Projet Machine Learning pour la Prévision: agrégation séquentielle d'experts

Yannig Goude

Introduction

Le problème que l'on se pose ici sort du cadre statistique classique présentés dans les cours précédent (échantillon de (X, Y), loi stationnaire).

Plus précisément on cherche à prévoir des valeurs $y_1, ..., y_n$ observées de manière séquentielle au cours du temps.

Nous ne faisons pas l'hypothèse que ces observations sont des réalisations d'un processus stochastique sous-jacent. Le but est d'avoir des résultats valables sur l'ensemble des suite d'observations possibles.

Pour cela nous restreignons un peu le problème au cas suivant:

- on suppose que l'on dispose de N prédicteurs (appelés également experts) f_j, $j = 1, ..., N$ qui effectuent une prévision à l'instant t de y_t notée $f_{j,t}$.
- ces prédicteurs peuvent êtres issus d'une modélisation statistique, d'un modèle physique, d'une expertise humaine...

On se fixe comme objectif d'agréger séquentiellement ces experts de manière à produire la "meilleure" (au sens d'une fonction de perte à définir) prévision de y_t possible basée sur le passé: $y_1, ..., y_{t-1}$ et $(f_{j,1}, ..., f_{j,t-1})_{j=1,..,N}$.

Bibliographie

L'historique de ces travaux remonte aux publications de Hannan (1957) et Blackwell and others (1956) dans le cadre de la théorie des jeux.

En théorie de l'apprentissage, ce problème de prévision séquentielle de suites arbitraires a été proposé par Littlestone and Warmuth (1994) et V. G. Vovk (1990) ; Cesa-Bianchi et al. (1997), Freund et al. (1997) et V. Vovk (1998) ont présenté quelques-un des résultats fondamentaux. Pour un état de l'art complet voir Cesa-Bianchi and Lugosi (2006).

Formulation du problème

A chaque instant t le prévisionniste doit fournir une prévision $\widehat{y}_t$ de $y_t \in \mathbb{Y}$ basée sur les observations passées $y_1, ..., y_{t-1}$. $\widehat{y}_t \in \mathbb{X}$

ou $\mathbb{X}$ est un ensemble convexe potentiellement différent de $\mathbb{Y}$ (par exemple $\mathbb{X}$ est l'enveloppe convexe de $\mathbb{Y}$).

Pour mesurer les performances de prévision obtenues le prévisionniste dispose d'une fonction de perte: $l : \mathbb{X} \times \mathbb{Y} \to \mathbb{R}_+$ qu'il cherche à minimiser sur le long terme, ie à minimiser la perte cumulée après T instants (T étant amené à tendre vers l'infini):

$$\sum_{t=1}^{T} l(\widehat{y}_t, y_t)$$

Pour cela le prévisionniste a accès à des experts qui proposent à chaque échéance une prévision basée sur le passé.

Plus précisément, on suppose que l'on dispose de N experts f_j, $j = 1, ..., N$ qui effectuent une prévision à l'instant t de y_t notée $f_{j,t}$, qui dépend des observations $y_1, ..., y_{t-1}$ et éventuellement d'autres informations propres à chaque expert.

Le prévisioniste a accès au passé de ces experts $(f_{j,1}, ..., f_{j,t-1})_{j=1,..,N}$, les observations passés de y: $y_1, ..., y_{t-1}$ et les prévisions des experts pour la future observation t: $f_{j,t}$.

Pour l'instant, on suppose que le prévisionniste cherche à construire sa prévision comme une combinaison convexe des prévisions des experts.

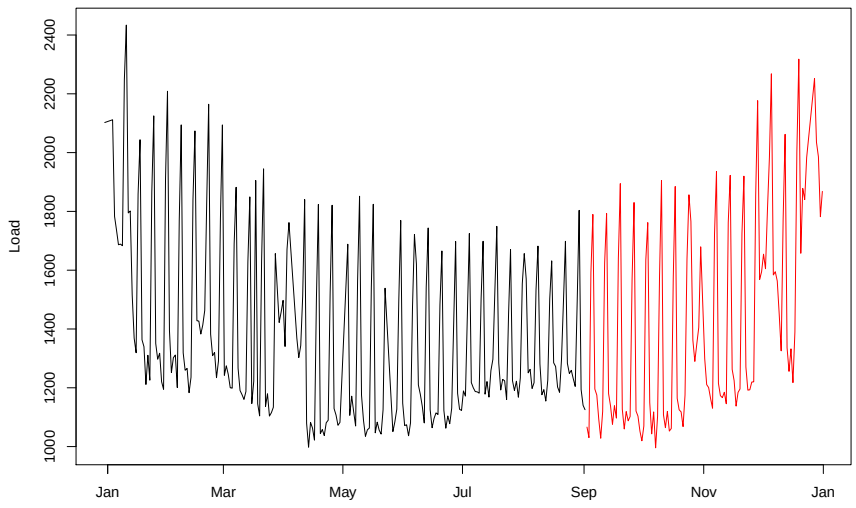
Une stratégie $\boldsymbol{S}$ de prévision associe à l'information disponible au début de chaque échéance t un vecteur de poids $p_t = (p_{1,t}, ..., p_{N,t})$ et une prévision $\widehat{y}_t$:

$$\widehat{y}_t = \sum_{j=1}^{N} p_{j,t} f_{j,t}$$

ou p_t est dans le simplexe $\mathbb{P} \in \mathbb{R}^N$, ie:

$$\sum_{j=1}^{N} p_{j,t} = 1;\ p_{j,t} \geq 0$$

Illustration, données Irlandaises de consommation électrique (agrégat d'environ 3000 consommateurs résidentiels):




```
##  [1] "time"     "toy"      "dow"      "holy"     "tod"      "temp"
##  [7] "dateTime" "dem"      "dem48"    "temp95"   "dem336"   "Date"
```

Nous proposons les experts suivants: persistance, RF, SARIMA, GAM, GBM, CART linéaire.

```
library(randomForest); library(gbm);
library(mgcv); library(party);

##########################################persistance
expert_persistancy_forecast <- IrishAgg1$dem336

##########################################RF
set.seed(150)
formula_ML <- dem ~ toy + dow  + tod + temp + dem48 +
  dem336 + temp95
expert_rf <- randomForest(formula_ML, data = IrishAgg0)
expert_rf_forecast <- predict(expert_rf, newdata=IrishAgg1)

##########################################SARIMA
ts<-ts(IrishAgg0$dem, frequency = 7)
arima_fit <-forecast::Arima(ts, order=c(5,0,2),
                            seasonal = c(2, 0, 0),method=c("CSS"))
arima_forecast<-rollArima(arima_fit, ynew=IrishAgg1$dem,horizon=1)
```

```
#########################################GBM
expert_gbm <- gbm(formula_ML, data=IrishAgg0,
                  distribution = "gaussian",
                  n.trees=100, n.minobsinnode = 5,
                  shrinkage = 0.05, bag.fraction = 0.5,
                  train.fraction = 0.9)
best.iter=gbm.perf(expert_gbm, method="OOB", plot.it = F,
                   oobag.curve = T)
expert_gbm_forecast <- predict(expert_gbm, newdata=IrishAgg1,
                               n.trees=best.iter)

#########################################GAM
formula_gam <- dem ~ dow + s(dem48, k=10) + s(dem336, k=10)  +
  s(temp, k=20) + s(temp95, k=20)
expert_gam <- mgcv::gam(formula_gam, data=IrishAgg0,
                        family = gaussian)
expert_gam_forecast <- predict(expert_gam, newdata=IrishAgg1)

#########################################MOB
formula_mob <- dem ~ temp + dem48 + dem336 + temp95 | toy +
  dow  + tod + temp + dem48 + dem336 + temp95
expert_mob <- mob(formula_mob, data = IrishAgg0, model = linearModel)
expert_mob_forecast <- predict(expert_mob, newdata=IrishAgg1)
```

Pour quantifier les performances du prévisionniste, il est utile d'avoir une référence à laquelle se comparer. On pourrait évaluer la performance de l'agrégation de manière absolue mais ça aurait peu de sens. En effet, si tout les experts sont "mauvais" (plus généralement si toutes combinaison convexe des experts est mauvaise), il est peu problable que l'agrégation soit performante. Ainsi, on définie la notion de regret d'une stratégie d'agrégation $\boldsymbol{S}$:

Le regret de $\boldsymbol{S}$ relativement à q (combinaison convexe à poids fixés q) après T prévisions successives est :

$$R_T^{conv}(\boldsymbol{S}) = \widehat{L}_T(\boldsymbol{S}) - \min_{q \in \mathbb{P}} L_T(q)$$

et le regret $\boldsymbol{S}$ relativement au meilleur expert après T prévisions successives est:

$$R_T^{best}(\boldsymbol{S}) = \widehat{L}_T(\boldsymbol{S}) - \min_{j \in 1,\dots,N} L_T(\delta_j)$$

Calcul des Oracles

```
library(opera)
experts <- cbind(expert_persistancy_forecast, expert_rf_forecast,
                 arima_forecast, expert_gbm_forecast,
                 expert_gam_forecast, expert_mob_forecast)
colnames(experts)<-c("pers", "rf", "arima", "gbm", "gam",
                     "mob")
or <- oracle(IrishAgg1$dem, experts, model = "convex",
             loss.type = "square")
```

```
or
```

```
## Call:
## oracle.default(Y = IrishAgg1$dem, experts = experts, model = "convex
##     loss.type = "square")
##
## Coefficients:
##   pers        rf arima      gbm  gam   mob
##  0.171 -3.87e-18     0 3.09e-17 0.61 0.219
##
##                          rmse   mape
## Best expert oracle:       169 0.0672
## Uniform combination:      174 0.0725
## Best convex oracle:       161 0.0618
```

```
rmse_exp <- apply(experts, 2,
                  function(x){sqrt(mean((x-IrishAgg1$dem)^2))})
rmse_exp%>%round(, digits=0)%>%sort
```

```
##   gam   mob    rf   gbm arima  pers
##   169   177   184   190   210   221
```

Le **regret** de $\boldsymbol{S}$ relativement à q sur les T premières échéances est:

$$R_T(\boldsymbol{S}) = \widehat{L}_T(\boldsymbol{S}) - \min_{q \in \mathbb{P}} L_T(q)$$

si l'on suppose que la fonction de perte est bornée (ce qui est raisonnable dans la plupart des applications pratiques), le regret $R_T(\boldsymbol{S})$ est de l'ordre au plus de T. L'objectif du prévisionniste est de se construire une stratégie $\boldsymbol{S}$ telle que:

$$\text{limsup}_{t \to \infty} \sup \frac{R_T(\boldsymbol{S})}{T} \leq 0$$

où le supremum porte sur l'ensemble des suites d'observations et de prévisions possibles.

Stratégies et algorithmes d'agrégation

Algorithme Exponentially Weighted Aggregation (EWA)

- fixer le paramètre $\eta > 0$
- fixer les poids initiaux $p_{j,1} = 1/N$
- calculer la prévision de y_1: $\widehat{y}_1 = \sum_{j=1}^{N} p_{j,1} f_{j,1}$
- Pour $t = 2, ..., T$:
  - calculer les poids:

$$p_{j,t} = \frac{e^{-\eta L_{t-1}(\delta_j)}}{\sum_{q=1}^{N} e^{-\eta L_{t-1}(\delta_q)}}$$

  - puis l'agrégation:

$$\widehat{y}_t = \sum_{j=1}^{N} p_{j,t} f_{j,t}$$

Théoriquement, cet algorithme et la stratégie d'agrégation $\boldsymbol{E}_\eta$ associée permet d'obtenir un des résultats les plus connus en prévision de suites individuelles.

Théorème

On suppose que

- la fonction de perte $l : \mathbb{X} \times \mathbb{Y} \to [0, M]$ est bornée
- l est convexe en son premier argument, l'application $x \to l(x, y)$ est convexe

alors $\forall \eta$,

$$\sup \widehat{L}_T(\boldsymbol{E}_\eta) - \min_j L_T(\delta_j) \leq \frac{\ln(N)}{\eta} + \frac{\eta M^2}{8} T$$

le sup est pris sur l'ensemble des trajectoires possibles d'observations et de prévisions des experts.

Si l'on choisit $\eta = 1/M\sqrt{8\ln(N)/T}$, on obtient la majoration suivante:

$$\sup \widehat{L}_T(\boldsymbol{E}_\eta) - \min_j L_T(\delta_j) \leq M\sqrt{T/2\ln(N)}$$

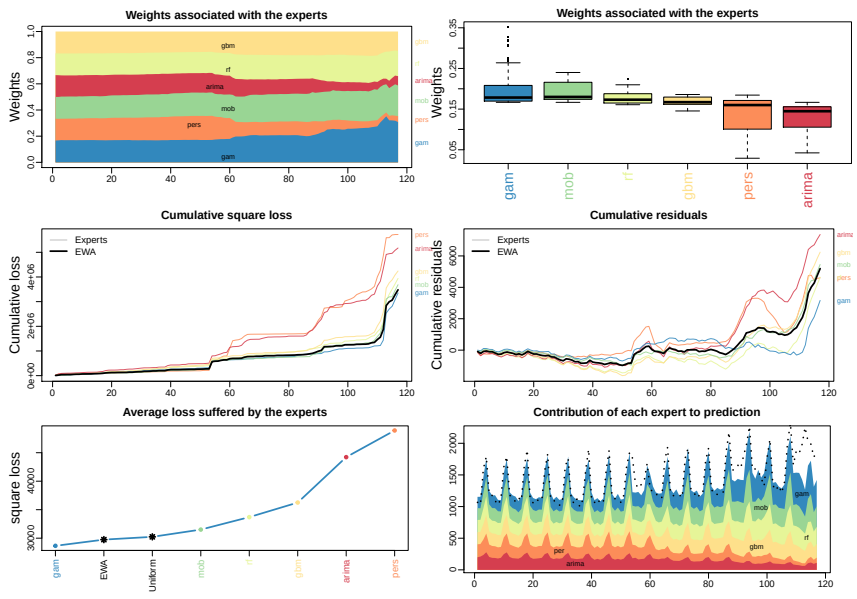
pour la preuve, cf Stoltz (2010).

EWA, learning rate théorique

```
M <- (IrishAgg0$dem-IrishAgg0$dem336)^2%>%mean
learning.rate <-  (1/M)*sqrt(8*log(ncol(experts)))/nrow(IrishAgg1)

agg.online_theoric<- mixture(Y = IrishAgg1$dem , experts = experts,
                             model = 'EWA', loss.type = "square",
                             loss.gradient = F,
                             parameter=list(eta=learning.rate))

plot(agg.online_theoric)
```



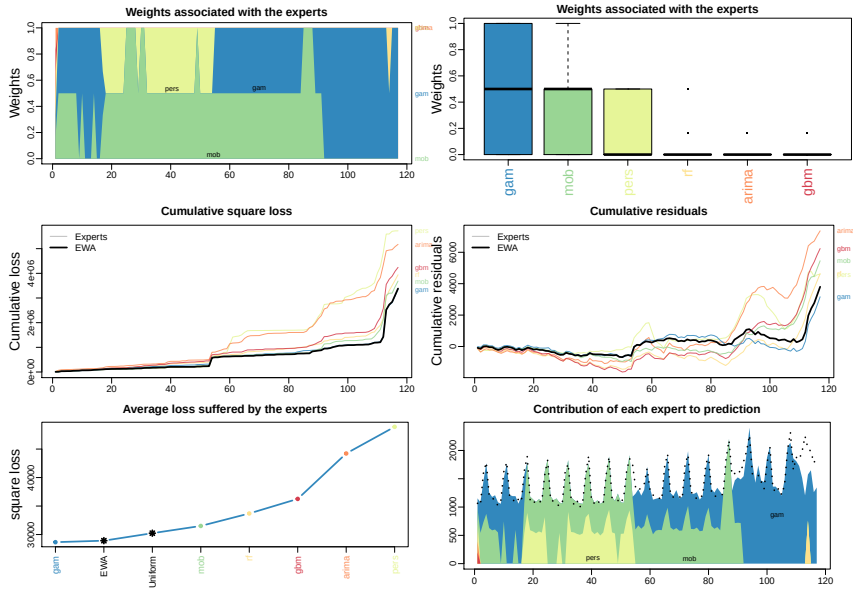
EWA, learning rate théorique

```
summary(agg.online_theoric)
```

```
## Aggregation rule: EWA
## Loss function:  square loss
## Gradient trick:  FALSE
## Coefficients:
##    pers    rf  arima   gbm   gam   mob
##  0.0522 0.191 0.0779 0.154 0.294 0.231
##
## Number of experts:  6
## Number of observations:  117
## Dimension of the data:  1
##
##         rmse   mape
## EWA      172 0.0697
## Uniform  174 0.0725
```

EWA, learning rate optimisé en ligne

```
agg.online<- mixture(Y = IrishAgg1$dem , experts = experts,
                     model = 'EWA', loss.type = "square",
                     loss.gradient = F)
plot(agg.online)
```



EWA, learning rate optimisé en ligne

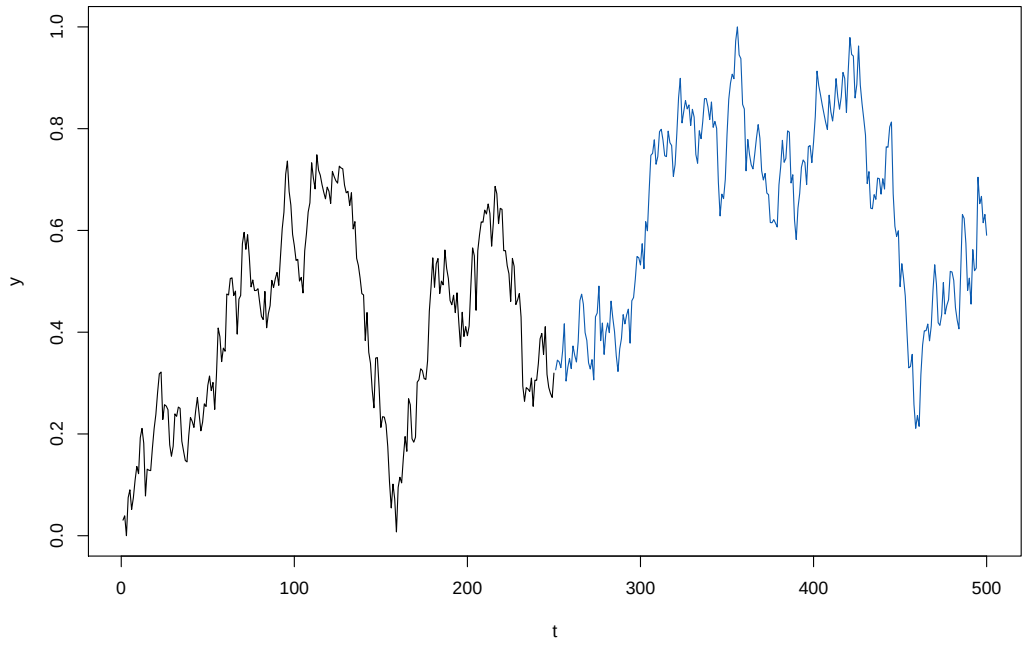
```
summary(agg.online)
```

```
## Aggregation rule: EWA
## Loss function:  square loss
## Gradient trick:  FALSE
## Coefficients:
##  pers rf arima gbm gam mob
##     0  0     0   0   1   0
##
## Number of experts:  6
## Number of observations:  117
## Dimension of the data:  1
##
##         rmse   mape
## EWA      170 0.0633
## Uniform  174 0.0725
```

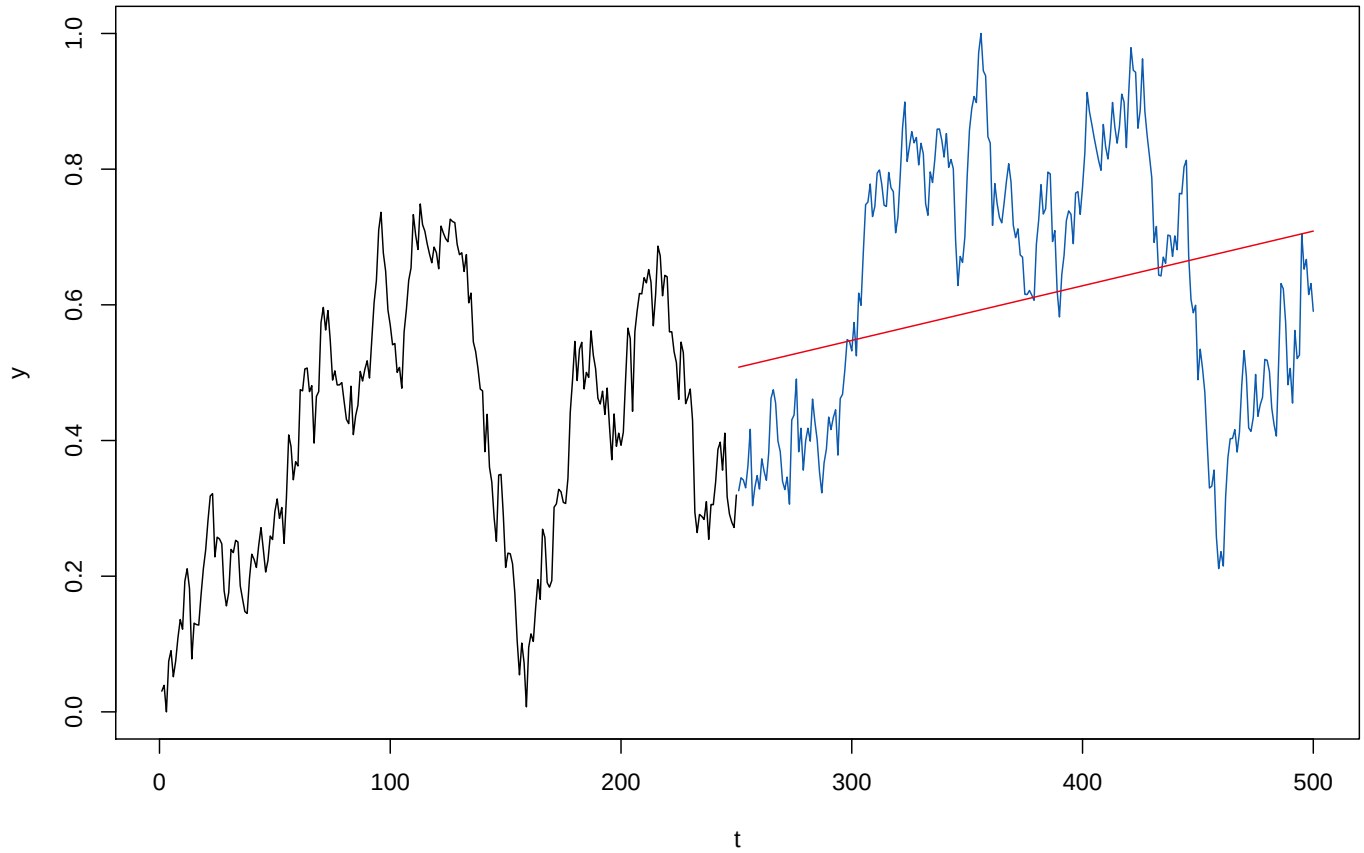
Illustration pratique: données simulées

Nous considérons le problème de prévision (simulé) suivant. Un prévisionniste a observé 250 observations d'un processus y_t et cherche à prévoir les 250 suivantes.

```
T <- 500
set.seed(1)
eps <- rnorm(T, 0, 1)
y <- cumsum(eps)
y <- (y-min(y))/(max(y)-min(y))
```

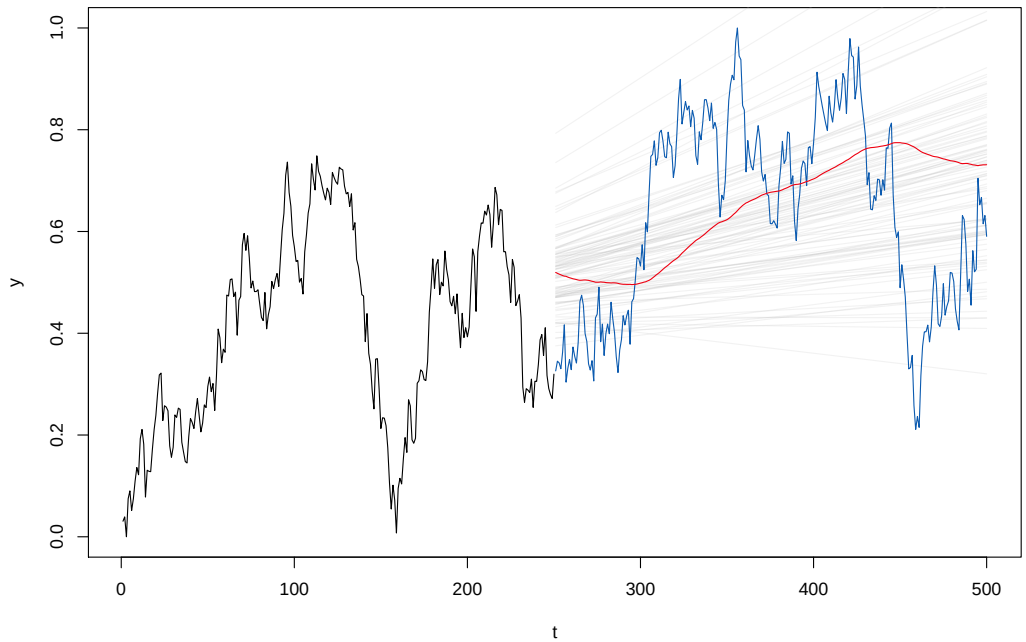


Il choisit d'utiliser une méthode de régression linéaire pour cela.



Pour améliorer sa prévision, il a l'idée de se fabriquer un ensemble d'experts en "bagguant" son prédicteur initial N fois puis en agrégeant en ligne ces experts à l'aide de l'algorithme EWA.

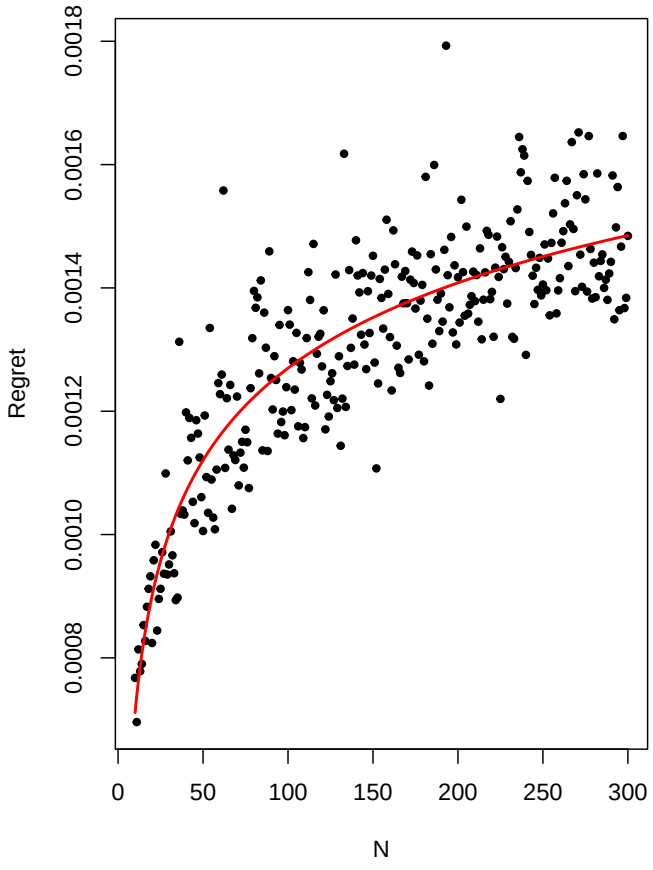
Il choisit comme paramètre η la valeur théorique optimale $\eta = 1/M\sqrt{8\ln(N)/T}$. L'horizon de prévision étant connu, $T = 250$. Il choisit de se construire $N = 100$ experts. Reste à choisir M. Il décide d'estimer M sur l'échantillon de données out of bag résultant du sous échantillonnage.

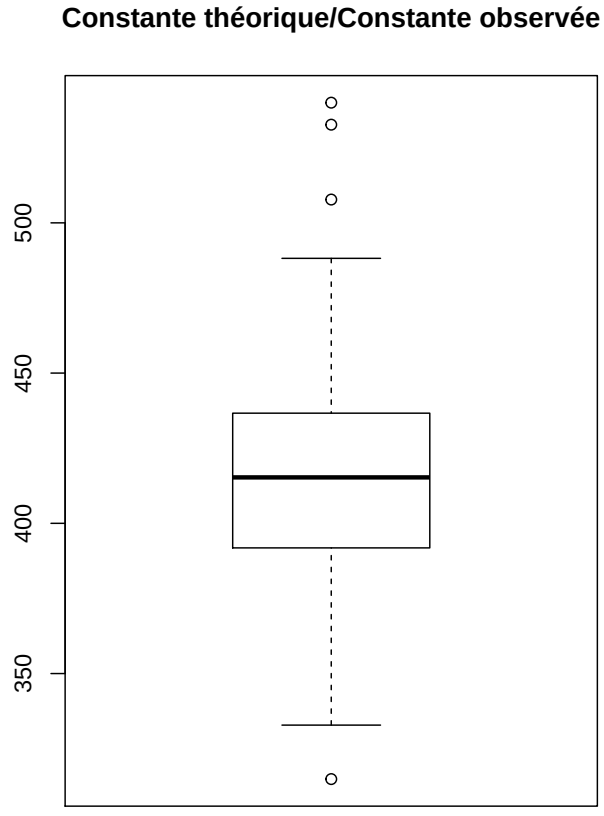


Il décide ensuite de vérifier borne oracle associée à EWA. Pour cela il fait varier le nombre d'experts de 10 à 300 et calcul le regret $\widehat{L}_T(\boldsymbol{E}_\eta) - \min_j L_T(\delta_j)$, qu'il ajuste par régression sur $\sqrt{log(N)}$

```
##
## Call:
## lm(formula = borne ~ I(sqrt(log(N))))
##
## Residuals:
##       Min         1Q     Median         3Q        Max
## -2.473e-04 -6.470e-05 -1.226e-05  5.777e-05  3.915e-04
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -6.366e-04  6.468e-05  -9.842   <2e-16 ***
## I(sqrt(log(N)))  8.883e-04  2.947e-05  30.149   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.405e-05 on 289 degrees of freedom
## Multiple R-squared:  0.7588, Adjusted R-squared:  0.7579
## F-statistic: 908.9 on 1 and 289 DF,  p-value: < 2.2e-16
```

soit graphiquement:





L'ajustement observé est très bon et le regret évolue bien en $\sqrt{\log(N)}$. En revanche la constante observée est très loin de la constante théorique $M\sqrt{T/2}$. La borne théorique est excessivement prudente, ce qui était attendu car cette borne prévaut pour toutes les séquences d'observations possibles alors que nous sommes ici sur un modèle de processus particulier.

La stratégie $\boldsymbol{E}_\eta$ permet d'approcher les performances en prévision du meilleur expert. On peut chercher à approcher la meilleure combinaison convexe d'experts. Pour ça on suppose que X est un sous-ensemble convexe de $\mathbb{R}^d$, que la fonction $l(., y)$ est différenciable sur $\mathbb{X}$, $\forall y \in \mathbb{Y}$. Alors:

$\forall y$, $\exists\ \partial(., y)$ un gradient de l en tout point de $\mathbb{X}$ tel que $\forall\ (u, v) \in \mathbb{X}$:

$$l(u, y) - l(v, y) \leq \partial(u, v).(u - v)$$

On note $\tilde{l}_{j,t} = \partial l(\sum_{k=1}^{N} p_{j,t} f_{j,t}, y_t) f_{j,t}$ et $\tilde{L}_T(\delta_j) = \sum_{t=1}^{T} \tilde{l}_{j,t}$

Exponential Gradient (EG)

Algorithme Exponential Gradient (EG)

- fixer le paramètre $\eta > 0$
- fixer les poids initiaux $p_{j,1} = 1/N$
- calculer la prévision de y_1: $y_1 = \sum_{j=1}^{N} p_{j,1} f_{j,1}$
- pour $t = 2, ..., T$:
  - calculer les poids:

$$p_{j,t} = \frac{e^{-\eta \tilde{L}_{t-1}(\delta_j)}}{\sum_{q=1}^{N} e^{-\eta \tilde{L}_{t-1}(\delta_q)}}$$

  - puis l'agrégation:

$$\widehat{y}_t = \sum_{j=1}^{N} p_{j,t} f_{j,t}$$

Exponential Gradient (EG)

Cet algortihme et la stratégie E_η^{grad} associée permet d'obtenir la borne oracle:

Theorem

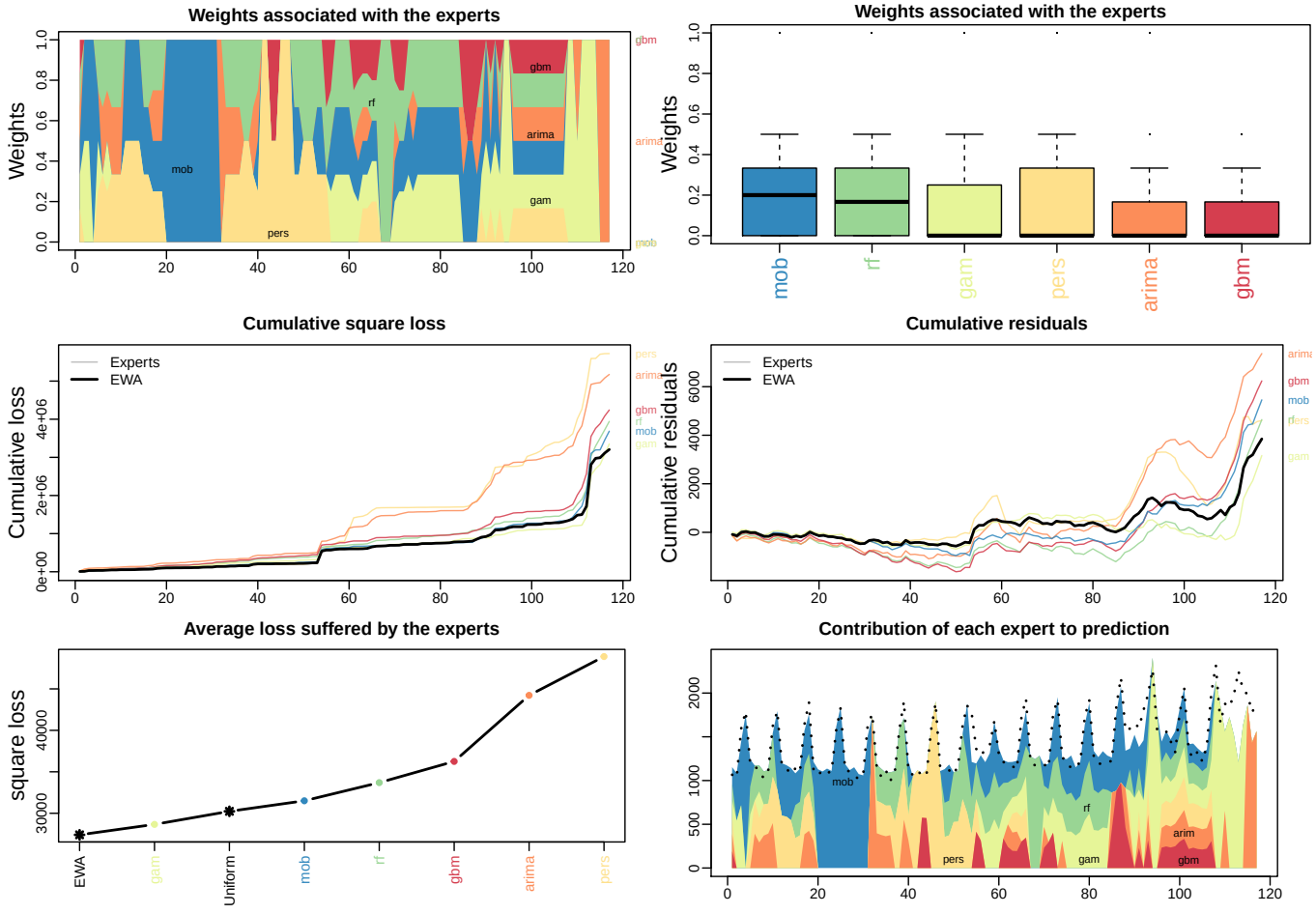
$$\sup \widehat{L}_T(\boldsymbol{E}_\eta^{grad}) - \min_q L_T(q) \leq \frac{\ln(N)}{\eta} + \frac{\eta C^2}{2} T$$

les pseudo-pertes $\tilde{l}_{j,t} \in]-C, C[$.

Exponential Gradient (EG)

```
## Aggregation rule: EWA
## Loss function:  square loss
## Gradient trick:  250
## Coefficients:
##  pers rf arima gbm gam mob
##     0  0     1   0   0   0
##
## Number of experts:  6
## Number of observations:  117
## Dimension of the data:  1
##
##         rmse   mape
## EWA      166 0.0651
## Uniform  174 0.0725
```

Exponential Gradient (EG)



Agregation avec paramètre d'apprentissage adaptatif

Du fait de l'importance du paramètre d'apprentissage η et de sa calibration, des algorithmes ont été developpés pour le calibré et le mettre à jour en ligne. L'algorihme MLpol réalise celà.

Soit $R_t(S) = L_t(S) - \min_{j \in 1,\dots,N} L_T(\delta_j)$ le regret d'une stratégie d'agrégation (ici S est soit la stratégie δ_j soit MLpol) relativement au meilleurs expert et $R_t(S)^+$ sa partie positive.

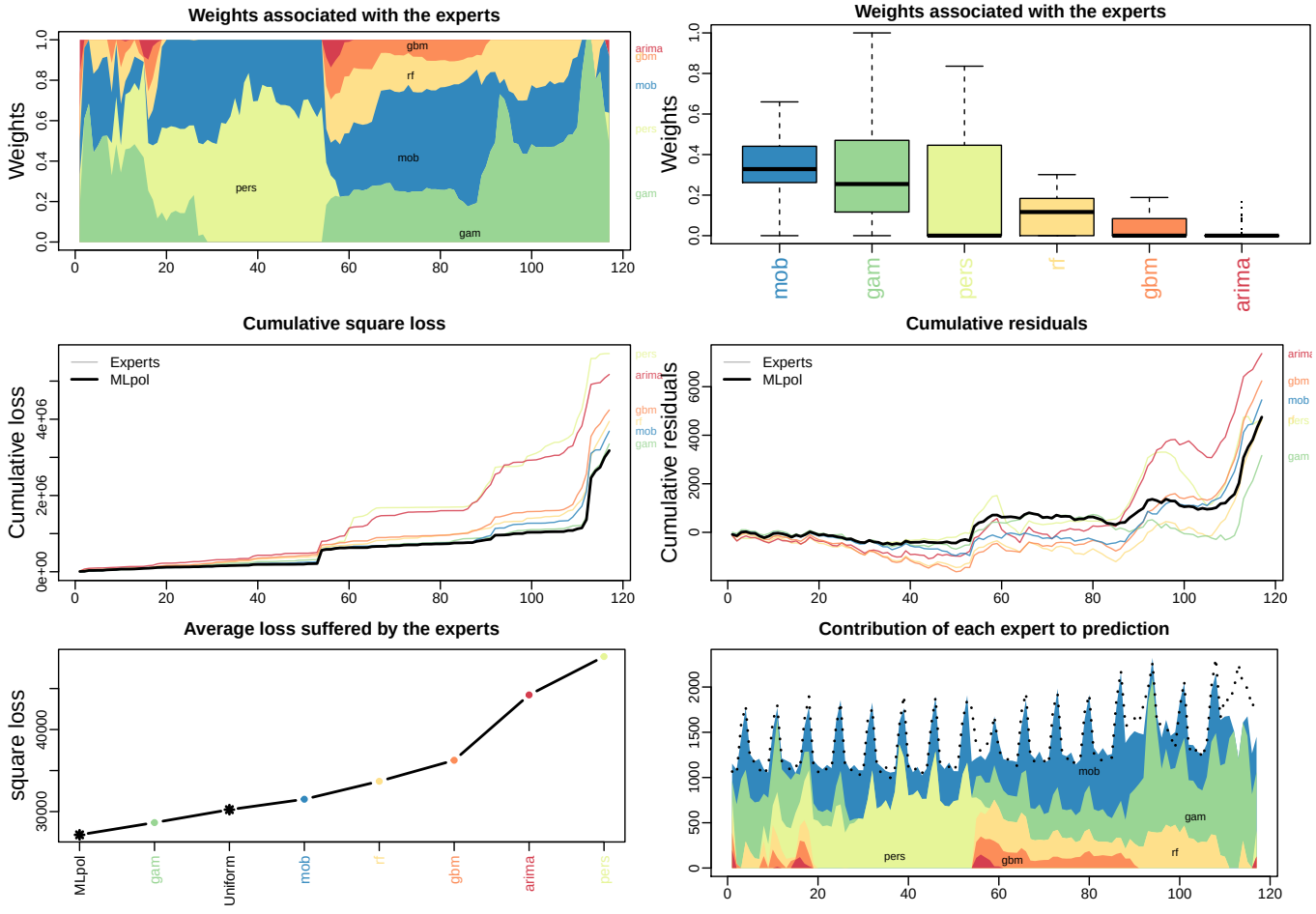
Algorithme MLpol

- set $\eta > 0$
- set initial weights to $p_{j,1} = 1/N$
- initialize $\widehat{y}_1 = \sum_{j=1}^{N} p_{j,1} f_{j,1}$
- for $t = 2, \dots, T$
  - for each expert j, pick the learning rates: $\eta_{j,t-1} = 1/\left(1 + \sum_{s=1}^{t-1} (l(\widehat{y}_s, y_s) - l(f_{j,s}, y_s))^2\right)$
  - update the weights: $p_{j,t} = \eta_{j,t-1} \frac{R_t(\delta_j)^+}{R_t(MLpol)^+}$
  - then aggregation: $\widehat{y}_t = \sum_{j=1}^{N} p_{j,t} f_{j,t}$

MLpol

```
## Aggregation rule: MLpol
## Loss function:  square loss
## Gradient trick:  250
## Coefficients:
##   pers rf arima gbm   gam   mob
##  0.295  0 0.139   0 0.353 0.213
##
## Number of experts:  6
## Number of observations:  117
## Dimension of the data:  1
##
##          rmse   mape
## MLpol     165 0.0603
## Uniform   174 0.0725
```

MLpol


Weights associated with the experts
Weights
gbm
rf
mob
pers
gam
arima
Weights associated with the experts
Weights
mob
gam
pers
rf
gbm
arima
Cumulative square loss
Cumulative loss
Experts
MLpol
Cumulative residuals
Cumulative residuals
Experts
MLpol
Average loss suffered by the experts
square loss
MLpol
gam
Uniform
mob
rf
gbm
arima
pers
Contribution of each expert to prediction
mob
gam
pers
gbm
rf

non-convex aggregation: Ridge

si les données sont hors de l'enveloppe convexe générée par les experts: agrégation linéaire. Les experts étant souvant corrélés on envisage une régression pénalisée de type ridge. Le paramètre de pénalité $\lambda > 0$ peut être optimisé en ligne (online grid search) ou fixé. L'option "Ridge" de la fonction mixture d'opera permet les 2 options.

Algorithme Ridge

- set $\eta > 0$
- set initial weights to $p_{j,1} = 1/N$
- intialize to: $\widehat{y}_1 = \sum_{j=1}^{N} p_{j,1} f_{j,1}$
- for $t = 2, ..., T$:
  - compute the weights :

$$p_{j,t} = \text{argmin}_{u \in \mathbb{R}^N} \{ \sum_{s=1}^{t-1} (y_t - \sum_{j=1}^{N} u_j f_{j,s}))^2 + \lambda (u - p_1)^2 \}$$

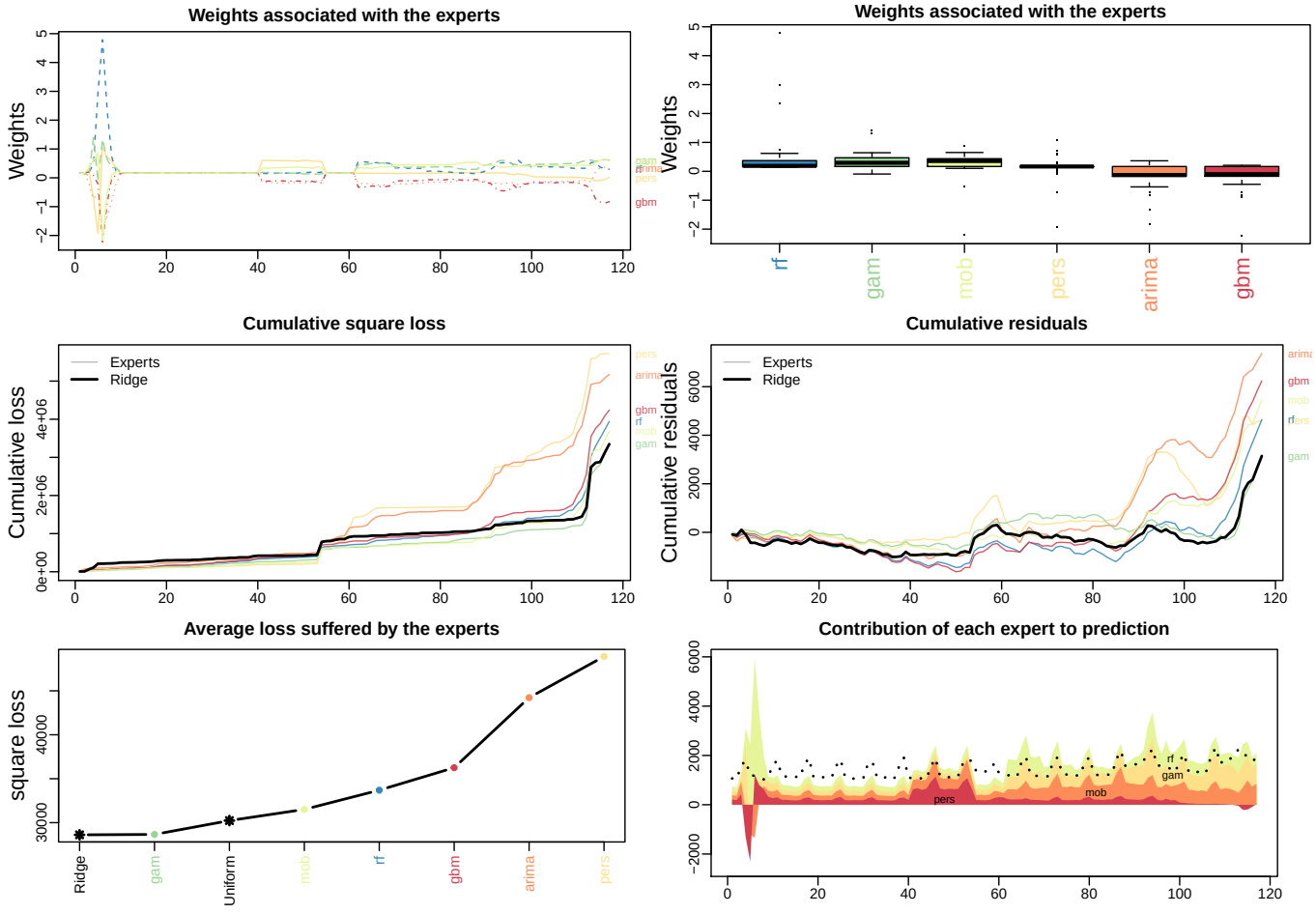
  - and the aggregation :

$$\widehat{y}_t = \sum_{j=1}^{N} p_{j,t} f_{j,t}$$

Ridge

```
## Aggregation rule: Ridge
## Loss function:  square loss
## Gradient trick:
## Coefficients:
##   pers    rf arima    gbm   gam   mob
##  0.098 0.159  0.27 -0.565 0.596 0.467
##
## Number of experts:  6
## Number of observations:  117
## Dimension of the data:  1
##
##         rmse   mape
## Ridge    169 0.0670
## Uniform  174 0.0725
```

Ridge


Weights associated with the experts
Weights
Weights associated with the experts
Weights
rf
gam
mob
pers
arima
gbm
Cumulative square loss
Experts
Ridge
Cumulative loss
Cumulative residuals
Experts
Ridge
Cumulative residuals
Average loss suffered by the experts
square loss
Ridge
gam
Uniform
mob
rf
gbm
arima
pers
Contribution of each expert to prediction
rf
gam
pers
mob

Comment choisir les experts?

comme dans les méthodes d'ensemble en général, il faut favoriser la diversité:

$$(y_t - \widehat{y}_t)^2 = \frac{1}{K}\sum_{k=1}^{K}(y_t - x_{k,t})^2 - \frac{1}{K}\sum_{k=1}^{K}(x_{k,t} - \widehat{y}_t)^2$$

Pour celà on peut envisager plusieurs approches:

- entrainer des modèles exploitant des données variées: périodes d'estimation, variables d'entrées, résolution spatiale ou temporelle
- considérer des méthodes d'apprentissage variées: functional data analysis, méthodes linéaires, boosting...
- considérer des fonctions de pertes variées: quantile loss, L2, L1...

on peut aussi générer des experts issus d'un ensemble d'expert (voir Gaillard and Goude (2015)): bagging, boosting, diversity boosting (Reeve and Brown (2018), Bourel et al. (2020)), stacking (Ting and Witten (1997), Capezza et al. (2020)), perturbation aléatoires

References

Blackwell, David, and others. 1956. "An Analog of the Minimax Theorem for Vector Payoffs." *Pacific Journal of Mathematics* 6 (1). Pacific Journal of Mathematics: 1–8.

Bourel, Mathias, Jairo Cugliari, Yannig Goude, and Jean-Michel Poggi. 2020. "Boosting Diversity in Regression Ensembles."

Capezza, Christian, Biagio Palumbo, Yannig Goude, Simon N Wood, and Matteo Fasiolo. 2020. "Additive Stacking for Disaggregate Electricity Demand Forecasting." *arXiv Preprint arXiv:2005.10092*.

Cesa-Bianchi, Nicolo, and Gábor Lugosi. 2006. *Prediction, Learning, and Games*. Cambridge university press.

Cesa-Bianchi, Nicolo, Yoav Freund, David Haussler, David P Helmbold, Robert E Schapire, and Manfred K Warmuth. 1997. "How to Use Expert Advice." *Journal of the ACM (JACM)* 44 (3). ACM New York, NY, USA: 427–85.

Freund, Yoav, Robert E Schapire, Yoram Singer, and Manfred K Warmuth. 1997. "Using and Combining Predictors That Specialize." In *Proceedings of the Twenty-Ninth Annual Acm Symposium on Theory of Computing*, 334–43.

Gaillard, Pierre, and Yannig Goude. 2015. "Forecasting Electricity Consumption by Aggregating Experts; How to Design a Good Set of Experts." In *Modeling and Stochastic Learning for Forecasting in High Dimensions*, 95–115. Springer.

Hannan, James. 1957. "Approximation to Bayes Risk in Repeated Play." *Contributions to the Theory of Games* 3: 97–139.

Littlestone, Nick, and Manfred K Warmuth. 1994. "The Weighted Majority Algorithm." *Information and Computation* 108 (2). Elsevier: 212–61.

Reeve, Henry WJ, and Gavin Brown. 2018. "Diversity and Degrees of Freedom in Regression Ensembles." *Neurocomputing* 298: 55–68.

Stoltz, Gilles. 2010. "Agrégation Séquentielle de Prédicteurs: Méthodologie Générale et Applications à La Prévision de La Qualité de L'air et à Celle de La Consommation électrique." *Journal de La Société Française de Statistique* 151 (2): 66–106.

Ting, Kai Ming, and Ian H Witten. 1997. "Stacking Bagged and Dagged Models."

Vovk, Vladimir. 1998. "A Game of Prediction with Expert Advice." *Journal of Computer and System Sciences* 56 (2). Elsevier: 153–73.