on three simple prediction methods: the exponential gradient (EG, see Section 2.3) and exponentially weighted average forecasters (EWA, see Section 2.2), as well as the plug-in version of Zinkevich's forecaster (see Section 2.6).

We chose in all three cases a uniform logarithmic grid over

- [1e-6, 1e-2] for the exponentiated gradient;
- [1e-8, 1e-4] for the exponentially weighted average forecaster;
- [1e-7, 1e-3] for the plug-in version of Zinkevich's forecaster.

Below are summarized the RMSE of the resulting ELM calibrated forecasters, for different numbers of grid points

$$\left|\widetilde{\Lambda}\right| = 5,\ 9,\ 21,\ \text{and}\ 41$$

(for the plug-in version of Zinkevich's forecaster, only the case $= 5$ was computed).

Columns "best" and "worst" refer to the best (respectively, worst) performance obtained with a constant value η taken on the grid.

$\left\|\widetilde{\Lambda}\right\|$	5	9	21	41	best	worst
EWA	31.0	30.7	30.6	30.6	30.5	32.7
EG	28.2	28.2	28.2	28.2	28.2	33.5
Zink.	28.3				28.2	39.1

The performance of the exponentiated gradient forecaster is independent of the size of the grid. This is because for almost all t and all grids, $\widehat{\lambda}_t$ = 1e-4, which is close to the optimal off-line tuned parameter (here $\eta^\star$ = 1.1e-4, see Section 2.3). For the exponentially weighted average forecaster, there is a dependency on the grid: we can obtain better results if our grid ranges from 1e-8 to 1e-5 (of course, it is difficult to guess the best range in advance). Figure 3.1 shows the evolution of the calibrated parameters for these two base forecasters.

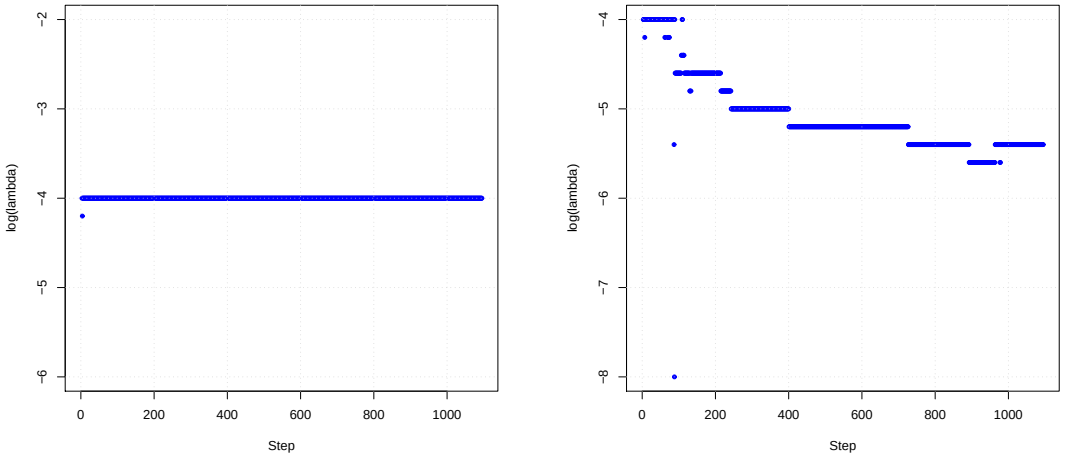

Figure 3.1: Evolution of the calibration parameters $\widehat{\lambda}_t$ for the exponentiated gradient (left) and the exponentially weighted average (right) forecasters. In both cases, the size of the grid is $\left|\widetilde{\Lambda}\right| = 21$.

3.1.6 Graphical evolution of the weights

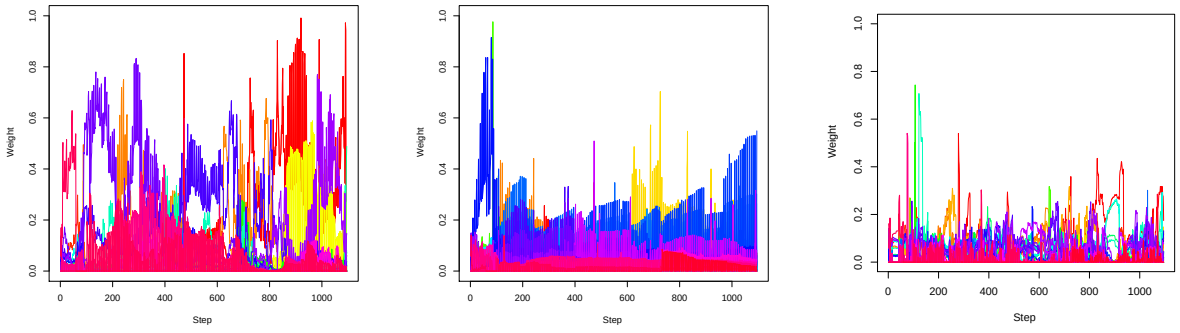

Figure 3.2: Evolution of the weights for the exponentiated gradient forecaster (left), the exponentially weighted average forecaster (middle) and Zinkevich's forecaster (right). For the first two forecasters, the size of the grid was $\left|\widetilde{\Lambda}\right| = 21$, while for the third forecaster, it was equal to 5.

3.2 Towards a reduction of the bias of the ridge-type forecasters?

3.2.1 Intuition

We have met serious bias issues in Sections 2.7 and 2.8. To overcome them, one may think of going back to an unbiased data set by applying the following transformation: we define new fake base forecasters $\widetilde{f}_{i,t} = f_{i,t} - y_t$ and fake outcomes $\widetilde{y}_t = 0$, for all $t = 1, \dots, n$ and $i = 1, \dots, N$. These modified base forecasters can be thought of as measuring the error of prediction of the original ones.

3.2.2 Adaptation of the algorithms

To obtain the linear combination to be used at round t to aggregate the $f_{i,t}$, we run the previous ridge regression forecasters (see Sections 2.7 and 2.8) on the $\widetilde{f}_{i,s}$ and $\widetilde{y}_s$, for $s \leqslant t-1$.

In addition we perform the following transformations on the obtained weights: we set to 0 the weights that correspond to inactive experts and renormalize the remaining weights so that they sum up to 1. Therefore, we obtain at round t an element of $\mathcal{X}_{E_t}$.

This additional step is needed because otherwise all weights are very close to 0 and the obtained prediction is almost meaningless.

3.2.3 Performance

The performance of this adaption for the case of the simple ridge regression of Section 2.7 is given below. It is better than the original forecaster of Section 2.7 mainly because of the renormalization step, but the performance remains disappointing and the fact that the best performance is achieved when λ grows to ∞ and the obtained performance indicate that the chosen convex combination is the uniform one.

λ	1e+3	1e+6	1e+9	1e+12	$\lambda^\star = \infty$
RMSE	35.5	35.3	32.2	31.8	31.8

3.2.4 Graphical evolution of the weights

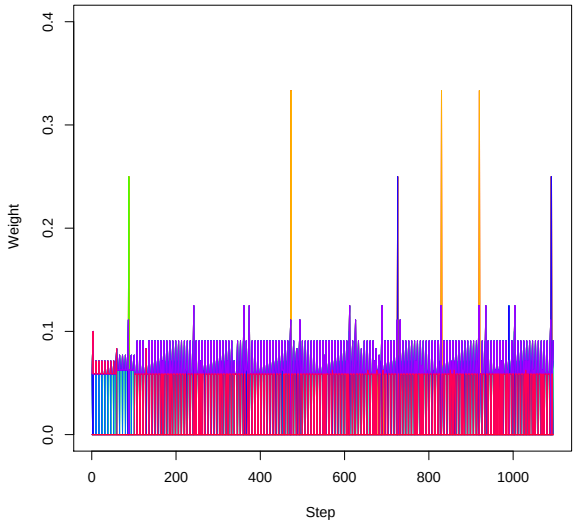

Figure 3.3: Evolution of the weights of the unbiased variant of ridge regression forecaster with $\lambda = 1e{+}20$.

3.3 Compensated Regrets

3.3.1 References

This is an adaptation of the forecaster of Section 2.2, which we designed.

3.3.2 Theoretical bound

We do not provide any theoretical bound yet.

3.3.3 Interpretation and/or comments

This forecaster results from an adaptation of our variant of the exponentially weighted average forecaster described in Section 2.2. Indeed, in the algorithm of Section 2.2, the weights at time t depend are based on a reweighting of the $R'_{i,t-1}$, which can potentially penalize an expert which is often active and incurs a small regret at each step against an expert which is rarely active but incurs at each active step a large regret.

Therefore in this version, instead of taking $R'_{i,t-1}$ in exponent in the weights, we use instead the renormalized version

$$R'_{i,t-1} \Big/ \sqrt{\sum_{t'=1}^{t} \mathbb{I}_{\{i \in E_t\}}}\,.$$

We have adapted the forecaster of Section 2.3 as well.

3.3.4 Statement and implementation

The parameter η belongs to $]0, +\infty[$. For $t \geqslant 1$, $\mathbf{p}_t$ is defined as

$$p_{i,t} = \frac{\mathbb{I}_{\{i \in E_t\}} \exp\left(\eta R'_{i,t-1} \Big/ \sqrt{\sum_{t'=1}^{t} \mathbb{I}_{\{i \in E_t\}}}\right)}{\sum_{j \in E_t} \exp\left(\eta R'_{j,t-1} \Big/ \sqrt{\sum_{t'=1}^{t} \mathbb{I}_{\{j \in E_t\}}}\right)}.$$

By convention $R'_{i,0} = 0$ for all $i = 1, \ldots, N$.

This forecaster is implemented as follows.

Parameters: learning rate $\eta \geqslant 0$

Initialization:

$$p_{i,1} = \begin{cases} \dfrac{1}{|E_1|} & \text{if } i \in E_1; \\ 0 & \text{if } i \notin E_1. \end{cases}$$

For each round $t = 1, 2, \ldots, n$,

(1) predict with $\widehat{y}_t = \mathbf{p}_t \cdot \mathbf{f}_t$;

(2) observe y_t, compute the regrets

$$R'_{i,t} = R'_{i,t-1} + \mathbb{I}_{\{i \in E_t\}} \left(\widehat{\ell}_t - \ell_{i,t} \right) \ ,$$

and update

$$\tau_{i,t} = \tau_{i,t-1} + \mathbb{I}_{\{i \in E_t\}}$$

for all $i = 1, \ldots, N$;

(3) compute $\mathbf{p}_{t+1}$ as

$$p_{i,t+1} = \frac{\mathbb{I}_{\{i \in E_{t+1}\}} \, e^{\eta R'_{i,t} / \sqrt{\tau_{i,t}}}}{\sum_{j \in E_{t+1}} e^{\eta R'_{j,t} / \sqrt{\tau_{j,t}}}}$$

for all $i = 1, \ldots, N$.

3.3.5 Performance

The performance of the compensated exponentially weighted average forecaster is slightly better than the one of the base sleeping variant of the exponentially weighted average forecaster.

η	1e-6	1e-5	1e-4	1e-3	$\eta^\star = 8.5\text{e-}5$
RMSE	31.3	31.0	30.5	31.9	30.4

This is not the case for the exponentiated gradient forecaster: the compensated variant is less efficient that the original version.

η	1e-6	1e-5	1e-4	1e-3	$\eta^\star = 8.5\text{e-}5$
RMSE	31.3	31.1	30.0	29.0	28.6

3.3.6 Graphical evolution of the weights

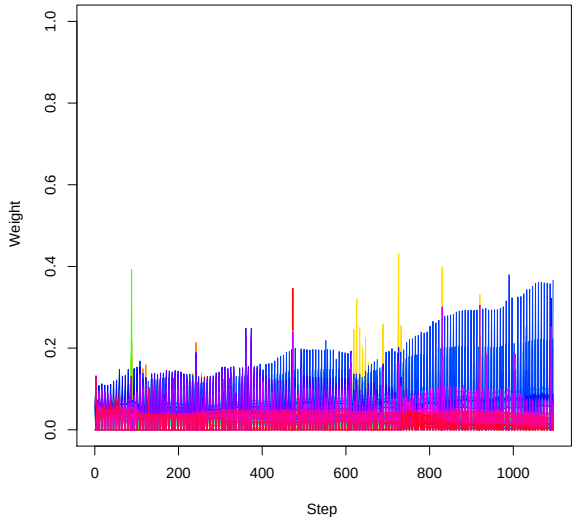

Figure 3.4: Evolution of the weights for the compensated exponentially weighted average forecaster with the choice $\eta = 8.5\text{e-}5$.

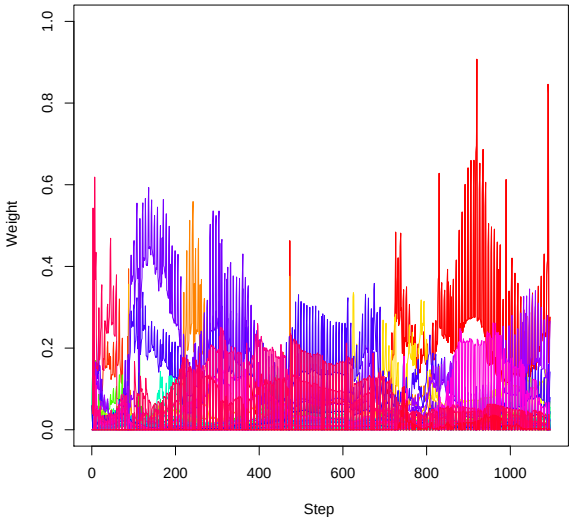

Figure 3.5: Evolution of the weights for the compensated exponentiated gradient forecaster with the choice $\eta = 6.8\text{e-}4$.

3.3.7 Note

The variant with the renormalized quantity

$$R'_{i,t-1} \Big/ \sum_{t'=1}^{t} \mathbb{I}_{\{i \in E_t\}}$$

obtained a worse performance (which we did not report in details here).

3.4 Calibration of the couple of parameters for fixed-share type forecasters

3.4.1 Overview

We propose here three methods to tune the couple of parameters on which the fixed-share type forecasters of Sections 2.10 (referred to as FS-EWA hereafter) and Section 2.11 (referred to as FS-EG hereafter) rely on:

- the use of a grid,
- the estimation of a break-points rate,
- online tuning of η and α as functions of the current time index t to minimize the theoretical bound.

One can also apply these methods to the forecasters of Section 2.12. The obtained performance is generally similar for the exponentially weighted average versions but is much better for the exponentiated gradient version. However, for simplicity, we only report below the performance of the forecasters derived from Sections 2.10 and 2.11.

3.4.2 Theoretical Bound / Comments

We do not provide any theoretical bound yet. The aim is to achieve a performance nearly as good as the one of the considered prediction method tuned with the best couple of parameters in hindsight.

3.4.3 On-line calibration on a grid

The first solution is to proceed as in Section 3.1 and define a grid for the couple of parameters (η, α). Since there are two parameters, the computation time is much longer than in the case of one parameter only studied therein. On the other hand we have seen in Section 3.1 that choosing a very thin grid, which is computationally much more demanding, does not result in a substantial improvement of the performance.

We thus provide here the performance obtained with the limited grid

$$\widetilde{\Lambda} = \{\text{1e-6, 1e-5, 1e-4, 1e-3, 1e-2}\} \times \{0.01, 0.05, 0.1, 0.2\}$$

for the forecaster derived from FS-EWA and

$$\widetilde{\Lambda} = \{\text{1e-5, 1e-4, 1e-3, 1e-2, 1e-1}\} \times \{0.01, 0.05, 0.1, 0.2\}$$

for the forecaster derived from FS-EG. Columns "best" and "worst" refer to the best (respectively, worst) performance obtained with a constant value (η, α) taken on the grid $\widetilde{\Lambda}$.

	$\widetilde{\Lambda}$	best	worst
FS-EWA	27.4	27.2	35.9
FS-EG	28.3	28.0	33.0

In conclusion, the performance of FS-EWA calibrated on-line on a grid is good. In addition, the computation time was reasonable.

Graphical evolution of the weights

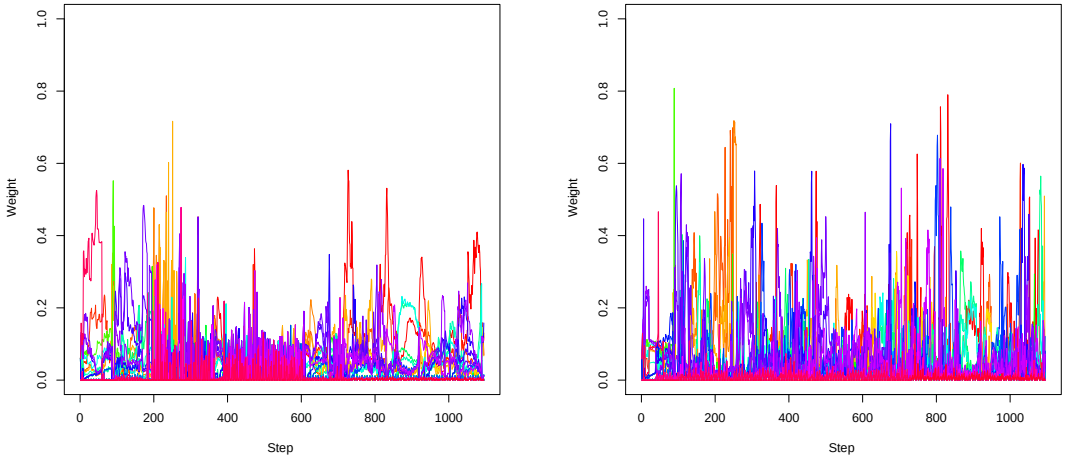

Figure 3.6: Evolution of the weights for FS-EG (left) and FS-EWA (right) calibrated on-line on a grid.

3.4.4 On-line calibration of α in terms of a break-points rate

This second method uses a time-varying α and a fixed η (to be calibrated on a grid, as before). It is easy to see that the implementations of the forecasters FS-EWA and FS-EG can be extended immediately to the case of such time-varying parameters α, we therefore do not detail this point.

As indicated in Section 2.10.2, a good choice of α to optimize the theoretical bound at round n is $\alpha = m/(n-1)$, where m is the number of switches used to define the comparison class formed by the compound experts. The method presented here therefore uses at rounds $t > 2$ a time-varying parameter of the form $\alpha(t) = m(t)/(t-2)$, computed with the help of the losses suffered from rounds 1 to $t-1$. (For $t \leqslant 2$, we take $\alpha(t) = 0$.)

We describe below how to define $m(t)$ at each step. This is done in two layers: first, we define a family of quantities indexed by integers m and based on the best cumulative square loss computed obtained by a compound expert with m switches. Second, we penalize each member of this family by a measure of the complexity of the considered compound experts: the larger the number of switches m, the larger the penalty $\mathrm{pen}(m)$. (Such a penalization is needed, otherwise we would select $m(t) = t-2$ for each t.)

Statement and implementation of the tuning of $m(t)$

We give here the description of the algorithm that selects the value of $m(t)$. For each $t \geqslant 2$, it proceeds in two steps. The first one is to compute the best cumulative square loss corresponding to each $0 \leqslant m < t-1$:

$$L(m, t-1) = \min_{\substack{(i_1,\dots,i_{t-1}) \in E_1 \times \dots \times E_{t-1} \\ \mathrm{size}(i_1,\dots,i_{t-1}) = m}} \sum_{s=1}^{t-1} \left(f_{i_s,s} - y_s\right)^2 .$$

Of course, for too small values of m, the quantity $L(m, t-1)$ may be undefined (the minimum is over an empty set), in which case we consider that it is equal to $+\infty$. The same occurs for $m \geqslant t-1$. The second step is to choose the value m that minimizes the penalized criterion, that is,

$$m(t) = \operatorname*{argmin}_{m \in \{0,\dots,t-2\}} L(m, t-1) + \mathrm{pen}(m)$$

for some penalty function pen. Note that this penalty function should depend on the current round index t but not on n.

We now explain how to perform the first step in an efficient and recursive manner, by backward induction. To do so, we show that it suffices to maintain the weights

$$L(m, i, t) = \min_{\substack{(i_1,\dots,i_{t-1}) \in E_1 \times \dots \times E_{t-1} \\ \mathrm{size}(i_1,\dots,i_{t-1},i) = m}} \left(f_{i,t} - y_t\right)^2 + \sum_{s=1}^{t-1} \left(f_{i_s,s} - y_s\right)^2$$

for all m, t, and $i \in E_t$.

A recursive definition can then be obtained as

$$L(m,i,t) = \min\Big\{L(m,i,t-1),\ \min_{j\in E_{t-1}\setminus\{i\}} L(m-1,j,t-1)\Big\} + \big(f_{i,t}-y_t\big)^2 .$$

It can be exploited as follows to compute the $m(t)$.

Initialization: $m(1)=0$ and $L(0,i,0)=0$ for all $i\in\{1,\dots,N\}$
For each round $t=2,\dots,n$,

(1) update the cumulative losses

$$L(m,i,t-1) = \min\Big\{L(m,i,t-2),\ \min_{j\in E_t\setminus\{i\}} L(m-1,j,t-2)\Big\} + \big(f_{i,t-1}-y_{t-1}\big)^2$$

for all $m\in\{0,\dots,t-2\}$ and all $i\in E_{t-1}$;

(2) compute

$$L(m,t-1) = \min_{i\in E_{t-1}} L(m,i,t-1)$$

for all $m\in\{0,\dots,t-2\}$;

(3) select

$$m(t) = \operatorname*{argmin}_{m\in\{0,\dots,t-2\}} L(m,t-1) + \mathrm{pen}(m) .$$

Performance

We start by indicating some benchmark values (some oracles). We provide below the best performance obtained by a compound expert of size at most m, for different values of m. Note that for $m=n-1$, we recover the oracle B_{oracle} defined in Section 1.1.

m	10	50	200	$n-1=1094$
RMSE	32.1	23.1	15.2	9.4

We tried several penalties for each forecaster (FS-EWA and FS-EG). The form of the bound obtained in Section 2.10.2 seems to argue in favor of a penalty equal to $\sqrt{nm}$: the cumulative loss of the forecaster is less than the cumulative loss of the best compound expert of size m plus something of the order of $\sqrt{nm}$. However, the performance is quite better for a penalty equal to nm for both forecasters. (Other choices of the penalty did not provide better results.) But since we do not know the value of n in advance we use penalties depending on the current round index t, that is, at round t we choose $\mathrm{pen}(m)=tm$. See Table 3.1 for a summary of the performance.

η	1e-4	1e-3	1e-2	$\eta^\star$
FS-EWA	29.8	27.6	28.1	27.3
FS-EG	29.3	28.5	28.6	28.5

Table 3.1: Performance of FS-EWA and FS-EG with a time-varying parameter $\alpha(t)$; the chosen penalty is $\mathrm{pen}_t(m) = tm$.

Of course, we can now apply the on-line calibration trick of Section 3.1 to deal with the tuning of η. The performance is provided below for the same grid as in Section 3.4.3.

	$\widetilde{\Lambda}$	best
FS-EWA	27.7	27.6
FS-EG	28.7	28.5

Graphical evolution of the weights

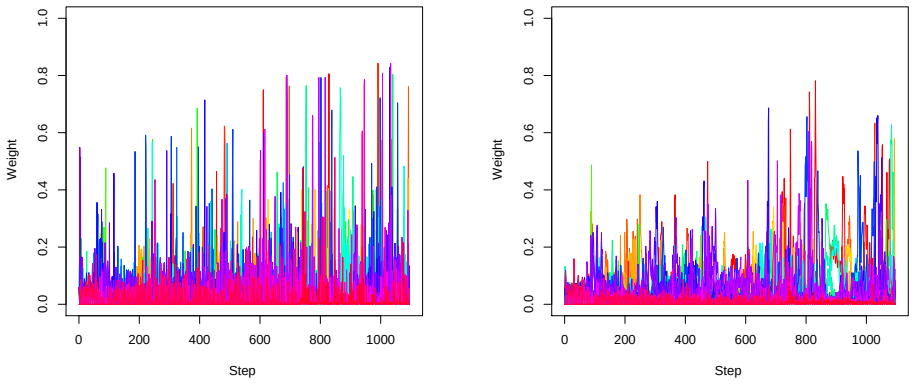

Figure 3.7: Evolution of the weights for FS-EWA, with the penaly $\mathrm{pen}_t(m) = tm$ and for $\eta^\star$ = 3.5e-2 (left) or with the calibration trick of Section 3.1 on a grid (right).

3.4.5 On-line calibration of η as a function of $\alpha(t)$

To take the idea of Section 3.4.4 even further, in addition of taking a time-varying $\alpha(t)$, we could do the same with η, in view of the updates proposed in Section 2.10 and 2.11. This $\eta(t)$ could be chosen so as to minimize the associated theoretical bound. That is, we could take

$$\eta(t) = \frac{1}{B^2}\sqrt{\frac{8}{t}\left((m(t)+1)\log N + (t-1)\,H\left(\frac{m(t)}{t-1}\right)\right)}\,.$$

In fact this choice of $\eta(t)$ give poor results whereas the choice

$$\eta(t) = \frac{1}{B}\sqrt{\frac{8}{t}\left((m(t)+1)\log N + (t-1)\,H\left(\frac{m(t)}{t-1}\right)\right)}$$

give better performance. As it is difficult to know in advance B, we replace it in the expression of $\eta(t)$ by the parameter M defined in Section 1.1.

Performance

The results are reported for $\mathrm{pen}_t(m) = tm$.

FS-EWA	FS-EG
27.5	28.5

Graphical evolution of the weights

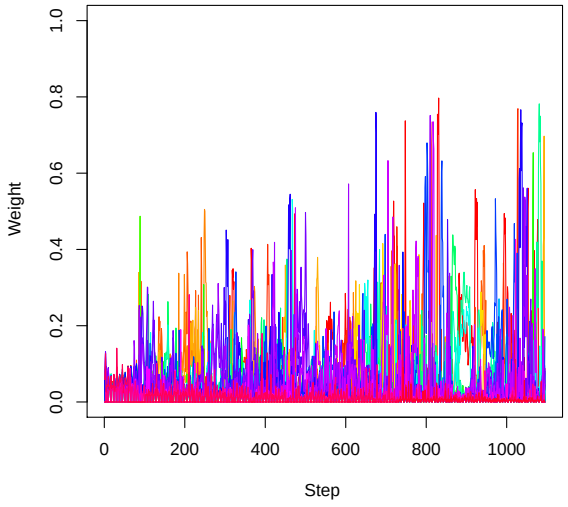

Figure 3.8: Evolution of the weights of the variant described above of the FS-EWA (with the penalty $\mathrm{pen}_t(m) = tm$).

Chapter 4

Another data set: French data

4.1 Description of the French data

We now consider an other data set. This data set consists in half-hourly observations of the French comsumption of electricity. The units are Gigawatts and the period of reference is from 01/09/2007 to 31/08/2008. For the sake of simplicity, we remove data that correspond to special days (e.g., public holidays), the day before and the day after them, as well as the first two weeks of May and the winter holidays.

4.1.1 Description of the experts

In this section we present the French Data and the individual experts used in this study to forecast French load data. These experts belong to three different classes of forecasting methods:

1. parametric methods – Eventail (based on the Metehore model);
2. semi-parametric methods – GAM (Generalized Additive Model);
3. non-parametric methods – similarity forecasts on functional data.

French load data

The data for this study are classically used for the estimation and the validation of the short-term EDF models calibration. Thus, we consider 5 years of consolidated half hourly data (CPRC) for the estimation set (from September 2002 to September 2007) and one year of real time load estimation (ETR) for the validation part. This data are given by RTE, the French national grid company.

The Corse and the Eurodif data are subtracted from the CPRC data from the Eventail and GAM model estimation. For the GAM model, the Eurodif data are also subtracted from the ETR data.

Meteorological data

The meteorological data consist in half hourly (interpolated) temperature and nebulosity data. This data are provided by MeteoFrance and come from 26 meteorological stations all over the French territory.

Eventail model

The Eventail software is one of the tools in used ad EDF to produce electricity load forecast based on the Metehore model. The reader is referred to [BDR05] for a presentation of this parametric model.

For the calibration of the basic Eventail model we follow the methodology in use at EDF (Kheops model). The short-term correction is performed with the classical weights $(0.4, 0.1, 0.09, 0.08, 0.07, 0.06, 0.2)$. From this basic model we can derive a family of Eventail models by considering different heating gradients (see Table 4.1) and different weights in percentage for the short-term correction (see Table 4.2).

Heating Gradient	-1350	-1450	**-1550**	-1650	-1750

Table 4.1: Values for the heating gradient for the Eventail individual predictors (MW/Celsius degrees)

Lag	48	96	144	192	240	288	336
Weights1	40	10	9	8	7	6	20
Weights2	80	20	-	-	-	-	-
Weights2	70	-	-	-	-	-	30

Table 4.2: Lags and Weights

GAM model

A GAM model has been calibrated to forecast the French electricity load (work in progress, to be presented at the CFE / CSDA conference in Cyprus, in October 2009). In this modeling the relation between the electricity load and the various predictors (meteorological conditions, calendar events, etc.) is supposed to be additive and non-linear. We summarize this model in the following equation:.

$$\begin{aligned} y_i^h =& f_1\left(y_{i-1}^h\right) + f_2\left(T_i^h\right) + f_3\left(T_{i-1}^h\right) + f_4\left(T_{i-2}^h\right) \\ &+ f_5\left(\mathrm{Tmin}_i^h\right) + f_6\left(\mathrm{Tmax}_i^h\right) + f_7\left(\mathrm{Neb}_i^h\right) \\ &+ f_8\left(\mathrm{posan}_i^h\right) + a^h J_i^h + b^h T_i^h \\ &+ \varepsilon_i \end{aligned} \tag{4.1}$$

where

- y_i^h is the electricity load at the h–th hour of the i–th day of the estimation set,
- Tmin_i^h is the minimal temperature over the last 23 hours, while Tmax_i^h has a similar definition with maxima,
- Neb_i^h is the cloud cover,
- posan_i^h is the position in the year (the number of days since the beginning of the year estimated with periodic splines),
- J_i^h is the day type effect,
- T_i^h is the trend.

In practice, the lag and the seasonality effects are specified by day type (one by day in a week).

Different GAM models can be obtained from this model. One effect of interest is the trend effect as one of the main issue is the extrapolation of low frequency effects from one

year to another. Thus, we derive 3 additional GAM models with different values for the intercept point of the trend effect ((-25,+25,+50) % of the basic intercept point). We also generate a GAM forecaster with linear lag effects instead of non-linear ones, a GAM with no lag effects, and a two-stage GAM model where the lag effects and the other effects are estimated separately in a two-stage process.

Similarity forecasts on functional data

This similarity forecaster can be assimilated to a "blind" method in comparison to the other methods considered above. The principles of this method applied on classical data are presented in [Pog94], see [APS06] for an application to functional data. The development of this method is the topicc of a PhD thesis at OSIRIS (EDF) and this forecaster was provided by Jairo Cugliari.

4.1.2 Numerical values

For practical purposes and computational issues we convert our data in Gigawatts. Unlike in previous chapters we do not split the data set in 48 fixed-hour data because otherwise we would not have large enough samples. Nevertheless, to preserve the operational constraint that every day at noon predictions should be made for the whole next day, the weights of our forecasters are computed every day at noon and used for the whole next day with a renormalization taking into account the sleeping aspects when necessary. (There are thus only 320 real rounds, as the table below shows.)

As in Section 1.1, M is some typical order of magnitude for the observations y_t, N is the number of experts, and K the number of different values taken by E_t.

n	$n/48$	M	N	K
15 360	320	50	24	7

Some standard RMSE are summarized below. As expected, $B_{\mathbb{R}^N}$ has a less disappointing performance than before since few experts only are sleeping occasionally (see Figure 4.2). $B_{\leqslant m}$ refers to the best performance obtained by a compound expert of size at most m.

B_{oracle}	$B_{\mathbb{R}^N}$	$B^{(a)}_{\text{ave}}$	$B^{(b)}_{\text{renorm}}$	$B^{(b)}_{\text{ave}}$	$B_{\text{best exp}}$	$B_{\leqslant 50}$	$B_{\leqslant 100}$
0.223	0.620	0.724	0.696	0.748	0.782	0.534	0.474

A summary of the global performance of the experts is plotted in Figure 4.1. In Figure 4.2, we plot the global performance of each expert with respect to its percentage of activity.

The colors refer to the type of the expert: dark blue is for the Eventail experts, pink is for the GAM experts, and light blue is for the functional expert. The Eventail and similarity experts are always active whereas the GAM experts are inactive the day before and after special days (those triples of days are totally excluded).

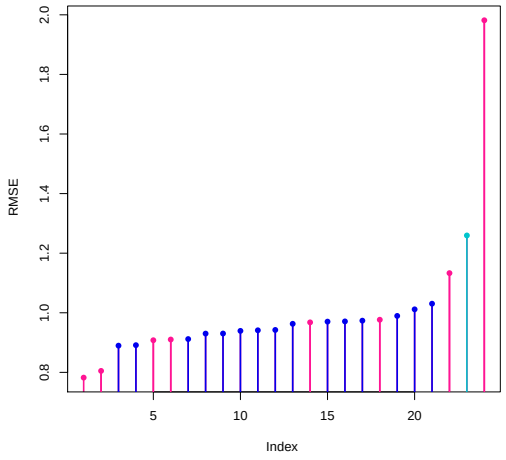

Figure 4.1: Performance of experts

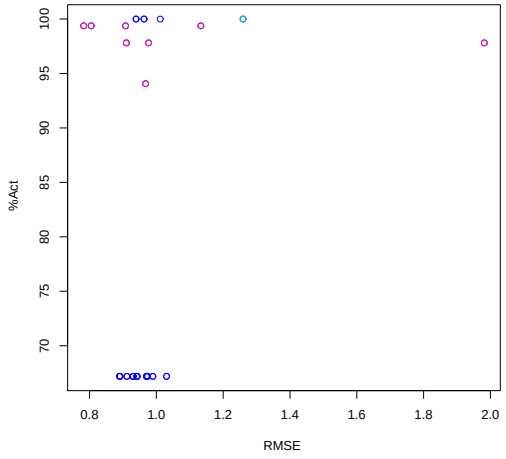

Figure 4.2: Performance of experts

4.2 Results obtained by the forecasters

We present in this section the performance of some forecasters presented in Chapter 2. As explained in Section 4.1 weights (living in $\mathbb{R}^N$ or in the simplex of order N) are computed each day at noon and used for the whole next day with a renormalization taking into account the sleeping aspects when necessary. In the sequel, we make the assumption that it is noon at $t = 1$, thus it is noon if and only if t is of the form $48k+1$ for some integer k.

For several forecasters we give results both with uniform initial weights and with the following initial distribution, which we call the fair distribution: $(1/3, 1/3, 1/3)$ uniform distribution between the three types of experts and uniform distribution over experts of the same type (for instance the initial weight of an Eventail expert is $1/(3 \times 15)$, whereas the functional expert has a $1/3$ weight).

4.2.1 Exponentially weighted average forecaster

We use a variant of the forecaster described in Section 2.2, where we still use an exponentially weighted average but where the cumulative regrets $R'_{i,t}$ are updated only at rounds of the form $t = 48k+1$, for integers k. (At these rounds, at most 48 instantaneous regrets with respect to i are added.)

Uniform distribution

η	1e-6	1e-5	1e-4	1e-3	1e-2	$\eta^\star =$ 1e-4
RMSE	0.724	0.722	0.718	0.731	0.788	0.718

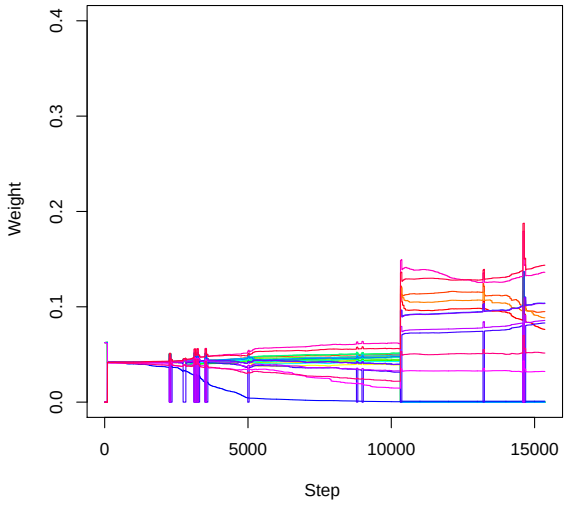

Figure 4.3: Evolution of the weights for $\eta^\star =$ 1e-4

Fair distribution

η	1e-6	1e-5	1e-4	1e-3	1e-2	$\eta^\star = 2e\text{-}4$
RMSE	0.736	0.731	0.695	0.722	0.789	0.683

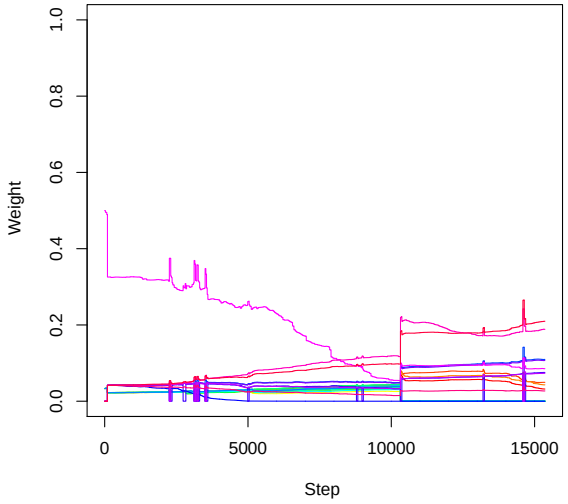

Figure 4.4: Evolution of the weights for $\eta^\star = 2e\text{-}4$

4.2.2 Exponentiated gradient forecaster

We adapt the forecaster described in Section 2.3 as we adapted the exponentially weighted average forecaster in the section above (by updating only every 48 rounds and taking into account sums of at most 48 instantaneous regrets).

Uniform distribution

η	1e-6	1e-5	1e-4	1e-3	1e-2	$\eta^\star =$ 5e-3
RMSE	0.724	0.722	0.712	0.683	0.668	0.650

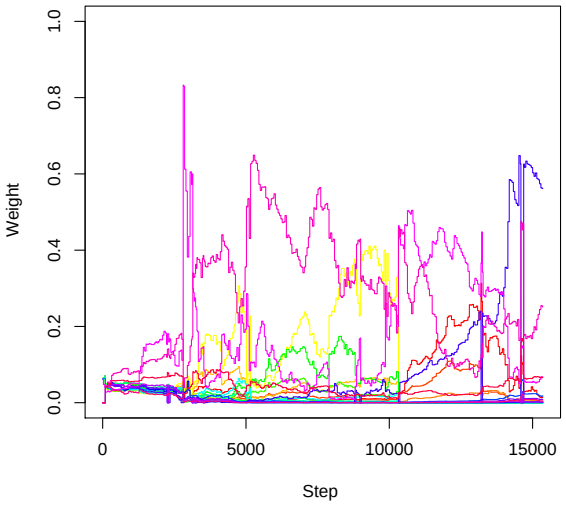

Figure 4.5: Evolution of the weights for $\eta^\star =$ 5e-3

Fair distribution

η	1e-6	1e-5	1e-4	1e-3	1e-2	$\eta^\star = 5e\text{-}3$
RMSE	0.737	0.733	0.711	0.674	0.670	0.651

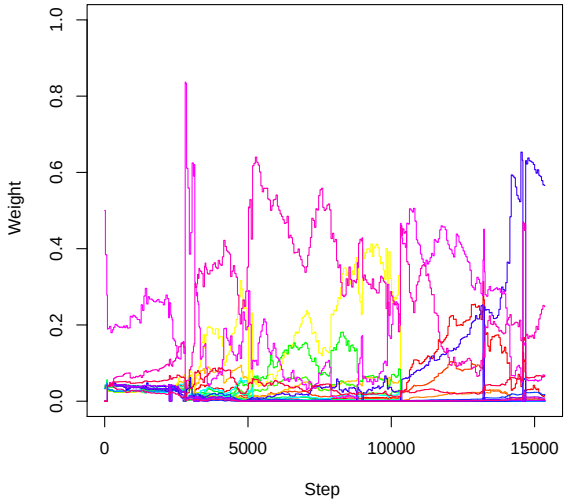

Figure 4.6: Evolution of the weights for $\eta^\star = 5e\text{-}3$

4.2.3 Mixed lazy–plug-in version of Zinkevich's forecaster

We adapt the forecasters described in Section 2.6 as follows. (We only provide the version with uniform initial allocation.)

Parameters: $\eta > 0$
Initialization: $\mathbf{p}_1 = (1/N, \dots, 1/N)$
For each round $t = 1, 2, \dots, n$,

(1) predict with $\widehat{y}_t = P_t(\mathbf{p}_t) \cdot \mathbf{f}_t$;

(2) if t is not a multiple of 48, then $\mathbf{p}_{t+1} = \mathbf{p}_t$;

(3) if t is a multiple of 48, then

$$\mathbf{p}_{t+1} = P_{t-47}(\mathbf{p}_{t-47}) - 2\eta \sum_{s=1}^{48} (\widehat{y}_{t+1-s} - y_{t+1-s})(\mathbb{I}_{t+1-s} * \mathbf{f}_{t+1-s}) \ .$$

Comments: A totally lazy version would simply omit P_{t-47} in step (3); its results are worse than the mixed version we provide here. No totally plug-in version exists, since there is no commun value for the simplexes of probability of the next 48 steps. This is why we provide this mixed version. The choice of P_{t-47} in step (3) is quite arbitrary, we checked that choosing different values (e.g., P_t, P_{t-1}, P_{t-2}, ..., P_{t-46}) had only a limited impact.

Uniform distribution

We provide the results in the case of uniform initial weights only (the results are similar in the case of an initial fair distribution).

η	1e-5	1e-4	1e-3	1e-2	$\eta^\star =$ 6e-4
RMSE	0.725	0.727	0.690	1.127	0.676

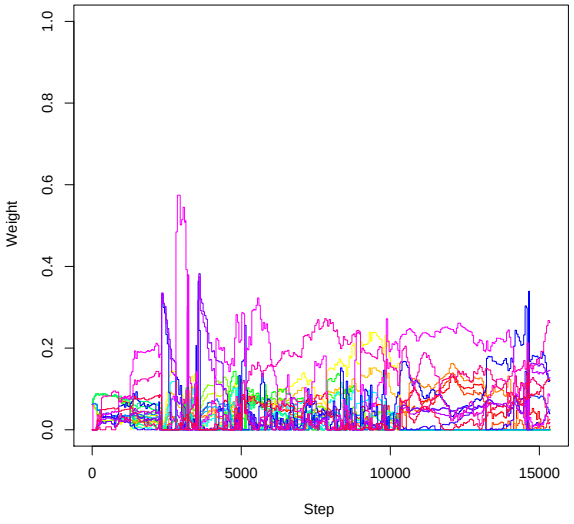

Figure 4.7: Evolution of the weights for $\eta^\star$ =6e-4

4.2.4 Ridge regression forecaster

We use the forecaster described in Section 2.7, except that we force it to only output a new vector of weights $\mathbf{u}_t$ at rounds of the form $t = 48k + 1$ and to use it for the next 48 rounds. (No renormalization is involved this time.)

Uniform distribution

We give the performance of this forecaster run with uniform initial weights and the graphical evolution of the weights it outputs. Even though the target oracle $B_{\mathbb{R}^N}$ is quite good, the performance of this forecaster is poor and is quite far from this target.

We checked that this is because we fix the vectors of weights for the whole next day (i.e., for 48 rounds): in an additional experiment we allowed ridge updates at each half-hour and the performance came close to $B_{\mathbb{R}^N}$.

λ	1	1e+3	1e+6	1e+9	$\lambda^\star$ = 2e+2
RMSE	1.176	0.840	0.863	0.867	0.833

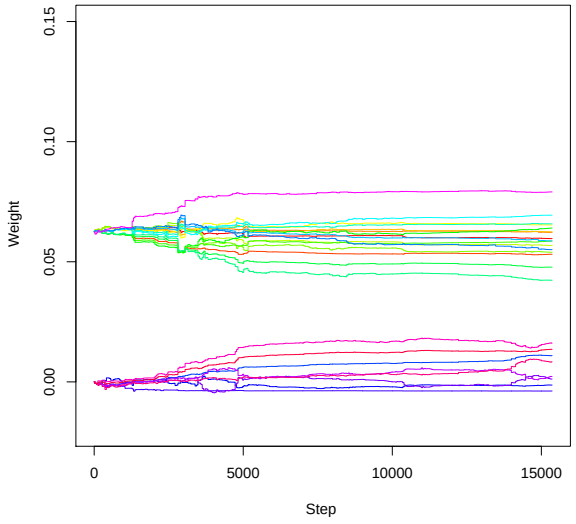

Figure 4.8: Evolution of the weights for $\lambda^\star$ = 2e+2

4.2.5 Renormalized exponentiated gradient forecaster

We adapt the forecaster described in Section 2.5.

Parameters: learning rate η
Initialization: $p_{i,1} = 1/N$ for $i = 1, \dots, N$
For each round $t = 1, 2, \dots, n$

(1) predict with $\widehat{y}_t = \mathbf{p}_t^{E_t} \cdot \mathbf{f}_t$;

(2) if t is not a multiple of 48, then $\mathbf{p}_{t+1} = \mathbf{p}_t$;

(3) if t is a multiple of 48, then compute $\mathbf{p}_{t+1}$ as follows: for all $i = 1, \dots, N$,

$$p_{i,t+1} = p_{i,t-47} \prod_{s=0}^{47} \left(\mathbb{I}_{\{i \notin E_{t-s}\}} + \mathbb{I}_{\{i \in E_{t-s}\}} \frac{e^{-2\eta f_{i,t-s}(\widehat{y}_{t-s} - y_{t-s})} \sum_{i \in E_{t-s}} p_{i,t-47}}{\sum_{i \in E_{t-s}} p_{i,t-47}\, e^{-2\eta f_{i,t-s}(\widehat{y}_{t-s} - y_{t-s})}} \right).$$

Uniform distribution

We only provide the performance in case of uniform initial weights.

η	1e-7	1e-6	1e-5	1e-4	$\eta^\star =$ 3e-6
RMSE	0.724	0.723	0.742	0.940	0.722

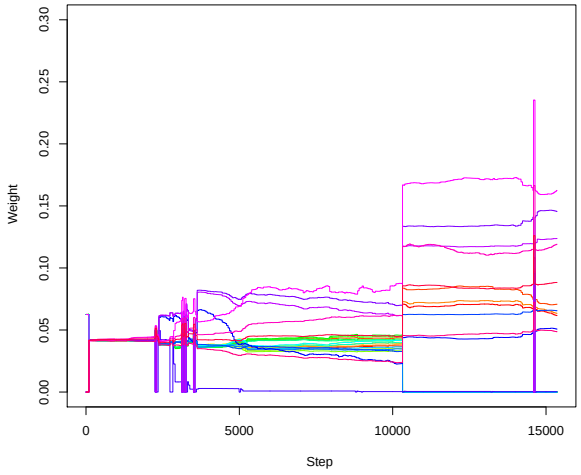

Figure 4.9: Evolution of the weights for $\eta^\star = 3\text{e-}6$

4.2.6 Fixed-share forecasters

We now present the adaptations of the forecasters of Sections 2.10 and 2.11. Since they are similar we only present in details the one for the exponentially weighted average variant of the fixed-share forecaster.

Exponentially weighted average variant of the fixed-share forecaster

The forecaster is implemented as follows.

Parameters: $\eta > 0$ and $0 \leqslant \alpha \leqslant 1$
Initialization: $\mathbf{w}_1 = \mathbf{v}_1 = (1/N, \dots, 1/N)$
For each round $t = 1, 2, \dots, n$,

(1) predict with $\widehat{y}_t = \mathbf{p}_t^{E_t} \cdot \mathbf{f}_t$;

(2) if t is not a multiple of 48, then $\mathbf{p}_{t+1} = \mathbf{p}_t$ and $\mathbf{v}_{t+1} = \mathbf{w}_t$

(3) if t is a multiple of 48, perform the loss update: for each $i = 1, \dots, N$,

$$v_{i,t+1} = w_{i,t}\, e^{\eta \sum_{s=t-48}^{t-1} \mathbb{I}_{\{i \in E_s\}} \left(\widehat{\ell}_s - \ell_{i,s}\right)} ;$$

(4) perform the share update: for each $i = 1, \dots, N$,

$$w_{i,t+1} = \frac{1}{|E_{t+1}|} \sum_{j \in E_t \setminus E_{t+1}} v_{j,t+1} + \frac{\alpha}{|E_{t+1}|} \sum_{j \in E_t \cap E_{t+1}} v_{j,t+1} + (1-\alpha)\, \mathbb{I}_{\{i \in E_t \cap E_{t+1}\}} v_{i,t+1}$$

(with the convention that an empty sum is null);

(5) if t is a multiple of 48, then $\mathbf{p}_{t+1} = \mathbf{w}_{t+1}$.

The results with the uniform initial distribution follow.

	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.1$	$\alpha = 0.2$
$\eta =$ 1e-5	0.725	0.724	0.724	0.724
$\eta =$ 1e-4	0.723	0.723	0.723	0.723
$\eta =$ 1e-3	0.708	0.711	0.711	0.713
$\eta =$ 1e-2	0.681	0.677	0.678	0.682
$\eta =$ 1e-1	0.753	0.716	0.716	0.715

The best performance is achieved for $(\eta, \alpha) =$ (3e-2, 0.1) and its value is 0.665.

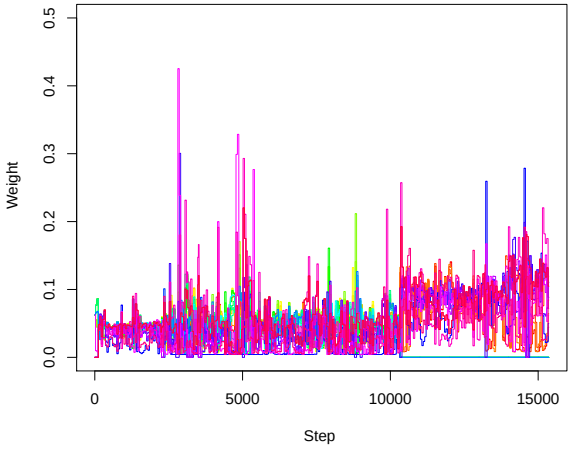

Figure 4.10: Evolution of the weights for $(\eta, \alpha)^\star =$ (3e-2, 0.1)

Exponentiated gradient variant of the fixed-share forecaster

The results with the uniform initial distribution follow.

	$\alpha = 0.01$	$\alpha = 0.05$	$\alpha = 0.1$	$\alpha = 0.2$
$\eta =$ 1e-5	0.725	0.724	0.724	0.725
$\eta =$ 1e-4	0.723	0.723	0.723	0.723
$\eta =$ 1e-3	0.676	0.685	0.684	0.683
$\eta =$ 1e-2	0.761	0.778	0.767	0.754
$\eta =$ 1e-1	0.708	0.712	0.713	0.714

The best performance is achieved for $(\eta, \alpha)^\star =$ (6e-3, 3e-4) and its value is 0.656.

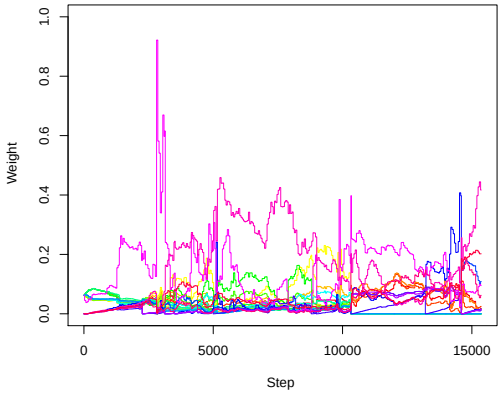

Figure 4.11: Evolution of the weights for $(\eta, \alpha)^\star = (6e\text{-}3, \quad 3e\text{-}4)$

These results are somewhat disappointing compared to the performance obtained in Chapter 2. This can be explained by the fact that the sleeping aspect is reduced in comparison to the Slovakian data set.

4.2.7 On-line calibration of the parameters with a grid

We use here the trick presented in Section 3.1. We apply calibration via empirical loss minimization (ELM) at five prediction methods adapted above: the exponentially weighted average forecaster, the exponentiated gradient forecaster, the plug-in version of Zinkevich's forecaster, the exponentially weighted average and the exponentiated gradient variants of the fixed-share forecaster.

We chose in the first three cases a uniform logarithmic grid on [1e-6, 1e-2], with $\left|\widetilde{\Lambda}\right| = 21$. For the fixed-share forecasters we use the following grid:

$$\widetilde{\Lambda} = \{\text{1e-5, 1e-4, 1e-3, 1e-2, 1e-1}\} \times \{0.01, 0.05, 0.1, 0.2\} \, .$$

The performance is summarized below. The "offline" line stands for the results obtained by the best parameters tuned offline whereas "online" refers to the performance of the calibration method alluded at above. In parentheses we indicate whether we used a uniform or a fair initial weight allocation.

	EWA (fair)	EG (fair)	Zink (unif.)	FS-EWA (unif.)	FS-EG (unif.)
online	0.692	0.661	0.684	0.697	0.662
offline	0.683	0.651	0.676	0.675	0.655

The evolutions of the weights are shown in Figure 4.12 for the first three algorithms and in Figure 4.13 for the fixed-share forecasters.

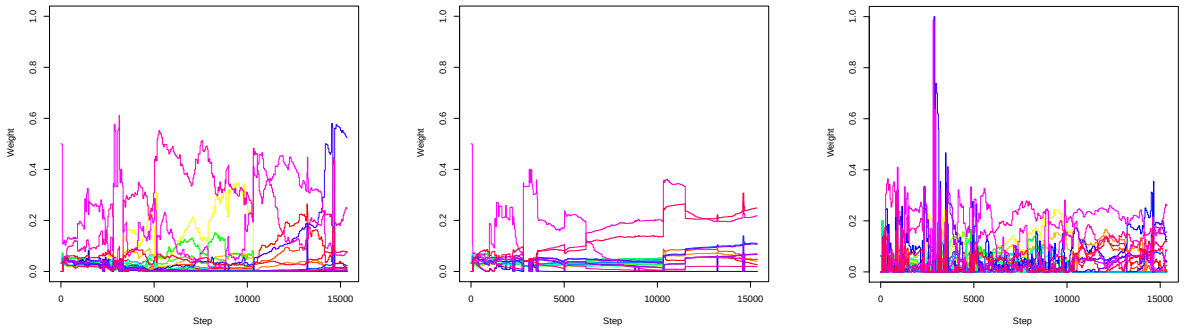

Figure 4.12: Evolution of the weights for the exponentiated gradient forecaster (left), the exponentially weighted average forecaster (middle) and Zinkevich's forecaster (right). For the three forecasters, the size of the grid was $\left|\widetilde{\Lambda}\right| = 21$.

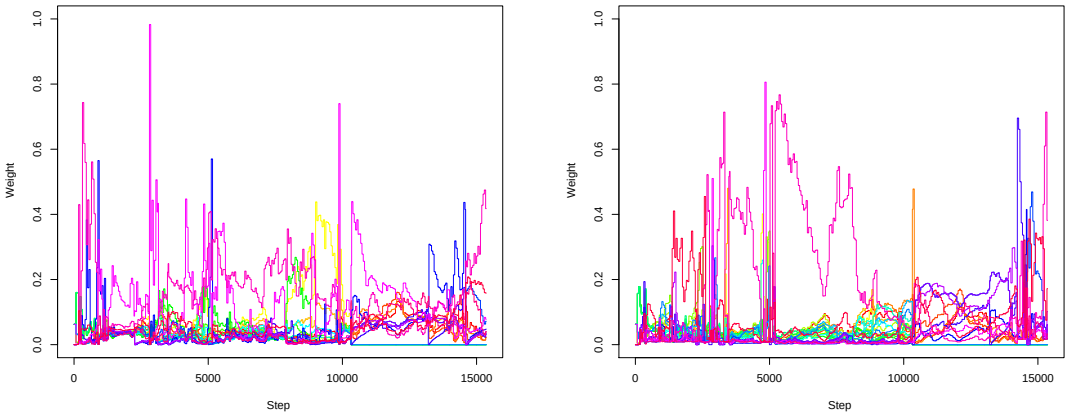

Figure 4.13: Evolution of the weights for exponentiated gradient (left), and the exponentially weighted average (right) versions of the fixed-share forecaster.

We also provide in Figure 4.14 the evolution of the calibrated parameters for the first three forecasters.

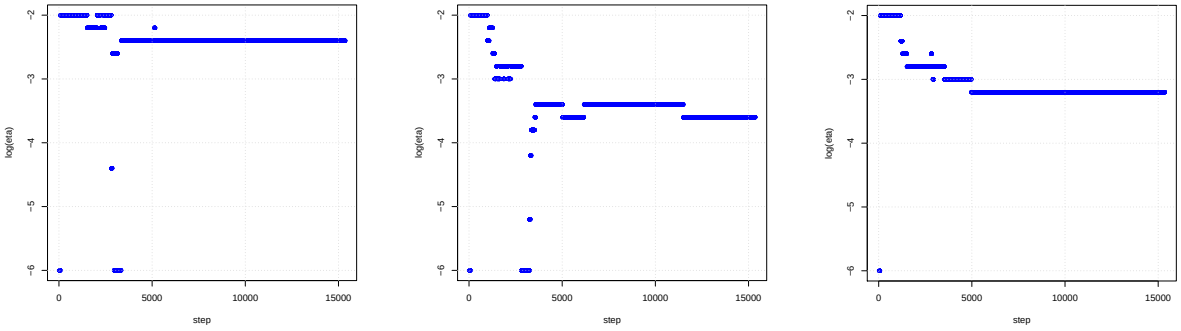

Figure 4.14: Evolution of the calibrated parameters for the exponentiated gradient forecaster (left), the exponentially weighted average forecaster (middle) and Zinkevich's forecaster (right). For the three forecasters, the size of the grid was $\left|\tilde{\Lambda}\right| = 21$.

Bibliography

[APS06] Anestis Antoniadis, Efstathios Paparoditis, and Theofanis Sapatinas. A functional wavelet-kernel approach for continuous-time prediction. *Journal of the Royal Statistical Society (Series B)*, 68:837–857, 2006.

[BDR05] Alexander Bruhns, Gilles Deurveilher, and Jean-Sébastien Roy. A non-linear regression model for mid-term load forecasting and improvements in seasonnality. In *15th Power Systems Computation Conference*, 2005.

[BM05] Avrim Blum and Yishay Mansour. From external to internal regret. In *Proceedings of COLT*, pages 621–636, 2005.

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

[CBL06] Nicolò Cesa-Bianchi and Gàbor Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[FSS97] Yoav Freund, Robert E. Schapire, and Yoram Singer. Using and combining predictors that specialize. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing*, pages 334–343, 1997.

[GMS08] Sébastien Gerchinovitz, Vivien Mallet, and Gilles Stoltz. A further look at sequential aggregation rules for ozone ensemble forecasting, 2008. Available at http://www.dma.ens.fr/~stoltz/GeMaSt-report.pdf.

[Pog94] Jean-Michel Poggi. Prévision non paramétrique de la consommation électrique. *Revue de Statistique Appliquée*, 42:83–98, 1994.

[Zin03] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the Twentieth International Conference on Machine Learning*, pages 928–936, 2003.