Contributions à l'agrégation séquentielle robuste d'experts : Travaux sur l'erreur d'approximation et la prévision en loi. Applications à la prévision pour les marchés de l'énergie.

Pierre Gaillard

UNIVERSITÉ PARIS SUD
Comprendre le monde, construire l'avenir®

EDF

Université Paris-Sud

Ecole Doctorale de Mathématiques de la région Paris-Sud

Discipline : mathématiques

Thèse de doctorat

Soutenue le 6 juillet 2015 par

Pierre GAILLARD

Contributions à l'agrégation séquentielle robuste d'experts : travaux sur l'erreur d'approximation et la prévision en loi. Applications à la prévision pour les marchés de l'énergie.

Directeur de thèse :	M. Gilles STOLTZ	Chargé de Recherche (HEC Paris)
Rapporteurs :	M. Peter GRÜNWALD	Professeur (Leiden University et CWI)
	M. Eric MOULINES	Professeur (Télécom ParisTech)

Composition du jury :

Directeur de thèse :	M. Gilles STOLTZ	Chargé de Recherche (HEC Paris)
Président du jury :	M. Pascal MASSART	Professeur (Université Paris-Sud)
Rapporteur :	M. Peter GRÜNWALD	Professeur (Leiden University et CWI)
Examinateurs :	Mme. Anne PHILIPPE	Professeur (Université de Nantes)
	M. Georges HEBRAIL	Chercheur senior (EDF R&D)
	M. Olivier WINTENBERGER	Professeur (Copenhagen University et Université Pierre et Marie Curie)
Responsable industriel :	M. Yannig GOUDE	Ingénieur chercheur (EDF R&D et Université Paris-Sud)

Contributions à l'agrégation séquentielle robuste d'experts : travaux sur l'erreur d'approximation et la prévision en loi. Applications à la prévision pour les marchés de l'énergie.

Résumé : Nous nous intéressons à prévoir séquentiellement une suite arbitraire d'observations. À chaque instant, des experts nous proposent des prévisions de la prochaine observation. Nous formons alors notre prévision en mélangeant celles des experts. C'est le cadre de l'agrégation séquentielle d'experts. L'objectif est d'assurer un faible regret cumulé. En d'autres mots, nous souhaitons que notre perte cumulée ne dépasse pas trop celle du meilleur expert sur le long terme. Nous cherchons des garanties très robustes : aucune hypothèse stochastique sur la suite d'observations à prévoir n'est faite. Celle-ci est supposée arbitraire et nous souhaitons des garanties qui soient vérifiées quoi qu'il arrive. Un premier objectif de ce travail est l'amélioration de la performance des prévisions. Plusieurs possibilités sont proposées. Un exemple est la création d'algorithmes adaptatifs qui cherchent à s'adapter automatiquement à la difficulté de la suite à prévoir. Un autre repose sur la création de nouveaux experts à inclure au mélange pour apporter de la diversité dans l'ensemble d'experts. Un deuxième objectif de la thèse est d'assortir les prévisions d'une mesure d'incertitude, voire de prévoir des lois. Les applications pratiques sont nombreuses. En effet, très peu d'hypothèses sont faites sur les données. Le côté séquentiel permet entre autres de traiter de grands ensembles de données. Nous considérons dans cette thèse divers jeux de données du monde de l'énergie (consommation électrique, prix de l'électricité,...) pour montrer l'universalité de l'approche.

Contributions to online robust aggregation: work on the approximation error and on probabilistic forecasting. Applications to forecasting for energy markets.

Abstract: we are interested in online forecasting of an arbitrary sequence of observations. At each time step, some experts provide predictions of the next observation. Then, we form our prediction by combining the expert forecasts. This is the setting of online robust aggregation of experts. The goal is to ensure a small cumulative regret. In other words, we want that our cumulative loss does not exceed too much the one of the best expert. We are looking for worst-case guarantees: no stochastic assumption on the data to be predicted is made. The sequence of observations is arbitrary. A first objective of this work is to improve the prediction accuracy. We investigate several possibilities. An example is to design fully automatic procedures that can exploit simplicity of the data whenever it is present. Another relies on working on the expert set so as to improve its diversity. A second objective of this work is to produce probabilistic predictions. We are interested in coupling the point prediction with a measure of uncertainty (i.e., interval forecasts,...). The real world applications of the above setting are multiple. Indeed, very few assumptions are made on the data. Besides, online learning that deals with data sequentially is crucial to process big data sets in real time. In this thesis, we carry out for EDF several empirical studies of energy data sets and we achieve great forecasting performance.

Remerciements

Ma thèse n'est pas le fruit d'un travail solitaire (loin de là) et n'existerait pas sans l'aide de nombreuses personnes. Il me paraît donc indispensable de les remercier dans ces quelques lignes.

Mes premières pensées vont à mon directeur de thèse Gilles Stoltz et à Yannig Goude, mon encadrant à EDF. Ces quatre années passées sous votre encadrement m'ont considérablement enrichi. Gilles et Yannig, pour le temps que vous m'avez accordé, pour m'avoir proposé un excellent sujet, pour avoir calibré avec justesse le compromis encadrement/liberté durant ma thèse, merci. Vous m'avez proposé des conditions idéales pour réaliser une thèse. J'ai été extrêmement chanceux de vous avoir rencontré. Je ne connais que peu de personnes (voire aucune) dont la qualité de la rédaction, la rigueur mathématique, l'organisation, la beauté des documents LaTeXet l'originalité colorée des chemises égalent celles de Gilles. Je tiens à te remercier Gilles pour avoir (autant que possible) essayé de me transmettre les quatre premières de ces qualités. Yannig, j'espère que tu ne regrettes pas de t'être battu pour sauver mon entretien Menway catastrophique. Je te remercie de m'avoir fait confiance, pour m'avoir donné l'opportunité de travailler sur de nombreux jeux de données, pour tes idées et ta motivation. Tu as apporté un point de vue pratique et industriel essentiel à cette thèse.

Je suis très reconnaissant à Peter Grünwald et Éric Moulines pour avoir pris le temps de rapporter ma thèse. Je suis vraiment honoré que vous ayez accepté et accompli le travail avec beaucoup d'attention. Je remercie également Georges Hébrail, Pascal Massart, Anne Philippe et Olivier Wintenberger d'avoir accepté de faire partie de mon jury de thèse. Olivier, merci de me faire confiance en m'accueillant prochainement à Copenhague. Je suis sûr que notre collaboration sera fructueuse.

Je remercie vivement mes différents co-auteurs auxquels ce manuscrit doit énormément : Côme, Gábor, Nicolò, Sébastien et Tim. En particulier, je souhaite remercier profondément mon grand frère de thèse, Sébastien, pour son enthousiasme et pour nos passionnantes discussions. J'espère que nous aurons encore de nombreuses opportunités de travailler ensemble.

J'ai vécu cette aventure au sein de trois lieux de travail différents. Avec mon sens de l'organisation peu développé, il n'a pas toujours été facile de m'y retrouver et de bien planifier mes journées. Mais j'ai beaucoup apprécié de découvrir plusieurs équipes aux ambiances aussi différentes qu'agréables. Je remercie tous les membres du laboratoire d'Orsay, de l'équipe de prévision du département

Osiris d'EDF R&D, du DMA. En particulier, merci au personnel de secrétariat (Nathalie, Audrey, Valérie, Zaïna, Benedicte,...) pour votre patience, votre efficacité et votre gentillesse au cours des différentes formalités administratives. Merci à tous les doctorants et aux nombreuses personnes qui, au sein de discussions, de pauses café, ou de pique-niques ensoleillés, ont animé ma vie quotidienne au travail. Un grand merci à mes différents co-bureaux : Émilien pour les parties d'échecs ou de tennis (jamais finies...), Vincent pour m'avoir accompagné dans le froid du bureau B314 et Paul pour son enthousiasme sans limites. Un merci spécifique à Audrey, Amandine, Virgine, et Raphaël pour avoir mis un peu de piment dans la compétition GEFCom. Merci également à toutes les personnes qui ont guidé mes premiers pas en enseignement au cours de mon monitorat. Je pense entre autres à Vincent pour, en plus d'avoir si bien préparé les feuilles de TD, m'avoir dépanné quand j'étais bloqué à Venise à cause de la neige...

Si j'en suis arrivé ici, c'est également grâce à toutes les personnes que j'ai croisées lors de mon parcours scolaire puis universitaire. En vrac, merci à mes anciens professeurs de mathématiques pour m'y avoir donné goût, merci à Sylvain Arlot pour m'avoir si bien orienté à l'ENS et merci aux différentes rencontres au passage de séminaires ou de conférences. Je remercie également Shie Mannor pour son accueil chaleureux à Haifa juste avant ma thèse. Ces six mois en Israël ont été une formidable expérience scientifique mais aussi culturelle.

La thèse a également tendance à déborder dans la vie personnelle. Un grand merci à tout mon entourage qui m'a permis d'avoir une vie riche à côté. Merci donc à mes colocs, Clarus et Pablo, pour les moments partagés au 34 rue Maurice-Arnoux. Merci également à mes amis d'école (Ulminfo, C6, ...), ou plus anciens pour les soirées bière, cinéma, coinche, ou jeux de société. Je pense également à toute ma famille, mes parents, mes soeurs, qui m'ont toujours soutenu et accompagné. Un clin d'oeil à Raphaël qui a égayé la fin de ma thèse. Un grand merci à Pépé et Mamie pour l'excellent pineau qui accompagne mon pot de thèse ! Enfin, Élodie, je te suis infiniment reconnaissant. Tu as été à mes côtés tout au long de la thèse pour me soutenir quand j'en avais besoin. Je te remercie du fond du coeur pour ta patience, ta générosité et pour tous les merveilleux instants passés ensemble.

Contents

Publications and main activities

I list below by type the different activities that were performed during my Ph.D. thesis.

Article in an international journal

[1] M. Devaine, P. Gaillard, Y. Goude, and G. Stoltz. Forecasting electricity consumption by aggregating specialized experts – a review of the sequential aggregation of specialized experts, with an application to Slovakian and French country-wide one-day-ahead (half-)hourly predictions. *Machine Learning*, 90(2):231–260, 2013.

Book chapters

[2] P. Gaillard and Y. Goude. Forecasting the electricity consumption by aggregating experts; how to design a good set of experts. In A. Antoniadis, X. Brossat, and J-M. Poggi, editors, *Modeling and Stochastic Learning for Forecasting in High Dimensions*, volume 217 of *Lecture Notes in Statistics*, pages 95–115. Springer, 2015.

Articles in proceedings of selective international conferences

[3] N. Cesa-Bianchi, P. Gaillard, G. Lugosi, and G. Stoltz. Mirror descent meets fixed share (and feels no regret). In *Proceedings of NIPS'12*, pages 989–997, 2012.

[4] P. Gaillard and S. Gerchinovitz. A chaining algorithm for online nonparametric regression. In *Proceedings of COLT'15*. JMLR: Workshop and Conference Proceedings, 2015. To appear.

[5] P. Gaillard, G. Stoltz, and T. van Erven. A second-order bound with excess losses. In *Proceedings of COLT'14*, volume 35, pages 176–196. JMLR: Workshop and Conference Proceedings, 2014. Finalist best student paper.

Submitted articles

[6] P. Gaillard and P. Baudin. A consistent deterministic regression tree for non-parametric prediction of time series. http://arxiv.org/abs/1405.1533. Submitted, 2015.

[7] P. Gaillard, Y. Goude, and R. Nedellec. Semi-parametric models for GEFCom2014 probabilistic electric load and electricity price forecasting. Invited paper to *International Journal of Forecasting* (Special Issue on Probabilistic Energy Forecasting), 2015.

Miscellaneous

[8] M. Faure, P. Gaillard, B. Gaujal, and V. Perchet. Online learning and game theory. A quick overview with recent results and applications. In A. Garivier et al., editor, *ESAIM: Proceedings*. EDP Sciences, 2015. Submitted.

Technical reports for EDF R&D

[9] P. Gaillard. Aggregate disaggregate predictions. Technical report, EDF R&D, 2015.

[10] P. Gaillard. How to handle uncertainties in a deterministic world. Technical report, EDF R&D, 2015.

[11] P. Gaillard, Y. Goude, and C. Bissuel. Heat load forecasting by aggregating experts. Technical report, EDF R&D, 2015.

Softwares

[12] P. Gaillard. `opera`: Online Prediction by ExpeRts Aggregation. `R`-package for online robust aggregation. To be submitted, 2015.

Competitions

- First price in the electricity price forecasting track of the Global Energy Forecasting Competition 2014 (GEFCom2014), August – December 2014. See [7] and Chapter 8 for details.
- First price in the electric load forecasting track of the Global Energy Forecasting Competition 2014 (GEFCom2014), August – December 2014. See [7] and Chapter 8 for details.
- Participation in the Allstate Purchase Prediction Challenge hosted by Kaggle, May 2014.

Time line of the (pre-)Ph.D. works

Hereafter I trace back in chronological order the main periods of this work.

April – September 2011: M.S.c. internship
Yielded the empirical paper [3] (not included in the present document).

October – December 2011: Predoctoral internship
Yielded the theoretical article [3] which corresponds to Chapter 3.

January – June 2012: Predoctoral internship
Supervised by S. Mannor, at Technion University, Haifa, Israel.
Aimed at clustering and forecasting the electricity consumption of individual Israeli customers. I also worked on how to calibrate the learning rate of the exponentially weighted average forecaster.

September 2012 – July 2015: Ph.D. Program
The `R`-package `opera` [12] was developed and improved throughout the Ph.D.

- 2012 Work on handling uncertainties in robust aggregation: technical report [10] / Chapter 7
- 2013 Empirical work on designing experts: book chapter [2] / Chapter 4
  Theoretical work on second-order bounds: article [5] / Chapter 2
- 2014 Empirical studies on several data-sets: technical reports [9, 11] / Chapters 9 and 10
  Theoretical work on nonparametric prediction: article [6] / Chapter 6
  Machine learning competitions: article [7] / Chapter 8
- 2015 Theoretical work on nonparametric prediction: article [4] / Chapter 5

1

Introduction et vue d'ensemble des résultats

Ce chapitre introductif présente les principales contributions de cette thèse. Après une brève exposition des enjeux industriels en lien avec EDF, nous introduisons le cadre théorique dans lequel nous nous placerons dans la majeure partie de la thèse : la prévision séquentielle de suites arbitraires. Ensuite, nous détaillons les apports tant théoriques qu'appliqués de la thèse. Trois objectifs ont essentiellement guidé les travaux : améliorer la qualité des prévisions, assortir les prévisions d'une mesure d'incertitude et montrer l'universalité de la méthodologie en l'appliquant à divers jeux de données réelles d'EDF.

Sommaire

1.1. Motivation pratique : la prévision de la consommation électrique

Commençons par présenter les enjeux industriels qui ont motivé cette thèse à EDF R&D.

1.1.1. Contexte et enjeux industriels

La prévision de séries temporelles (comme la demande d'électricité) représente un enjeu majeur pour le groupe EDF. En effet, l'électricité ne se stockant pas, l'équilibre entre la production et la demande d'électricité doit être maintenu à tout moment afin d'éviter les risques physiques (coupures d'électricité, reconfiguration de réseaux, black-out) ou financiers. EDF doit donc bénéficier de bonnes prévisions à différents horizons de temps (court terme, moyen terme) afin d'optimiser et de planifier son parc de production (centrales nucléaires, thermiques, éoliennes,...).

Ces dernières années, en raison de l'évolution du paysage électrique français (ouverture du marché à la concurrence, modifications des usages de consommation), le groupe s'est intéressé à la création de méthodes plus réactives au changement remettant en question l'hypothèse de stationnarité du signal (voir Goude [80], Dordonnat [61]). EDF R&D a ainsi développé de nombreuses méthodes de prévision performantes (voir par exemple [18, 50, 105, 121]).

Dans cette thèse, on se propose d'agréger ces méthodes de façon robuste et adaptative. Cette thèse se place dans le prolongement de celle de Goude [80] qui présentait des résultats encourageants sur l'agrégation séquentielle de prédicteurs.

Un premier objectif est d'améliorer la performance des prévisions non seulement en développant de nouveaux algorithmes d'agrégation mais en travaillant aussi sur le choix des méthodes de base à inclure dans le mélange.

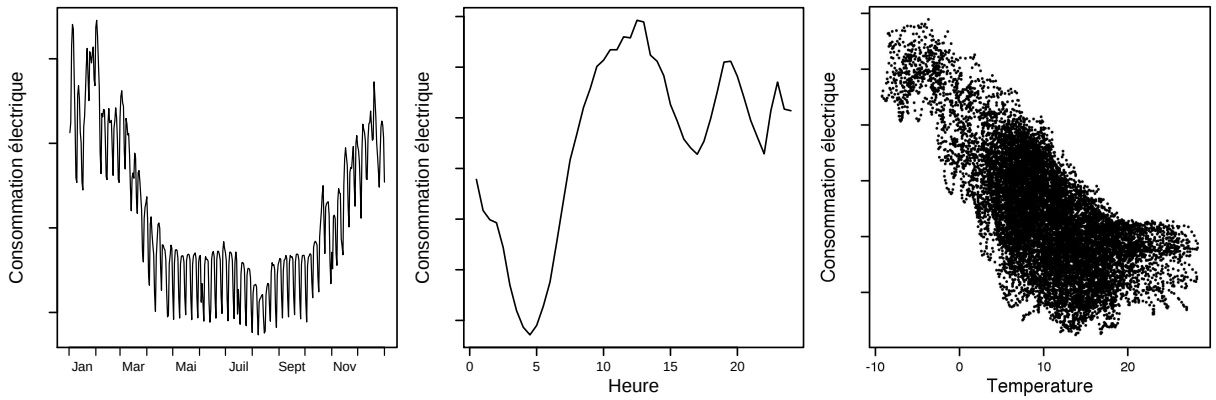
Un deuxième objectif est d'assortir les prévisions d'une mesure d'incertitude, voire de prévoir des lois. La prévision en loi est un problème d'actualité à l'égard d'EDF R&D en particulier avec le développement des énergies renouvelables. La production d'électricité dépend de conditions météorologiques non contrôlables (comme le vent pour les éoliennes) et devient elle-même une quantité aléatoire à prévoir. Dans de nombreuses situations pratiques, nous sommes alors plus intéressés par le risque qu'une situation extrême se produise (comme par exemple une demande d'électricité exceptionnellement forte) plutôt que par la prévision ponctuelle de la consommation moyenne. Les cas difficiles où cela ne se passe pas comme prévu sont importants car ils induisent de fortes contraintes au réseau ou des pertes financières.

1.1.2. Jeux de données

Dans cette thèse, on s'intéresse à la prévision à court terme de la demande d'électricité, de la demande de chaleur et du prix de l'électricité. Un des objectifs est de proposer des méthodes de prévision automatiques (pas de paramètre à calibrer) et universelles (qui fonctionnent sur toutes sortes de données).

La consommation électrique La prévision de consommation électrique a été l'application principale tout au long de la thèse (cf. chapitres 2, 4, 8, 9, et 10). Les jeux de données utilisés diffèrent légèrement entre les chapitres (dates différentes, signaux français ou américains, différentes mailles

géographiques,...) mais la consommation électrique agrégée (i.e., sommée sur suffisamment de consommateurs) présente globalement toujours les mêmes caractéristiques illustrées en figure 1.1. Pour résumer, la demande électrique présente une tendance et trois cycles temporels qui sont liés à



(a) Moyennes journalières de la consommation

(b) Moyennes horaires de la consommation

(c) Dépendance à la température

FIGURE 1.1. : Caractéristiques de la consommation électrique en France sur l'année 2011.

l'activité humaine : au niveau annuel (demande plus forte hiver), au niveau hebdomadaire (demande plus faible les week-ends), et au niveau journalier (demande plus faible la nuit). La consommation dépend aussi de variables exogènes comme les tarifs, ou des variables météorologiques comme la température (utilisation du chauffage pour les températures basses), la couverture nuageuse, ou la vitesse du vent.

La demande de chaleur Étudiée au chapitre 9, sa prévision est motivée par les centrales de cogénération qui produisent à la fois de l'électricité mais aussi de l'eau chaude pour chauffer la ville voisine. Le principal objectif est la production d'eau chaude, celle d'électricité n'étant que secondaire. Deux différences principales avec la demande électrique française sont à noter. Premièrement, le signal est ici d'une amplitude beaucoup plus faible que ceux étudiés pour la consommation électrique et est donc moins stable. Deuxièmement, la demande de chaleur présente des régimes extrêmement différents en été (pas de corrélation à la température) et en hiver, qui sont difficilement capturés par un seul modèle.

Le prix de l'électricité Ce signal est beaucoup plus volatil que les deux précédents. En cas de tension sur le réseau électrique (production trop faible par rapport à la demande), le prix peut, temporairement et de façon peu prévisible, exploser. Nous en étudions la prévision probabiliste dans le chapitre 8.

1.1.3. Méthodes de prévision à EDF

La prévision de séries temporelles est une activité essentielle au sein d'EDF R&D qui a développé de nombreuses méthodes de prévision ces dernières décennies. Avec le temps, le nombre de modèles compétitifs s'est donc naturellement accru en particulier dans le domaine de la prévision de la consommation électrique. Certains modèles occupent une place importante dans cette thèse : GAM (Generalized Additive Models), KWF (Kernelized Wavelet Regression), et CLR (Curve Linear Regression). Nous les décrivons très brièvement ci-après.

GAM Les modèles additifs généralisés ont été introduits par Hastie et Tibshirani [87]. Ils expliquent la consommation comme une somme d'effets non linéaires des variables exogènes. Ces modèles présentent d'excellentes performances pour la prévision de la consommation électrique (cf. Pierrot et Goude [121]). Ils commencent à être utilisés par les opérateurs ayant en charge la prévision d'électricité et font l'objet d'une grande attention au sein du département Osiris d'EDF R&D avec notamment la thèse en cours de Vincent Thouvenot (2012).

KWF Le deuxième modèle de prévision, KWF est un processus autorégressif fonctionnel. C'est un modèle non paramétrique qui s'appuie sur les similarités de la consommation récente avec le passé dans une base d'ondelettes. Ce modèle a été introduit par Antoniadis et al. [16] et approfondi par Cugliari [55]. Une de ses particularités est de ne pas utiliser de variable exogène (comme la température) contrairement aux autres modèles. La qualité de ses prévisions n'est donc pas dépendante de celle des prévisions météorologiques.

CLR Le troisième modèle, CLR, introduit et appliqué à la consommation électrique par Cho et al. [50, 51], est aussi un modèle autorégressif de processus fonctionnels. Il considère la série temporelle comme une courbe et effectue une réduction de la dimension suivie d'une transformation des données pour se ramener à un problème de régression linéaire.

D'autres modèles existent et sont utilisés à EDF, comme le modèle historique paramétrique utilisé par les opérationnels (le modèle Météore implémenté dans le logiciel Éventail [39]). Au cours de ma thèse, j'ai également exploré des modèles issus de l'apprentissage statistique comme les forêts aléatoires ou le boosting. Les forêts aléatoires, introduites par Breiman [35], sont une méthode d'ensemble qui construit un grand nombre d'arbres de régression aléatoires avant de les moyenner (voir également le travail de Deswartes [59] sur la consommation électrique). Le boosting (cf. Friedman [72]) est une autre méthode d'ensemble. Les prédicteurs de base sont des méthodes de régression très faibles (par exemple des arbres de régression très peu profonds). Ceux-ci sont construits séquentiellement dans l'objectif de réduire le biais du prédicteur moyenné.

1.2. Introduction à la prévision de suites arbitraires

Motivés par la diversité des méthodes de prévision développées à EDF R&D ces dernières décennies, nous avons cherché à utiliser un cadre théorique robuste qui puisse réunir des modèles statistiques reposant sur des hypothèses très différentes. C'est le cadre des suites individuelles (ou arbitraires) présenté maintenant et qui a la particularité de ne faire aucune hypothèse stochastique sur la suite à prévoir.

1.2.1. Le cadre des suites arbitraires

Nous cherchons à prévoir de façon séquentielle une suite d'observations $y_1, \dots, y_T$ à valeurs dans un espace $\mathcal{Y}$. Dans l'approche statistique classique, des hypothèses de régularité sont faites. Typiquement, on suppose l'existence d'un processus stochastique stationnaire sous-jacent. L'hypothèse la plus courante est que les observations sont les réalisations de variables aléatoires indépendantes et identiquement distribuées.

Nous nous plaçons ici dans un cadre beaucoup plus robuste : nous ne faisons aucune hypothèse stochastique. La suite est dite arbitraire. Quand l'approche statistique classique demande des garanties

de performance en espérance ou avec grande probabilité, l'objectif ici est qu'elles soient valables uniformément sur toutes les suites d'observations possibles. C'est le cadre des suites individuelles qui connaît un large succès ces dix dernières années en raison de ses garanties extrêmement robustes. Une excellente (et approfondie) introduction est proposée par le livre de Cesa-Bianchi et Lugosi [43].

Critère de performance Considérons $\mathcal{X}$ un ensemble convexe de prévision (typiquement $\mathcal{X}$ est un intervalle borné de $\mathbb{R}$). Pour évaluer la performance d'une prévision $\widehat{y}_t \in \mathcal{X}$, produite par le statisticien au temps t, nous considérons une fonction de perte $\ell : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}_+$ qui est supposée convexe en son premier argument. À l'instant t, $\ell(\widehat{y}_t, y_t)$ mesure l'écart entre la prévision $\widehat{y}_t$ et l'observation y_t. L'objectif du statisticien est alors de minimiser sa perte cumulée définie par

$$\widehat{L}_T \stackrel{\text{def}}{=} \sum_{t=1}^{T} \ell(\widehat{y}_t, y_t)\,. \tag{1.1}$$

Exemple 1.1. Un exemple de perte ℓ convexe est donné par la perte carrée $(x, y) \mapsto (x - y)^2$ pour $\mathcal{X} = \mathcal{Y} \subset \mathbb{R}$. D'autres pertes utilisées dans ce manuscrit sont la perte absolue $(x, y) \mapsto |x - y|$ ou le pourcentage d'erreur absolue $(x, y) \mapsto |x - y|/y$. Pour $\alpha \in (0, 1)$, la perte quantile de Koenker et Bassett [101] définie par $(x, y) \mapsto (y - x)(\alpha - \mathbf{1}_{\{y<x\}})$ joue également un rôle essentiel dans la prévision probabiliste (cf. chapitre 7). D'autres pertes plus opérationnelles pourraient aussi être envisagées pour refléter directement les pertes financières ou les risques de black-out.

Des experts pour nous aider Bien sûr, sans aide ou hypothèse supplémentaire, il est illusoire de réussir à assurer de bonnes prévisions uniformément sur toutes les suites d'observations possibles. Heureusement, n'oublions pas notre point de départ, nous disposons d'un ensemble de méthodes de prévision (développées par EDF R&D dans notre cas et appelées experts dans la suite) qui nous proposent leurs propres prévisions. La performance du statisticien est alors évaluée relativement à celle des experts. Ce cadre est celui de la prévision séquentielle avec avis d'experts. Nous le résumons en figure 1.2.

Notations : $T \geqslant 1$, nombre d'observations ; $K \geqslant 1$, nombre d'experts
À chaque instant $t = 1, \ldots, T$

1. Des experts indexés par $k = 1, \ldots, K$ fournissent chacun une prévision $x_{k,t} \in \mathcal{X}$;
2. Le statisticien prévoit $\widehat{y}_t \in \mathcal{X}$ fondé sur
   - les observations passées $y_1, \ldots, y_{t-1} \in \mathcal{Y}$
   - les prévisions présentes et passées des experts $x_{k,s}$ pour $1 \leqslant s \leqslant t$ et $1 \leqslant k \leqslant K$.
3. Le statisticien observe $y_t \in \mathcal{Y}$
4. Le statisticien subit la perte $\widehat{\ell}_t \stackrel{\text{def}}{=} \ell(\widehat{y}_t, y_t)$ et les experts les pertes $\ell_{k,t} \stackrel{\text{def}}{=} \ell(x_{k,t}, y_t)$.

FIGURE 1.2. : Prévision séquentielle avec avis d'experts

Généralement, la prévision $\widehat{y}_t$ formée par le statisticien à l'instant $t \geqslant 1$ est un mélange des prévisions des experts $x_{k,t}$ pour $1 \leqslant k \leqslant K$, où les poids sont calculés à l'aide des performances passées

des experts.

Un compromis entre deux erreurs La perte du statisticien est décomposée en deux termes

$$\underbrace{\sum_{t=1}^{T} \widehat{\ell}_t}_{\stackrel{\text{def}}{=} \widehat{L}_T} \stackrel{\text{def}}{=} \underbrace{\min_{k=1,\dots,K} \sum_{t=1}^{T} \ell_{k,t}}_{\stackrel{\text{def}}{=} L_T^\star} + \underbrace{\sum_{t=1}^{T} \widehat{\ell}_t - \min_{k=1,\dots,K} \sum_{t=1}^{T} \ell_{k,t}}_{\stackrel{\text{def}}{=} \mathrm{Reg}_T} . \tag{1.2}$$

performance du statisticien — performance de référence (erreur d'approximation) — regret cumulé (erreur d'estimation)

Le premier terme de droite est la performance de référence ou l'erreur d'approximation. L'objectif du statisticien va être de se rapprocher de cette performance sur le long terme. Le plus fréquemment l'erreur de référence est celle du meilleur expert (comme présenté en équation (1.2)). Cependant, en élargissant la classe des stratégies de référence, nous allons voir comment la diminuer sensiblement (voir paragraphe 1.3.1).

Le second terme de droite est le regret cumulé (parfois simplement appelé regret dans la suite). Il représente la difficulté rencontrée par ce dernier à ne pas connaître à l'avance le meilleur expert et à devoir l'estimer séquentiellement. Lorsque les pertes instantanées sont bornées, le regret cumulé augmente au plus linéairement avec le temps. L'objectif du statisticien est qu'il soit sous-linéaire uniformément sur l'ensemble des suites d'observations et de prévisions d'experts possibles. Autrement dit, on veut que, quoi qu'il arrive, le statisticien donne en moyenne des prévisions presque aussi performantes que le meilleur des experts ; ce qui s'écrit

$$\limsup_{T\to\infty} \left(\sup_{(y_t)_t,(x_{k,t})_{k,t}} \left\{ \frac{1}{T} \sum_{t=1}^{T} \widehat{\ell}_t - \min_{k=1,\dots,K} \frac{1}{T} \sum_{t=1}^{T} \ell_{k,t} \right\} \right) \leqslant 0 .$$

Il existe des stratégies de prévision, comme l'agrégation par poids exponentiels détaillée ci-après (cf. Équation (1.3)), qui garantissent un regret Reg_T de l'ordre de $\mathcal{O}\big(\sqrt{T \log K}\big)$ et donc un regret moyen $\mathrm{Reg}_T / T = \mathcal{O}\big(\sqrt{(\log K)/T}\big)$. On peut montrer que cette borne est optimale, dans le sens où aucun algorithme ne peut obtenir de meilleure vitesse uniformément sur toutes les suites d'observations et de prévisions d'experts et ce pour toute fonction de perte ℓ bornée et convexe en son premier argument.

Peut-on faire mieux ? Cela ne signifie cependant pas que cette vitesse n'est jamais améliorable. Par exemple, si la fonction de perte est exp-concave*, on peut obtenir un regret cumulé Reg_T constant $\mathcal{O}\big(\log K\big)$. C'est par exemple le cas de la perte carrée quand les observations sont bornées.

Un autre axe d'amélioration consiste à développer des stratégies qui jouissent de bonnes garanties dans le pire des cas mais réussissent aussi à tirer profit de la simplicité de la suite d'observations jouées par l'environnement. Par exemple, un regret de l'ordre $\mathcal{O}\big(\sqrt{L_T^\star \ln K}\big)$, où $L_T^\star$ est la performance du meilleur expert (cf. équation (1.2)), est aussi possible et est meilleur que $\mathcal{O}(\sqrt{T \ln K})$ si au moins un expert est bon. Cette amélioration est connue sous le nom de « improvement for small losses » (voir par exemple Cesa-Bianchi et Lugosi [43] pour plus de détails). De telles stratégies qui réussissent à s'adapter à la difficulté de l'environnement ambiant pour améliorer leurs bornes

* Une fonction de perte $\ell : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ est dite exp-convave si il existe $\eta > 0$ tel que $x \mapsto \exp\big(-\eta \ell(x, y)\big)$ est concave pour tout $y \in \mathcal{Y}$.

de regret sans connaître à l'avance la suite d'observations sont appelées adaptatives. Une façon d'en obtenir consiste à prouver des bornes de regret dépendant des observations. C'est ce que nous faisons au chapitre 2.

1.2.2. La stratégie la plus répandue : le mélange par poids exponentiels

Généralement le statisticien produit ses prévisions en agrégeant les prévisions des experts de façon convexe. À chaque instant $t \geqslant 1$, il forme un vecteur de poids $\widehat{\boldsymbol{p}}_t = (\widehat{p}_{1,t}, \dots, \widehat{p}_{K,t})$ dans le simplexe $\Delta_K = \{\boldsymbol{p} \in \mathbb{R}_+^K : \sum_{k=1}^K p_k = 1\}$ et prévoit la moyenne pondérée $\widehat{y}_t = \sum_{k=1}^K \widehat{p}_{k,t} x_{k,t}$. C'est le cadre de l'agrégation robuste séquentielle d'experts. La stratégie la plus connue est l'agrégation par poids exponentiels (EWA[†]) introduite en apprentissage séquentiel par Littlestone et Warmuth [107] et Vovk [139]. Elle attribue les pondérations

$$\widehat{p}_{k,t} \stackrel{\text{def}}{=} \frac{\exp\left(-\eta \sum_{s=1}^{t-1} \ell(x_{k,s}, y_s)\right)}{\sum_{j=1}^K \exp\left(-\eta \sum_{s=1}^{t-1} \ell(x_{j,s}, y_s)\right)}, \tag{1.3}$$

où $\eta > 0$ est un paramètre d'apprentissage à calibrer. Si la fonction de perte ℓ est convexe en son premier argument et bornée à valeurs dans $[a, b]$, on peut montrer qu'EWA a un regret cumulé uniformément borné par

$$\mathrm{Reg}_T \leqslant \frac{\log K}{\eta} + \frac{\eta T (b-a)^2}{8}. \tag{1.4}$$

Pour le choix optimal du paramètre d'apprentissage $\eta = (b-a)^{-1}\sqrt{8(\log K)/T}$, on obtient la borne supérieure $(b-a)\sqrt{(\log K)T/2}$ pour le regret cumulé. La preuve est par exemple disponible au chapitre 2.2 de Cesa-Bianchi et Lugosi [43].

Calibration du paramètre d'apprentissage Le statisticien ne connaît pas forcément les valeurs de a, b, et T à l'avance. Il doit calibrer le paramètre η de façon séquentielle. Dans ce but, plusieurs solutions sont possibles au coût d'un facteur multiplicatif constant. La plus simple est connue sous le nom de « doubling trick ». Pour la calibration de T par exemple, l'idée consiste à réinitialiser l'algorithme avec $\eta = (b-a)^{-1}\sqrt{8(\log K)/t}$ chaque fois que le temps t est une puissance de 2 (en oubliant toute l'information gagnée dans le passé). La borne de regret n'est alors détériorée que d'un facteur $\sqrt{2}/(\sqrt{2}-1) \approx 3.41$ (cf. Cesa-Bianchi et Lugosi [43, chapitre 2.3]). De façon plus satisfaisante, on peut aussi redéfinir η_t à chaque instant en fonction des données passées sans avoir à réinitialiser l'algorithme (cf. [43]).

La calibration du paramètre d'apprentissage joue un rôle essentiel dans l'obtention d'algorithmes adaptatifs mentionnés plus tôt : si la suite d'observations est régulière, de grandes valeurs de η permettent parfois une convergence beaucoup plus rapide. Les calibrations théoriques dans le pire des cas ne sont souvent que peu efficaces et des calibrations plus judicieuses utilisant l'information passée (voir chapitre 2) ou plus pratiques, comme celle proposée par Devaine et al. [60], sont nécessaires.

[†]Exponentially Weighted Average

1.3. Amélioration de la qualité des prévisions

Une grande partie de cette thèse se concentre sur un objectif de performance pure : comment diminuer l'erreur (1.1) ? Comme nous l'avons remarqué dans l'équation (1.2), celle-ci se décompose en deux termes : l'erreur d'approximation et le regret que nous cherchons à diminuer.

Remarquez que ces deux termes d'erreurs sont en compétition : il s'agit de trouver le bon compromis entre les deux. Typiquement, augmenter le nombre d'experts détériore légèrement les garanties sur le regret. Ce n'est donc intéressant que si cela permet de diminuer suffisamment l'erreur d'approximation. Il s'agit du même phénomène que le compromis biais-variance en statistiques classiques.

1.3.1. Diminution de l'erreur d'approximation

La moyenne uniforme des experts est souvent plus robuste et plus efficace que le meilleur des experts (cf. méthodes d'ensemble [150]). Il pourrait donc être judicieux de l'ajouter à notre ensemble d'experts au faible coût de remplacer $\log K$ par $\log(K+1)$ dans la borne de regret (1.3). Plus généralement, l'erreur d'approximation (ou performance de référence) peut-être grandement réduite en essayant de se comparer à des ensembles de stratégies plus importants que les K experts de base.

Plus formellement, on considère un ensemble de stratégies possibles indexées par un ensemble Θ. Chaque stratégie nous propose à chaque instant $t \geqslant 1$ une prévision $f_{\theta,t}$ reposant uniquement sur les observations passées y_s pour $1 \leqslant s \leqslant t-1$ et les prévisions passées et présentes des experts $x_{k,s}$ pour $1 \leqslant k \leqslant K$ et $1 \leqslant s \leqslant t$. L'objectif est de développer des algorithmes atteignant sur le long terme la performance de référence

$$L_T(\Theta) \stackrel{\text{def}}{=} \inf_{\theta \in \Theta} \quad \sum_{t=1}^{T} \ell(f_{\theta,t}, y_t)\,. \tag{1.5}$$

Remarque 1.2. On fera attention à ne pas confondre la prévision du statisticien $\widehat{y}_t \in \mathcal{X}$ sur laquelle ce dernier est évalué, des prévisions des stratégies de référence $f_{\theta,t}$ auxquelles le statisticien est comparé. Les $f_{\theta,t}$ jouent le même rôle que les experts $x_{k,t}$ précédemment. En particulier, dans le cas $\Theta = \{1,\dots,K\}$ et pour les prévisions de référence $f_{\theta,t} = x_{\theta,t}$ pour $\theta \in \Theta$ nous retrouvons le cadre précédent de la prévision avec avis d'experts dans lequel la performance de référence est celle du meilleur expert.

Si l'ensemble de stratégies de référence Θ est judicieusement choisi, l'erreur d'approximation (1.5) peut-être sensiblement inférieure à celle de de la définition (1.2) (pour laquelle $\Theta = \{1,\dots,K\}$ et $f_{\theta,t} = x_{\theta,t}$) sans que cela ne soit trop coûteux au niveau du regret.

Si l'ensemble Θ est fini de faible cardinal $|\Theta|$, nous pouvons simplement utiliser par exemple l'algorithme de mélange par poids exponentiels (EWA) sur les experts $f_{\theta,t}$ au lieu de $x_{k,t}$ menant à un regret $\mathcal{O}\big(\sqrt{T \log |\Theta|}\big)$. Cependant, si Θ est très grand, voire même infini (non paramétrique par exemple), il est nécessaire de tirer profit de la structure des stratégies $f_{\theta,t}$ pour améliorer la complexité algorithmique et éventuellement les garanties théoriques.

Nous listons maintenant différents ensembles de stratégies de référence possibles. Chacun correspond à une performance de référence (celle atteinte par la meilleure stratégie définie en (1.5)) avec laquelle les algorithmes de mélange peuvent être comparés. Dans la suite, nous supposons $\mathcal{X} \subset \mathbb{R}^d$.

1.3.1.a. Meilleure combinaison convexe

Un premier ensemble de stratégies de référence judicieux est l'ensemble de toutes les combinaisons convexes. Il correspond au simplexe $\Theta = \left\{\boldsymbol{p} \in \mathbb{R}_+^K : \sum_{k=1}^K p_k = 1\right\}$ et aux prévisions de référence

$$f_{\boldsymbol{p},t} = \sum_{k=1}^K p_k x_{k,t} \qquad \text{pour tout } \boldsymbol{p} \in \Theta \text{ et tout } t = 1, \dots, T\,.$$

Se rapprocher sur le long terme de $L_T(\Theta)$ semble être un objectif bien moins simple que de se comparer seulement à la performance du meilleur expert. Le nombre de stratégies de référence dans Θ est en effet ici infini. Cependant, en utilisant le fait que les stratégies de Θ sont très corrélées (elles ne dépendent que de $K-1$ paramètres réels), on peut montrer que si la fonction de perte ℓ est convexe et sous-différentiable en son premier argument, ce n'est pas plus difficile. Ainsi, le « gradient trick » permet dans ce cas de construire un algorithme approchant la meilleure combinaison convexe à partir d'un algorithme d'agrégation, noté $\mathcal{A}$, n'approchant que le meilleur expert (cf. Cesa-Bianchi et Lugosi [43, chapitre 2.5]). Pour cela, il suffit de remplacer dans $\mathcal{A}$ les pertes $\ell_{k,s} = \ell(x_{k,s}, y_s)$ subies par les experts par les pseudo-pertes $\widetilde{\ell}_{k,s} = \partial\ell(\widehat{y}_s, y_s) \cdot x_{k,s}$ où $\partial\ell$ est un (sous-)gradient de ℓ en son premier argument et $\cdot$ est le produit scalaire. Le regret cumulé est du même ordre $\mathcal{O}(\sqrt{T \log K})$.

Par exemple, EWA décrit en (1.3) donne l'algorithme EG qui attribue les poids

$$\widehat{p}_{k,t} = \frac{\exp\left(-\eta \sum_{s=1}^{t-1} \partial\ell(\widehat{y}_s, y_s) \cdot x_{k,s}\right)}{\sum_{j=1}^K \exp\left(-\eta \sum_{s=1}^{t-1} \partial\ell(\widehat{y}_s, y_s) \cdot x_{j,s}\right)}\,.$$

Exemple 1.3. Pour comprendre l'intuition de cette modification de l'algorithme, donnons l'exemple de prévisions réelles ($\mathcal{X} \subset \mathbb{R}$) évaluées par la perte carrée $\ell\colon (x,y) \in \mathbb{R}^2 \mapsto (x-y)^2$. Celle-ci est différentiable et les pseudo-pertes valent $\widetilde{\ell}_{k,s} = 2(\widehat{y}_s - y_s)x_{k,s}$. On remarque alors que cette perte avantage les experts qui rapprochent la prévision du mélange $\widehat{y}_s$ vers l'observation y_s :

- si $\widehat{y}_s > y_s$: les pseudo-pertes sont positives et favorisent les experts ayant prévu de faibles valeurs $x_{k,s}$.
- si $\widehat{y}_s < y_s$: les pseudo-pertes sont négatives et donc plus faibles pour les experts ayant produit de grandes prévisions $x_{k,s}$.

1.3.1.b. Meilleure combinaison linéaire

L'ensemble des combinaisons linéaires correspond à $\Theta \subset \mathbb{R}^K$ et aux prévisions

$$f_{\boldsymbol{u},t} = \sum_{k=1}^K u_k x_{k,t} \qquad \text{pour tout } \boldsymbol{u} \in \Theta \text{ et tout } t = 1, \dots, T\,.$$

Dans le cas des fonctions de pertes ℓ convexes générales, Kivinen et Warmuth [97] ont introduit une méthode générique pour convertir un algorithme d'agrégation convexe, noté $\mathcal{A}$, compétitif face au simplexe $\Theta = \Delta_K$, en un algorithme approchant la meilleure stratégie de $\Theta = \left\{\boldsymbol{u} \in \mathbb{R}^K : \|\boldsymbol{u}\|_1 \leqslant \beta\right\}$ la boule ℓ_1 de rayon $\beta > 0$. Pour cela, il suffit d'utiliser $\mathcal{A}$ en remplaçant les K prévisions des experts

$x_{1,t},\dots,x_{K,t}$ par les $2K$ prévisions $-\beta x_{1,t},\dots,-\beta x_{K,t},\beta x_{1,t},\dots,\beta x_{K,t}$ (cf. Kivinen et Warmuth [97]). Nous utilisons cette réduction au chapitre 8 pour produire des prévisions des quantiles de y_t en utilisant la perte quantile sur des experts prévoyant la moyenne. Nous en déduisons un regret $\mathrm{Reg}_T(\Theta)$ du même ordre de grandeur $\mathcal{O}(\sqrt{T\log K})$ que celui obtenu face au meilleur expert.

Dans le cas particulier de la perte carrée (avec des prévisions réelles), l'algorithme par excellence est la régression ridge (Ridge) introduite en apprentissage séquentiel par Azoury et Warmuth [21] et Vovk [142] qui choisit à l'instant $t \geqslant 1$ le vecteur $\widehat{\boldsymbol{u}}_t \in \mathbb{R}^K$ minimisant l'erreur passée régularisée en norme ℓ_2 :

$$\widehat{\boldsymbol{u}}_t \in \operatorname*{arg\,min}_{\boldsymbol{u}\in\mathbb{R}^K} \left\{ \sum_{s=1}^{t-1} (y_s - \boldsymbol{u}\cdot\boldsymbol{x}_s)^2 + \lambda \|\boldsymbol{u}\|_2^2 \right\},$$

où $\boldsymbol{x}_s = (x_{1,s},\dots,x_{K,s})$ est le vecteur de prévision des experts. Ridge prévoit ensuite $\widehat{y}_t = \sum_{k=1}^K \widehat{u}_{k,t} x_{k,t}$. Ridge a un regret de l'ordre de $\mathcal{O}\big(K\sqrt{T\log T}\big)$ face aux stratégies bornées $\Theta = \big\{\boldsymbol{u}\in\mathbb{R}^K : \|\boldsymbol{u}\|_2 \leqslant \beta\big\}$ pour un paramètre de régularisation λ de l'ordre de $\mathcal{O}(\sqrt{KT\log T})$ (cf. Astolfi et al. [19]). L'ajout d'un terme de régularisation $\big(\boldsymbol{u}\cdot\boldsymbol{x}_s\big)^2$ utilisant les prévisions des experts permet d'obtenir une meilleure vitesse $\mathcal{O}(K\log T)$ (cf. [21, 142]).

Jusqu'à présent, nous nous sommes intéressés à une notion de regret usuelle qui consiste à comparer la perte cumulée du statisticien avec celle du meilleur élément d'un ensemble Θ suffisamment petit pour pouvoir faire appel directement aux algorithmes de mélanges usuels. Les vitesses du regret en $\mathcal{O}(\sqrt{T})$ n'étaient pas modifiées. Nous détaillons maintenant d'autres ensembles de stratégies de référence beaucoup plus grands étudiés dans la suite de ce manuscrit. Ceux-ci nécessitent des algorithmes adaptés pour faire face à la complexité de Θ.

1.3.1.c. Meilleure suite de combinaisons convexes (cf. chapitre 3)

Un objectif beaucoup plus ambitieux consiste par exemple à comparer la performance du statisticien à celle d'une suite arbitraire d'éléments de Θ et non plus à un même élément fixe au court du temps. Ce cadre intéresse particulièrement EDF qui cherche à avoir des méthodes de prévision robustes aux ruptures liées aux changements du marché de l'énergie. Plus formellement, dans le cas des combinaisons convexes, cela revient à considérer toutes les stratégies de référence indexées par $\Theta = \Delta_K^T$ et formant les prévisions

$$f_{\boldsymbol{p}_{1:T},t} = \sum_{k=1}^K p_{k,t} x_{k,t} \qquad \text{pour tout } \boldsymbol{p}_{1:T} = (\boldsymbol{p}_1,\dots,\boldsymbol{p}_T) \in \Theta \text{ et tout } t = 1,\dots,T\,.$$

Malheureusement, cet ensemble de stratégies de référence est trop grand. Il n'est pas réaliste de contrôler le regret uniformément sur toutes les stratégies $\boldsymbol{p}_{1:T} \in \Delta_K^T$. Cela reviendrait en effet à connaître à l'avance à chaque instant le vecteur de mélange à utiliser pour prévoir la prochaine observation. Ce problème est souvent appelé « tracking the best expert ».

État de l'art Plusieurs solutions ont été amenées dans la littérature à faire des hypothèses de régularité ou de parcimonie sur les suites $\boldsymbol{p}_{1:T}$ avec lesquelles nous nous comparons. Des exemples d'algorithmes sont la descente miroir de Herbster et Warmuth [93], l'algorithme fixed share de Herbster et Warmuth [92] et sa généralisation par Bousquet et Warmuth [32].

Par exemple, Herbster et Warmuth [92] garantissent pour fixed share une borne de regret de l'ordre de $\mathcal{O}\Big(Tm_0 \log(KT/m_0)\Big)$, par rapport à l'ensemble $\Theta = \big\{\boldsymbol{p}_{1:T} \in \Delta_K^T : m(\boldsymbol{p}_{1:T}) \leqslant m_0\big\}$. Ici, la régularité des suites est mesurée par la somme cumulée des différences en norme ℓ_1 entre deux vecteurs consécutifs :

$$m(\boldsymbol{p}_{1:T}) \stackrel{\text{def}}{=} \sum_{t=1}^{T} \|\boldsymbol{p}_t - \boldsymbol{p}_{t-1}\|_1 \,, \qquad \text{où } \boldsymbol{p}_0 = (1/K, \dots, 1/K) \,.$$

Bousquet et Warmuth [32] ont proposé un algorithme améliorant la borne de regret quand la suite de poids utilisée $\boldsymbol{p}_{1:t}$ est parcimonieuse, dans le sens où peu de vecteurs de pondération différents sont utilisés (i.e., card $\big\{p_t : t \in \{1, \dots, T\}\big\}$ est faible).

Contributions 1.4. Nous proposons au chapitre 3 une analyse unifiée et simplifiée des algorithmes cités ci-dessus dont nous retrouvons (et parfois améliorons) les garanties. Notre analyse généralise les résultats à un cadre où les poids des vecteurs de comparaison se somment pas nécessairement à 1. Cela nous permet d'unifier plusieurs notions de regret jusqu'ici non reliées dans la littérature, entre autres :

- le *regret escompté* (cf. Cesa-Bianchi et Lugosi [43, chapitre 2.11]) où les pertes des différents instants sont pondérées par un facteur β_t (pas forcément connu à l'avance par le statisticien) ;
- le *regret adaptatif* introduit par Hazan et Seshadhri [90] dans lequel le statisticien n'est évalué qu'entre les instants r et $s \in \{1, \dots, T\}$ qu'il ne connaît pas à l'avance. Autrement dit, il cherche à minimiser
$$\sum_{t=r}^{s} \ell(\widehat{y}_t, y_t) - \min_{k=1,\dots,K} \sum_{t=r}^{s} \ell(x_{k,t}, y_t) \,,$$
simultanément pour tout couple $(r, s) \in \mathbb{N}^2$ tel que $1 \leqslant r \leqslant s \leqslant T$.

1.3.1.d. Meilleure fonction des experts (cf. chapitres 5 et 6)

Un autre objectif consiste à se comparer à un ensemble de stratégies non paramétriques. C'est ce que l'on fait aux chapitres 5 et 6, où Θ est une classe de fonctions de comparaison.

Ici, nous autorisons les prévisions du statisticien à vivre dans un espace différent de celles des experts. Plus précisément, nous supposons que les experts fournissent des prévisions $x_{k,t} \in \mathcal{X}$ quelconques, mais que les prévisions du statisticien doivent être réelles. Chaque stratégie de référence est une fonction $g : \mathcal{X}^K \to \mathbb{R} \in \Theta \subset \mathbb{R}^{(\mathcal{X}^K)}$ et produit les prévisions

$$f_{g,t} = g(\boldsymbol{x}_t) \qquad \text{pour tout } t = 1, \dots, T \,,$$

où $\boldsymbol{x}_t = (x_{1,t}, \dots, x_{K,t}) \in \mathcal{X}^K$ est le vecteur de prévision des experts. L'objectif du statisticien est d'avoir un regret

$$\mathrm{Reg}_T(\Theta) \stackrel{\text{def}}{=} \sum_{t=1}^{T} \ell(\widehat{y}_t, y_t) - \inf_{g \in \Theta} \sum_{t=1}^{T} \ell(g(\boldsymbol{x}_t), y_t) \tag{1.6}$$

le plus faible possible. Dans la suite, nous nous restreignons au cas $K = 1$ (sans perte de généralité), c'est-à-dire $\Theta \subset \mathbb{R}^{\mathcal{X}}$.

L'entropie métrique Bien entendu, plus l'ensemble Θ est grand, plus il est difficile de s'y comparer et cela joue significativement sur l'ordre de grandeur en T des bornes de regret. La taille d'un ensemble peut être mesurée par la notion d'*entropie métrique*.

Definition 1.1. *L'entropie métrique* $\log \mathcal{N}_\infty(\Theta, \varepsilon)$ *est définie par le logarithme du nombre minimal d'éléments de* Θ *nécessaires pour former un* ε*-recouvrement de* Θ *en norme infinie* $\|\cdot\|_\infty$. *Autrement dit,*

$$\log \mathcal{N}_\infty(\Theta, \varepsilon) \stackrel{\text{def}}{=} \min \Big\{ \operatorname{card}(\mathcal{G}) : \mathcal{G} \subset \Theta \quad \textit{tel que} \quad \forall f \in \Theta, \quad \exists g \in \mathcal{G} : \ \|g - f\|_\infty \leqslant \varepsilon \Big\}.$$

Exemple 1.5. Donnons les valeurs de l'entropie métrique de quelques classes de fonctions usuelles qui nous intéresseront dans la suite pour illustrer leurs tailles.

- L'espace des fonctions 1-Lipschitz en dimension $d \geqslant 1$ que l'on note Θ^{lip} a une entropie métrique de l'ordre $\log \mathcal{N}_\infty(\Theta^{\text{lip}}, \varepsilon) \sim \varepsilon^{-d}$ quand $\varepsilon \to 0$, cf. Lorentz [108].
- L'espace $\Theta^{\text{höld}}$ des fonctions donc les $q \in \mathbb{N}$ premières dérivées existent et sont bornées en norme infinie par $B > 0$ et telles que les q-èmes dérivées sont Hölder d'ordre $\alpha \in (0, 1]$. En d'autres termes, pour toute fonction $f \in \Theta^{\text{höld}}$,

$$\forall x, y \in \mathcal{X}, \quad \left| f^{(q)}(x) - f^{(q)}(y) \right| \leqslant |x - y|^\alpha . \tag{1.7}$$

  Par exemple, si $\mathcal{X} = [0, 1]^d$ l'entropie métrique est de l'ordre $\log \mathcal{N}_\infty(\Theta^{\text{höld}}, \varepsilon) \sim \varepsilon^{-d/(q+\alpha)}$ quand $\varepsilon \to 0$, cf. Lorentz [108]. La régularité de la classe $\Theta^{\text{höld}}$ dépend donc du coefficient de régularité $\beta = q + \alpha$ et de la dimension d.

Un algorithme simple ? Si la fonction de perte ℓ est suffisamment régulière (1-Lipschitz en son premier argument) et que l'entropie métrique est finie pour tout $\varepsilon > 0$, une solution naturelle est d'utiliser un algorithme de mélange (comme EWA) sur les centres des boules d'un ε-recouvrement $\Theta^{(\varepsilon)}$ de cardinalité minimale $\mathcal{N}_\infty(\Theta, \varepsilon)$. On obtient une borne supérieure (en omettant les constantes)

$$T\varepsilon + \sqrt{T \log \mathcal{N}_\infty(\Theta, \varepsilon)} \tag{1.8}$$

sur le regret cumulé (1.6). Le premier terme $T\varepsilon$ est l'erreur d'approximation que l'on commet en se restreignant à $\Theta^{(\varepsilon)}$. Le deuxième terme est le regret (voir (1.4)) subi par EWA sur la classe finie d'experts $\Theta^{(\varepsilon)}$. Pour la classe des fonctions Lipschitz Θ^{lip} nous obtenons pour la valeur optimale $\varepsilon \approx T^{-1/(d+2)}$, un regret $\operatorname{Reg}_T\left(\Theta^{(\text{lip})}\right)$ de l'ordre de $\mathcal{O}(T^{(d+1)/(d+2)})$. Pour la classe de fonctions Hölder, $\Theta^{\text{höld}}$ le regret est de l'ordre $\mathcal{O}(T^{(d+\beta)/(d+2\beta)})$. L'inconvénient principal de cet algorithme est sa complexité : il nécessite de garder en mémoire et de mettre à jour à chaque instant un nombre exponentiellement croissant de poids.

Remarque 1.6. Si la fonction de perte ℓ est exp-concave (comme par exemple la perte carrée qui nous intéresse au chapitre 5), EWA peut garantir de meilleures vitesses (cf. paragraphe 1.2.2). Son regret cumulé (1.6) est alors majoré de la même façon par

$$T\varepsilon + \log \mathcal{N}_\infty(\Theta, \varepsilon) . \tag{1.9}$$

Nous en déduisons les regrets pour les classes Lipschitz $\operatorname{Reg}_T\left(\Theta^{(\text{lip})}\right) = \mathcal{O}\left(T^{d/(d+1)}\right)$ et Hölder $\operatorname{Reg}_T\left(\Theta^{\text{höld}}\right) = \mathcal{O}\left(T^{d/(d+2\beta)}\right)$.

Contributions 1.7. L'objectif des chapitres 5 et 6 est de trouver des stratégies algorithmiquement efficaces et plus performantes (meilleures garanties théoriques) que EWA pour les cas particuliers des classes Hölder et Lipschitz. Dans le chapitre 5, nous nous intéressons principalement à la perte carrée. Nous effectuons une analyse multi-échelle qui tire ses idées de techniques de chaînage. Cela nous permet de tirer profit :

- de la vitesse (1.9) à une grande échelle pour effectuer une première approximation grossière face à un ensemble de référence Θ petit ;
- de la vitesse (1.8) à de petites échelles pour faire des corrections plus fines et permettre de se comparer à des ensemble Θ plus important.

Nous obtenons ainsi des vitesses optimales pour le regret plus rapides que celles relatives aux regrets (1.8) et (1.9). Nous proposons le premier algorithme de complexités (en temps et en mémoire) raisonnables (polynomiales en T) pour les classes de fonctions Hölder (et donc Lipschitz). Dans le chapitre 6, nous étudions des fonctions de pertes convexes générales mais dans le cadre des classes de fonctions de référence Lipschitz. Nous proposons un algorithme qui reprend l'idée des arbres de régression CART et divise l'espace des variables exogènes $\mathcal{X}$ en se basant sur les données passées. Les bornes sur le regret obtenues dans le pire des cas ne sont pas meilleures que celles de la borne supérieure (1.8) mais la procédure est algorithmiquement efficace et s'adapte mieux à la complexité inhérente du problème. Pour finir, nous justifions notre objectif initial en montrant que minimiser $\mathrm{Reg}_T\left(\Theta^{\mathrm{lip}}\right)$ permet d'obtenir des stratégies asymptotiquement consistantes dans un cadre stochastique. Nous détaillons maintenant ces contributions.

Le cas de la perte carrée Nous proposons au chapitre 5 un algorithme (Algorithme 9) reposant sur des techniques de chaînage qui vérifie (cf. Théorème 5.3), pour toute classe non paramétrique de fonctions de référence Θ, une borne de regret de la forme

$$\mathrm{Reg}_T(\Theta) \leqslant c_1 B^2\big(1+\log \mathcal{N}_\infty(\Theta,\gamma)\big) + c_2 B\sqrt{T}\int_0^\gamma \sqrt{\log \mathcal{N}_\infty(\Theta,\varepsilon)}\,\mathrm{d}\varepsilon\ , \tag{1.10}$$

pour tout $\gamma \in (B/T, B)$ où B est une borne sur $\max_{1\leqslant t\leqslant T}|y_t|$ et c_1, c_2 sont deux constantes positives. Comme l'ont montré Rakhlin et Sridharan [124], cette borne est optimale dès que l'entropie métrique est du même ordre de grandeur que l'entropie séquentielle (voir Rakhlin et Sridharan [124]), ce qui est le cas pour la plupart des espaces de fonctions usuels incluant les fonctions Lipschitz, Hölder, ou les espaces de Besov.

Exemple 1.8. Si Θ a une entropie métrique $\log \mathcal{N}_\infty(\Theta,\varepsilon) \leqslant \varepsilon^{-p}$ avec $p \in (0,2)$, la borne (1.10) devient $\mathcal{O}\big(T^{p/(p+2)}\big)$ qui améliore la complexité $\mathcal{O}\big(T^{p/(p+1)}\big)$ obtenue par un simple EWA (cf. remarque 1.6). Par exemple, dans le cas Hölder $p = d/\beta$ donne un regret en $\mathcal{O}\big(T^{d/(2\beta+d)}\big)$.

L'apport principal du chapitre 5 est une instanciation efficace de l'Algorithme 9 dans le cas des fonctions Hölder. Il permet pour la première fois d'obtenir, de façon constructive et efficace, la vitesse presque optimale $\mathcal{O}\big(T^{1/(2\beta+1)}(\log T)^{3/2}\big)$ pour le regret (quand $d = 1$).

Les fonctions Lipschitz Dans le chapitre 6, nous nous concentrons sur la classe des fonctions Lipschitz Θ^{lip}. Nous proposons un algorithme (Algorithme 12) qui divise séquentiellement l'espace des variables exogènes $\mathcal{X}$ de façon à arbitrer localement et efficacement entre les erreurs d'approximation et d'estimation. L'idée est d'approcher plus précisément la meilleure fonction $f^\star$ de Θ^{lip} soit dans les régions difficiles (mauvaise approximation) soit dans les régions où l'on dispose de

beaucoup de données (bonne estimation).

L'algorithme maintient pour cela un arbre de régression dans lequel chaque nœud n représente une région $\mathcal{X}(n) \subset \mathcal{X}$. Plus un nœud est profond dans l'arbre, plus la région correspondante est petite. Les feuilles de l'arbre forment à tout moment une partition de l'espace des variables exogènes $\mathcal{X}$. L'arbre approxime $f^\star$ par des fonctions constantes par morceaux (chaque feuille de l'arbre étant une fonction constante). Dès que l'algorithme estime qu'une feuille n de l'arbre approxime trop grossièrement $f^\star$, c'est-à-dire que la région de $\mathcal{X}(n)$ correspondante est trop vaste pour être approximée par une constante, l'arbre est agrandi au niveau de cette feuille. Le nœud n devient un nœud interne et sa région $\mathcal{X}(n)$ est séparée en deux sous-régions, chacune associée à une feuille enfant du nœud n.

Une justification stochastique de l'objectif Nous quittons momentanément le cadre des suites arbitraires. Nous supposons que le statisticien doit prévoir séquentiellement la prochaine observation $Y_t \in [0,1]$ d'un processus ergodique stationnaire $(Y_t)_{t=-\infty,\dots,\infty}$ à partir des observations passées $Y_1, \dots, Y_{t-1}$ seulement. La limite fondamentale suivante a été prouvée par Algoet [15]. Quelle que soit la stratégie du statisticien, presque sûrement,

$$\liminf_{T\to\infty} \left\{ \frac{1}{T} \sum_{t=1}^{T} \ell\big(\widehat{Y}_t, Y_t\big) \right\} \geqslant L^\star \,,$$

où $\widehat{Y}_t$ est la prévision du statisticien à l'instant t et

$$L^\star = \mathbb{E}\left[\inf_{f \in \mathcal{B}^\infty} \mathbb{E}\Big[\ell\big(f(Y_{-\infty}^{-1}), Y_0\big) \big| Y_{-\infty}^{-1} \Big] \right]$$

est la perte minimale en espérance de la prévision de Y_0 par une fonction Borélienne du passé. Ici $\mathcal{B}^\infty$ représente l'ensemble de toutes les fonctions Boréliennes de $[0,1]^\infty$ dans $[0,1]$ et ℓ est une fonction de perte Lipschitz et convexe en son premier argument. De nombreux papiers [25, 24, 85, 83] ont donc construit des stratégies dites *consistantes* qui atteignent la borne inférieure, à savoir des stratégies vérifiant

$$\limsup_{T\to\infty} \left\{ \frac{1}{T} \sum_{t} \ell\big(\widehat{Y}_t, Y_t\big) \right\} \leqslant L^\star \,.$$

Tous ces articles reposent dès le départ sur le caractère stochastique de la suite à prévoir. Dans le chapitre 6, nous montrons que la consistance n'est en réalité qu'un corollaire de nos bornes de regret. En d'autres mots, nous proposons une méthode générique pour obtenir une stratégie consistante à partir d'un algorithme assurant un regret $\mathrm{Reg}\left(\Theta^{\mathrm{lip}}\right)$ sous-linéaire.

1.3.1.e. Création de nouveaux experts (cf. chapitre 4)

D'un point de vue pratique, les experts sont formés par un ensemble de méthodes statistiques de base dont nous souhaitons améliorer la performance en les agrégeant (voir paragraphe 1.1.3). Ces méthodes peuvent reposer sur des hypothèses stochastiques et des modèles très différents, ce qui nous a motivé à adopter le cadre robuste des suites individuelles. Pour diminuer l'erreur d'approximation, nous n'avons pour l'instant pas touché à ces prévisions de base. En pratique, travailler sur le jeu d'experts pour qu'il soit suffisamment riche peut pourtant s'avérer très efficace.

Nous analysons cette approche dans le chapitre 4 et étudions comment créer de nouveaux experts efficacement à partir des méthodes statistiques à disposition. Nous proposons quatre procédures pour créer plus de diversité dans notre jeu d'experts :

- modification aléatoire d'un expert initial par méthode de bagging (cf. Breiman [34]) ;
- ajout séquentiel de nouveaux experts cherchant à corriger les erreurs des experts déjà présents dans le mélange en s'inspirant des méthodes de boosting (cf. Friedman [72]) ;
- spécialisation d'experts à des événements météorologiques particuliers (en utilisant des variables exogènes pour pondérer plus fortement certains instants lors de la phase d'estimation des experts) ;
- variation de l'horizon de prévision des experts et réalisation de recalages court terme des résidus.

Application à la prévision de la consommation électrique Nous appliquons la méthode ci-dessus à un jeu de données couramment employé à EDF pour la prévision court terme de la consommation électrique et qui a représenté une des motivations pratiques principales de cette thèse (cf. chapitres 4, 9, 10). Le jeu de données contient cinq années d'observations à un pas demi-horaire de la consommation globale des clients d'EDF en France ainsi que des variables exogènes qui ont un effet sur la consommation (température, couverture nuageuse, jours fériés, ...).

Comme dans toutes nos études empiriques, nous divisons le jeu de données en deux ensembles : un ensemble d'estimation (ici les quatre premières années) qui est utilisé pour entraîner un certain nombre d'experts ; et un ensemble de validation (ici la dernière année) sur lequel tous les experts produisent des prévisions et sont agrégés séquentiellement par un algorithme de mélange. Les performances finales sont évaluées sur l'ensemble de validation.

Dans le chapitre 4, nous disposons de trois méthodes statistiques de prévision de base pour produire nos experts. À partir de la méthodologie du paragraphe précédent, nous créons un jeu de 133 experts aussi varié que possible. Si nous remarquons une augmentation de l'erreur d'estimation (regret cumulé) subie par le mélange en fonction du nombre d'experts considérés, l'erreur d'approximation est très nettement diminuée. Nous obtenons un gain de performance global de l'ordre de 25% (en RMSE) par rapport au meilleur des 133 experts. Ce gain peut être décomposé en deux parties : environ la moitié est obtenue grâce aux techniques d'agrégation séquentielle, l'autre moitié provenant de la création des nouveaux experts.

1.3.2. Diminution de l'erreur d'estimation (cf. chapitre 2)

Comme nous l'avons précisé plus tôt, afin d'améliorer la performance finale de la prévision, il ne suffit pas de se concentrer seulement sur l'erreur d'approximation. Il faut aussi de construire des algorithmes de mélange efficaces avec une erreur d'estimation (i.e., un regret cumulé) la plus faible possible.

Dans ce but, il ne s'agit plus de se concentrer uniquement sur le regret dans le pire des cas déjà bien étudié et pour lequel de nombreuses stratégies optimales existent. Le pire des cas n'est souvent pas représentatif de la réalité et les bornes peuvent être significativement améliorées dans de nombreux scénarios.

Exemple 1.9. Imaginons par exemple qu'après un certain temps, un des experts, noté $k^\star$, a gagné tellement d'avance sur les autres experts, qu'il est clair qu'il sera le vainqueur à la fin du

jeu (après T instants). Dans ce cas, puisque le regret, que le statisticien cherche à minimiser ici, comparera à la fin la performance du statisticien avec celle de l'expert $k^\star$, il n'est plus nécessaire de se concentrer sur aucun autre expert, et le regret cesse d'augmenter pour le restant du temps. On peut donc ici faire beaucoup mieux que la vitesse $\mathcal{O}(\sqrt{T})$ assurée dans le pire des cas.

État de l'art Plusieurs solutions pour exploiter la simplicité de la suite d'observations ont ainsi été proposées dans la littérature. Elles s'appuient majoritairement sur la calibration d'un paramètre d'apprentissage (qui apparaît dans la plupart des algorithmes de mélange). Certaines reposent sur une calibration judicieuse (souvent décroissante) du paramètre d'apprentissage [45, 60, 89, 58, 146]. D'autres agrègent plusieurs paramètres d'apprentissage à partir d'approches souvent très différentes [46, 126, 48].

Les bornes prenant en compte la variance des experts (aussi appelées bornes du second-ordre) permettent de capturer la difficulté de la suite à prévoir. Cesa-Bianchi et al. [45], Hazan et Kale [89] proposent ainsi des stratégies qui vérifient

$$\sum_{t=1}^{T} \widehat{\ell}_t \leqslant \min_{1\leqslant k\leqslant K} \left\{ \sum_{t=1}^{T} \ell_{k,t} \;+\; c\sqrt{\ln K \sum_{t=1}^{T} v_t} + c \right\}, \tag{1.11}$$

où à l'instant t, $\widehat{\ell}_t = \sum_{k=1}^{K} \widehat{p}_{k,t}\ell_{k,t}$ est la perte linéarisée du statisticien[‡], $\ell_{k,t} = \ell(x_{k,t}, y_t)$ est la perte de l'expert k et $v_t = \sum_{k\leqslant K} \widehat{p}_{k,t}\big(\widehat{\ell}_t - \ell_{k,t}\big)^2$ est la variance des pertes sous la distribution $\widehat{\boldsymbol{p}}_t$ jouée par l'algorithme. Comme dans toute la suite de cette introduction, $c > 0$ est une constante indépendante de T (et éventuellement de K) qui peut être différente d'une écriture à une autre. Cette borne possède de nombreuses propriétés intéressantes montrées dans les références ci-dessus. En particulier, si les poids de l'algorithme se concentrent sur un expert, ce qui arrive si un expert surpasse les autres, le regret n'augmente plus à partir d'un certain temps. L'inconvénient de la borne ci-dessus est son uniformité : elle ne reflète pas le fait qu'il est plus difficile de se comparer à certains experts plutôt qu'à d'autres.

Une autre forme de borne du second ordre qui résoudrait cet inconvénient serait de la forme

$$\sum_{t=1}^{T} \widehat{\ell}_t \leqslant \min_{k=1,\dots,K} \left\{ \sum_{t=1}^{T} \ell_{k,t} + c\sqrt{\log K \sum_{t=1}^{T} \ell_{k,t}^2} + c \right\}. \tag{1.12}$$

Celle-ci exprimerait pleinement le compromis biais-variance auquel fait face le statisticien. Malheureusement, son obtention reste encore aujourd'hui un problème ouvert : pour l'obtenir le statisticien doit réussir à calibrer correctement le paramètre d'apprentissage sans connaître à l'avance le bon expert réalisant le meilleur compromis. Ce problème s'est avéré si difficile qu'il a hérité du surnom d'« impossible tuning » dans la communauté.

Contributions 1.10. Dans le chapitre 2, nous introduisons une nouvelle forme de borne du second ordre :

$$\sum_{t=1}^{T} \widehat{\ell}_t \leqslant \min_{k=1,\dots,K} \left\{ \sum_{t=1}^{T} \ell_{k,t} + c\sqrt{\log K \sum_{t=1}^{T} \big(\widehat{\ell}_t - \ell_{k,t}\big)^2} + c \right\}. \tag{1.13}$$

[‡]Par convexité de la fonction de perte en son premier argument, la perte du statisticien $\ell(\widehat{y}_t, y_t)$ est majorée par la perte linéarisée $\sum_{k=1}^{K} \widehat{p}_{k,t}\ell(x_{k,t}, y_t)$.

Nous avons déduit de cette borne des propriétés d'adaptation intéressantes. Quoi qu'il arrive, la vitesse en $\mathcal{O}\big(\sqrt{T \log K}\big)$ du regret est assurée. Mais la borne est significativement plus faible dans certaines situations où l'apprentissage est plus simple.

Le cas des faibles pertes Le théorème suivant, prouvé au chapitre 2, exprime le fait que pour une stratégie vérifiant le regret (1.13), le regret est au plus de l'ordre de $\mathcal{O}(\sqrt{L_T^\star \log K})$, où $L_T^\star$ est la perte cumulée du meilleur expert. Comme $L_T^\star \leqslant T$, la borne du pire des cas est garantie tout en obtenant une meilleure vitesse si un des experts subit une faible perte cumulée.

Théorème 1.2. *Si une stratégie vérifie la borne de regret (1.13) pour toute suite d'observations, alors elle vérifie également pour tout $k = 1, \dots, K$*

$$\sum_{t=1}^T \widehat{\ell}_t - \sum_{t=1}^T \ell_{k,t} \leqslant c\sqrt{\log K \sum_{t=1}^T \Big(\ell_{k,t} - \widehat{\ell}_t\Big)_+} + c \ln K \leqslant c\sqrt{\log K \sum_{t=1}^T \ell_{k,t}} + c \ln K \,,$$

où $(\,\cdot\,)_+ = \max\{\cdot, 0\}$.

Le cas de pertes i.i.d. Le Théorème 1.3 montre que dans un cadre stochastique, si un expert est significativement meilleur que les autres, la borne (1.13) garantit un regret borné.

Théorème 1.3. *On suppose que les pertes des experts $\ell_t = (\ell_{1,t}, \dots, \ell_{K,t}) \in [0,1]^K$ sont des variables aléatoires identiquement distribuées et qu'il existe un expert $k^\star$ tel que*

$$\forall t \geqslant 1 \qquad \min_{k \neq k^\star} \mathbb{E}\big[\ell_{k,t} - \ell_{k^\star,t}\big] \geqslant \alpha \qquad\qquad \text{pour un certain } \alpha > 0.$$

Alors si une stratégie vérifie la borne de regret (1.13) pour toute suite d'observations, alors elle vérifie également

$$\mathbb{E}\left[\max_{k=1,\dots,K} \sum_{t=1}^T (\widehat{\ell}_t - \ell_{k,t})\right] \leqslant c\frac{\log K}{\alpha} + c\sqrt{\frac{\log K}{\alpha}} \,.$$

Le résultat est également valable en grande probabilité. Les précédentes analyses d'algorithmes adaptatifs étaient spécifiques aux algorithmes utilisés. Notre contribution est de montrer que l'adaptation à ces situations faciles est intrinsèque à la borne (1.13). En d'autres mots, il suffit de montrer (1.13) pour assurer qu'une stratégie tire profit de ces situations faciles. Wintenberger [146] a récemment dérivé de la borne (1.13) dans un cadre stochastique très général des bornes sur le risque prédictif, qui sont optimales dans le cas d'observations i.i.d.

Comment cette borne est-elle obtenue ? Pour obtenir la borne (1.13), nous avons développé une nouvelle technique qui associe le choix d'un paramètre d'apprentissage associé à chaque expert. Ceux-ci ont la forme

$$\eta_k = \sqrt{\frac{\log K}{\sum_{t=1}^T \big(\widehat{\ell}_t - \ell_{k,t}\big)^2}} \,. \tag{1.14}$$

Nous avons instancié cette technique à deux algorithmes. Le premier est une extension de l'algorithme Prod de Cesa-Bianchi et al. [45]. Il choisit à chaque instant les poids

$$\widehat{p}_{k,t} = \frac{\eta_k \prod_{s=1}^{t-1} \left(1 + \eta_k \left(\widehat{\ell}_s - \ell_{k,s}\right)\right)}{\sum_{j=1}^{K} \eta_j \prod_{s=1}^{t-1} \left(1 + \eta_j \left(\widehat{\ell}_s - \ell_{j,s}\right)\right)} \qquad \text{pour tout } k = 1, \dots, K\,. \qquad \text{(ML-Prod)}$$

La seconde stratégie est inspirée de l'algorithme de mélange par poids polynomiaux de Cesa-Bianchi et Lugosi [42]. Elle choisit les poids

$$\widehat{p}_{k,t} = \frac{\eta_k^2 \left(\sum_{s=1}^{t-1} \left(\widehat{\ell}_s - \ell_{k,s}\right)\right)_+}{\sum_{j=1}^{K} \eta_j^2 \left(\sum_{s=1}^{t-1} \left(\widehat{\ell}_s - \ell_{j,s}\right)\right)_+} \qquad \text{pour tout } k = 1, \dots, K\,, \qquad \text{(ML-Poly)}$$

où $(\,\cdot\,)_+ \stackrel{\text{def}}{=} \max\{\cdot, 0\}$.

Bien sûr, le choix des paramètres (1.14) nécessite la connaissance à l'avance de toutes les données, ce qui n'est pas réaliste. Les deux algorithmes peuvent être obtenus de façon totalement adaptative en calibrant des paramètres d'apprentissage $\eta_{k,t}$ qui dépendent du temps et n'utilisent que l'information passée. Les versions adaptatives des deux algorithmes ont des regrets de l'ordre :

$$\begin{aligned} &\mathcal{O}\left(\sqrt{(\log\log T)(\log K)\sum_{t=1}^{T}\left(\widehat{\ell}_t - \ell_{k,t}\right)^2}\right) && \text{pour ML-Prod} \\ &\mathcal{O}\left(\sqrt{(\log T) K \sum_{t=1}^{T}\left(\widehat{\ell}_t - \ell_{k,t}\right)^2}\right) && \text{pour ML-Poly}\ . \end{aligned}$$

Même si la dépendance de ML-Poly en le nombre d'experts est légèrement sous-optimale, l'algorithme a démontré d'excellentes performances pratiques sur les différents jeux de données d'EDF. Nous l'utilisons dans les chapitres 4, 8 et 9.

1.4. Mélanges et prévision en loi

Un autre axe de la thèse a consisté à gérer les incertitudes des prévisions des algorithmes de mélange. L'objectif, ici, n'est plus de seulement proposer une prévision ponctuelle $\widehat{y}_t$ de l'observation y_t mais de l'assortir d'une mesure d'incertitude.

La littérature à ce sujet est riche dans un cadre stochastique avec de nombreuses techniques différentes : méthodes d'estimation de densité, de mélange (Bayesian model averaging), de bootstrap, de propagation d'incertitude, etc. Cependant, dans le cadre des suites arbitraires, la grande majorité des travaux se limite à fournir des prévisions ponctuelles de la moyenne (cf. Cesa-Bianchi et Lugosi [43]). C'est le cas notamment des précédents travaux effectués à EDF R&D sur la prévision de la consommation électrique, à savoir Devaine et al. [60], Goude [80].

Pourtant, le problème de prévision d'incertitudes ou de lois est un problème d'actualité important pour la gestion de l'énergie mais aussi dans de nombreux autres domaines, afin de gérer les risques encourus et d'optimiser la production. De nombreuses initiatives de recherche vont ainsi dans ce sens comme la compétition GEFCom2014 (cf. chapitre 8).

1.4.1. Gestion des incertitudes en prévision de suites arbitraires (cf. chapitre 7)

Nous cherchons à formaliser le cadre de la prévision en loi de suites arbitraires avec l'aide d'experts. Dans ce but, nous distinguons plusieurs cadres différents en fonction de l'information apportée par les experts sur l'incertitude de leurs prévisions :

- soit les experts fournissent des lois (ou des intervalles de prévision), il est naturel de chercher à les combiner, de la même manière que l'on sait déjà le faire pour les prévisions ponctuelles ;
- soit les experts ne fournissent que des prévisions ponctuelles, il s'agit alors de former une prévision d'incertitude ex-nihilo. Cela part d'une intuition pratique des opérateurs ayant en charge la prévision d'électricité. Quand les opérateurs reçoivent plusieurs prévisions, si celles-ci sont significativement différentes, cela signifie un risque accru pour la prévision. En revanche, si les prévisions des experts sont semblables, le risque semble plus faible.

Dans les deux cadres, de nombreuses définitions naturelles inspirées de la littérature stochastique sont possibles pour le critère de performance (i.e. l'objectif à réaliser par le statisticien). Malheureusement, parmi celles-ci, certaines sont impossibles à vérifier en suites arbitraires (comme le compromis niveau-largeur pour les intervalles de prévision) et les autres sont facilement résolues par le statisticien à partir de techniques standard d'agrégation séquentielle[§]. C'est le cas par exemple pour des critères reposant sur la log-vraisemblance, sur la statistique de Kolmogorov, ou sur la perte quantile de Koenker et Bassett [101]. La minimisation de la perte quantile de Koenker et Bassett [101] définie par

$$\ell\colon (x,y) \mapsto (y-x)\big(\alpha - \mathbb{1}_{\{y<x\}}\big) \tag{1.15}$$

illustrée en figure 1.3 nous a d'ailleurs beaucoup inspirés lors de la compétition GEFCom2014.

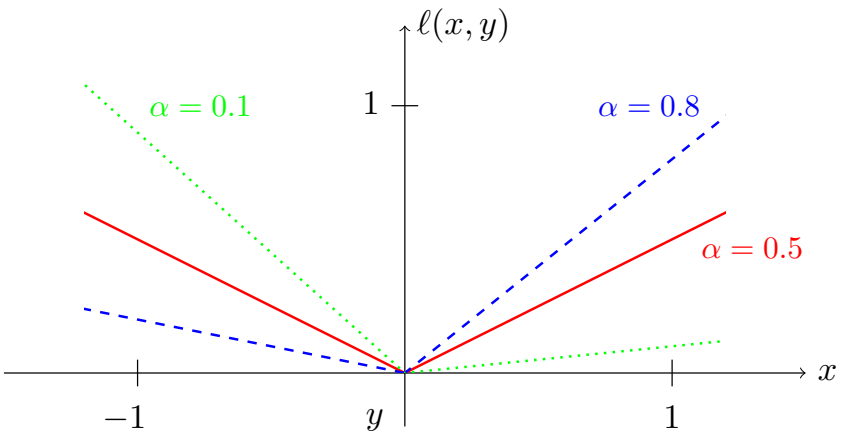


FIGURE 1.3. : La fonction de perte quantile de Koenker et Bassett [101] pour les quantiles $\alpha \in \{0.1, 0.5, 0.8\}$. La perte reflète le déséquilibre désiré dans la prévision quantile.

1.4.2. La compétition GEFCom2014 (cf. chapitre 8)

La compétition GEFCom2014 a eu lieu d'août à décembre 2014 avec pour objectif de développer et comparer des méthodes de référence pour la prévision en loi dans le monde de l'énergie. Nous avons participé à deux épreuves : l'une sur la consommation électrique et l'autre sur le prix de l'électricité. Les deux épreuves reposaient sur le même principe : nous devions prévoir la loi de la

[§]Il s'agit de prouver des bornes de regret pour la fonction de perte adéquate

quantité d'intérêt, nous étions évalués avec la perte quantile de Koenker et Bassett [101] cumulée et le jeu de données était révélé séquentiellement au fur et à mesure de la compétition.

Jeu 1 : la consommation électrique Dans cette épreuve, les participants devaient prévoir séquentiellement (mois par mois) la loi de la consommation électrique du prochain mois à l'aide des données passées de consommation et de variables de températures. Nous avons développé une méthode de prévision probabiliste semi-paramétrique s'inspirant de l'expérience d'EDF R&D sur les modèles additifs généralisés. L'idée consiste dans un premier temps à estimer les effets non-linéaires des variables exogènes (météorologiques, calendaires) sur la consommation électrique et sur les erreurs de prévision. Dans un deuxième temps, nous prévoyons la loi de la consommation électrique à l'aide d'une régression linéaire quantile qui minimise la perte quantile de Koenker et Bassett [101]. Un exemple de prévisions obtenues pour le mois de mai 2011 est représenté en figure 1.4.

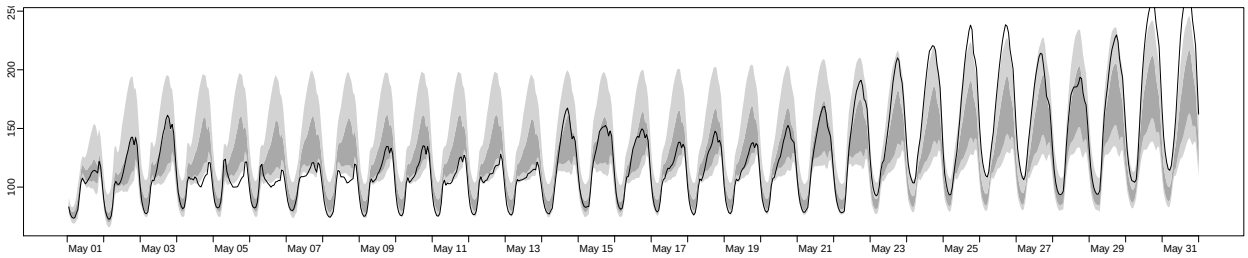


FIGURE 1.4. : Prévision probabiliste de la consommation électrique du mois de mai 2011. L'intervalle de confiance 50% (resp. 90%) est représenté en gris foncé (resp. gris clair).

Malheureusement, limités par le temps, nous n'avons pu élaborer qu'un seul expert. Les autres tentatives n'étant pas suffisamment différentes ou performantes. Cela n'était donc pas suffisant pour pouvoir appliquer avec succès les méthodes d'agrégation.

Jeu 2 : le prix de l'électricité Comme celui-ci est beaucoup plus volatil que la consommation électrique, il n'est pas raisonnable de le prévoir correctement à de grands horizons de temps. Dans cette épreuve, nous devions donc seulement prévoir le prix de la prochaine journée à l'aide des observations passées du prix (2 ans d'historique) et de variables exogènes (prévision de la consommation électrique). Ici, notre équipe disposait de beaucoup moins d'expertise. Nous avons pu tester de nombreux modèles de prévision : modèles autorégressifs, modèles additifs généralisés, forêts aléatoires (quantile), boosting, régression linéaire parcimonieuse,... En particulier, nous avons agrégé ces prévisions en utilisant une version linéarisée de ML-Poly avec la perte quantile. Plus précisément, pour former notre prévision quantile du prix y_t à l'instant t :

1. nous avons construit 13 méthodes de prévision (nos experts) formant des prévisions ponctuelles $x_{k,t}$ de y_t ;
2. nous avons utilisé l'astuce de Kivinen et Warmuth [97] pour pouvoir se comparer à l'ensemble des stratégies linéaires $\Theta = \left\{ \boldsymbol{u} \in \mathbb{R}^K : \|\boldsymbol{u}\|_1 \leqslant 2 \right\}$ (cf. paragraphe 1.3.1.b) ;
3. nous avons ensuite utilisé l'algorithme ML-Poly avec la fonction de perte quantile (1.15) sur une série de $m = 5\,000$ observations. À chaque pas $i = 1, \ldots, m$ de l'algorithme ML-Poly, celui-ci tirait aléatoirement uniformément avec remise une observation y_s dans les trente jours précédant y_t et formait un vecteur de poids $\widehat{\boldsymbol{u}}_i \in \Theta$;
4. nous avons formé notre prévision quantile $\widehat{y}_t^{(\alpha)}$ en combinant les prévisions des 13 experts selon les poids $\bar{\boldsymbol{u}}_m = (1/m) \sum_{i=1}^m \widehat{\boldsymbol{u}}_i$.

Pour les deux jeux de données nos résultats se sont révélés très prometteurs (nous avons remporté les deux compétitions).

1.5. Autres apports à la prévision pour les marchés de l'énergie

Un autre objectif de la thèse était de démontrer l'universalité des méthodes de mélange au sein d'EDF R&D. Dans ce but, nous les avons appliquées à de nombreux jeux de données du marché de l'énergie (consommation électrique à différentes mailles géographiques, prix de l'électricité, chaleur) avec différents objectifs finaux (différents horizons de prévision, prévision probabiliste). Nous décrivons ici deux contributions pratiques de la thèse.

Horizon de prévision Dans le chapitre 9, nous nous intéressons à la prévision à différents horizons de temps allant d'une heure à 72 heures à l'avance. Nous généralisons les méthodes de mélange à ce cadre et les appliquons à deux jeux de données : le premier sur la prévision de chaleur (voir paragraphe 1.1.2), le second sur la consommation électrique.

Prévisions spécialisées Dans le chapitre 4, nous créons des experts qui sont spécialisés à certaines situations météorologiques comme les périodes de grands froids. Cela permet d'enrichir le jeu d'expert en apportant de la variété. Nous réduisons ainsi significativement les erreurs de prévision du mélange.

Nous pouvons cependant nous demander s'il est judicieux de prendre en compte la prévision d'un expert dans une situation non adéquate (par exemple, un expert spécialisé pour le froid en cas de forte chaleur). Dans le cadre des experts spécialisés, à chaque instant t, chaque expert assortit sa prévision $x_{k,t} \in \mathbb{R}$ d'une mesure de confiance $I_{k,t} \in [0,1]$ en celle-ci. Dans le cas extrême $I_{k,t} = 0$, l'expert k est inactif et sa prévision (s'il en forme une) n'est pas prise en compte dans le mélange.

La généralisation des méthodes de mélange à ce cadre a déjà été étudiée dans Devaine et al. [60]. Dans le chapitre 2, nous montrons l'intérêt des bornes de regret de la forme (1.13) dans le cadre des experts spécialisés. Nous obtenons pour la première fois des bornes sur le regret spécialisé du bon ordre de grandeur pour les indices de confiance $I_{k,t}$. Toute stratégie qui vérifie la borne de regret (1.13) vérifie en effet également dans le cadre des experts spécialisés la borne

$$\sum_{t=1}^{T} I_{k,t}\big(\widehat{\ell}_t - \ell_{k,t}\big) \leqslant c\sqrt{(\log K)\sum_{t=1}^{T} I_{k,t}^2}\,.$$

1.6. Implémentation

Les différentes études empiriques de cette thèse sont réalisées avec le logiciel `R`. J'ai développé le package `opera`[¶] (cf. [12]) qui propose des outils pour la prévision séquentielle de séries temporelles. Une brève documentation ainsi qu'un exemple d'utilisation est disponible en annexe. On décrit ci-dessous les principales fonctionnalités du package dans sa version actuelle.

[¶] Online Prediction by ExpeRts Aggregation

Stratégies d'agrégation disponibles Le package `opera` inclut trois algorithmes d'agrégation séquentielle robuste éprouvés par la littérature : le mélange par poids exponentiels (Littlestone et Warmuth [107], Vovk [139]), l'algorithme fixed share (Herbster et Warmuth [92]) et la régression ridge (Azoury et Warmuth [21], Vovk [142]). Leurs paramètres d'apprentissage peuvent soit être fixés par l'utilisateur, soit optimisés séquentiellement par l'algorithme sur une grille en suivant la méthode de Devaine et al. [60]. Les stratégies introduites au chapitre 2 de cette thèse (ML-Poly, ML-Prod) ainsi que l'algorithme BOA de Wintenberger [146] sont aussi proposées avec la calibration séquentielle théorique des paramètres d'apprentissage.

Fonctions de perte Quatre fonctions de pertes ℓ sont disponibles : la perte carrée $(x, y) \mapsto (x - y)^2$, la perte absolue $(x, y) \mapsto |x - y|$, le pourcentage de perte absolue $(x, y) \mapsto |x - y|/y$ et la perte quantile $(x, y) \mapsto (y - x)(\alpha - \mathbb{1}_{\{y<x\}})$. Ces fonctions de perte peuvent être également utilisées dans leur version gradient pour pouvoir approcher la meilleure combinaison convexe (cf. paragraphe 1.3.1.a).

Le package contient également des fonctions pour calculer les pertes, subies par le statisticien ou les experts, ainsi que certaines erreurs d'approximation du paragraphe 1.3.1. Il propose pour finir l'extension des méthodes ci-dessus au cadre des experts spécialisés.

1.7. Conclusion et perspectives

Dans cette thèse nous sommes partis d'une problématique opérationnelle d'EDF R&D : comment réunir de façon robuste et adaptative les nombreux modèles de prévision développés à EDF ? Dans ce but, nous adoptons le cadre de l'agrégation séquentielle de prédicteurs et nous poursuivons les travaux initiés par Goude [80] et Devaine et al. [60] à EDF R&D. Nous apportons à ce cadre des contributions théoriques (travaux sur l'erreur d'approximation et l'erreur d'estimation) et pratiques (application et adaptation des méthodes à plusieurs jeux de données réelles).

L'organisation de ce manuscrit est résumée en figure 1.5. Les chapitres sont assemblés en quatre thèmes. Le premier réunit les chapitres 2 à 4 et étudie différentes améliorations de la performance du mélange. La seconde partie du manuscrit présente des résultats sur la prévision séquentielle non paramétrique (chapitres 5 et 6). Les chapitres 7 et 8 s'intéressent à la prévision d'incertitudes (ou même de lois). Enfin, le manuscrit s'achève par plusieurs études empiriques sur des jeux de données réelles d'EDF R&D (chapitres 9 et 10).

Un certain nombre de perspectives de recherche toujours ouvertes sont soulignées au long du manuscrit. Par exemple, la régression séquentielle non paramétrique a encore été très peu étudiée dans le cadre des suites individuelles. Dans ce manuscrit, nous proposons des méthodes efficaces pour traiter les classes de fonctions Lipschitz et Hölder en faible dimension. Il peut être intéressant de développer des méthodes de prévision efficaces pour d'autres classes de fonctions (espace de Besov, classes parcimonieuses) ou pour des dimensions plus importantes. Le problème de la grande dimension est d'ailleurs un problème d'actualité pour EDF R&D qui aura bientôt à disposition en temps réel la consommation électrique de 30 millions de foyers français. Tirer profit d'une telle masse d'informations représente à la fois un défi majeur et une grande opportunité pour EDF. Peu de résultats existent cependant dans le cadre des suites arbitraires (voir Gerchinovitz [78]) et l'obtention d'une méthode séquentielle robuste qui propose des combinaisons parcimonieuses de façon efficace reste pour l'instant une question ouverte.

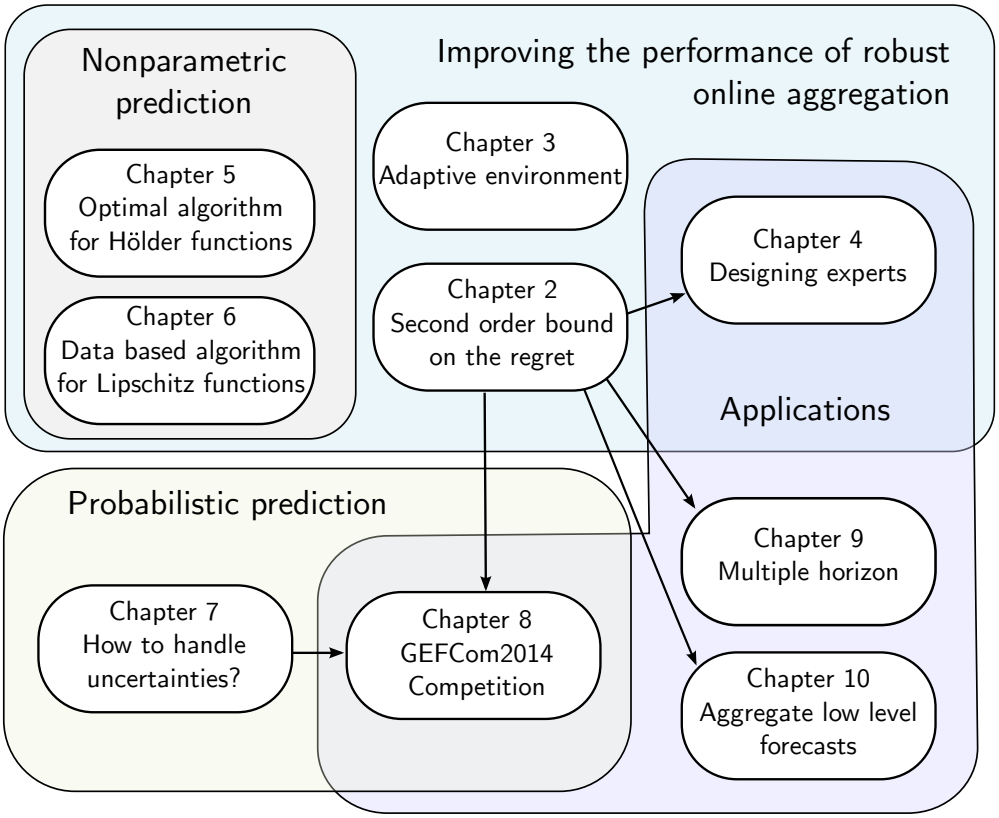


FIGURE 1.5. : Contents and organization of the thesis manuscript

I

Improving the performance of robust online aggregation

Introduction

In this part we consider the setting of sequential prediction of arbitrary sequences with expert advice (see the monograph of Cesa-Bianchi and Lugosi [43] for a nice introduction), which unfolds as follows. A player is asked to predict element by element a sequence of real observations $y_1, \dots, y_T$. Before each time step $t = 1, \dots, T$, the player has access to a finite number K of base forecasting methods (henceforth referred to as experts) that provide predictions $x_{k,t}$. The goal of the player is to aggregate sequentially the expert forecasts so as to minimize his cumulative loss, which can be split up into two terms as follows:

$$\underbrace{\sum_{t=1}^{T} \ell(\widehat{y}_t, y_t)}_{\text{performance of the player}} \stackrel{\text{def}}{=} \underbrace{\min_{k=1,\dots,K} \sum_{t=1}^{T} \ell(x_{k,t}, y_t)}_{\substack{\text{reference performance} \\ \text{(approximation error)}}} + \underbrace{R_T}_{\substack{\text{regret} \\ \text{(estimation error)}}}, \qquad \text{(R1)}$$

where ℓ is a nonnegative loss function and $\widehat{y}_t$ is the combined prediction formed by the player at time t. The first term in the right-hand side of (R1) is the reference performance (also referred to as approximation error) that the player aims at competing with. Usually it is the performance obtained by the best fixed expert (as displayed in Equation (R1)). The second term is the regret (also called estimation error). It evaluates the ability of the player to retrieve the reference strategy (i.e., the strategy that achieves the reference performance) online.

In the following three chapters we focus on improving player's performance by using two levers: we aim at reducing both the approximation error (Chapters 2, 3, and 4) and the regret (Chapter 2).

In Chapter 2 we control the regret by designing two new combining strategies (ML-Poly and ML-Prod). These strategies are based on fully data-driven calibration of their parameters and new tuning techniques (multiple learning rates). We prove a new form of second-order regret bound that adapts well to the inherent difficulty of the data. The chapter ends with the study of alternative comparison classes (based on Markov chain modeling) in order to improve the reference performance.

Chapter 3 reduces the approximation error by allowing the reference strategy to evolve over time. We prove performance bounds under an analysis that significantly unifies a large body of previous work.

Chapter 4 is an empirical paper that also focuses on the approximation error. It studies several ways to enlarge the set of experts based on ideas from bagging or boosting techniques in order to make the approximation error significantly smaller. Then, it studies the performance obtained by several algorithms (including some of Chapters 2 and 3) on a real world data set.

2

A second-order bound with excess losses

We study online aggregation of the predictions of experts, and first show new second-order regret bounds in the standard setting, which are obtained via a version of the Prod algorithm (and also a version of the polynomially weighted average algorithm) with multiple learning rates. These bounds are in terms of excess losses, the differences between the instantaneous losses suffered by the algorithm and the ones of a given expert. We then demonstrate the interest of these bounds in the context of experts that report their confidences as a number in the interval $[0, 1]$ using a generic reduction to the standard setting. We conclude by two other applications in the standard setting, which improve the known bounds in case of small excess losses and show a bounded regret against i.i.d. sequences of losses.

Contents

NOTA: This chapter is a joint work with Gilles Stoltz and Tim van Erven. It is based on the conference paper [5]. Section 2.6 is published here for the first time.

2.1. Introduction

In the (simplest) setting of prediction with expert advice, a learner has to make online sequential predictions over a series of rounds, with the help of K experts [70, 107, 140, 43]. In each round $t = 1, \dots, T$, the learner makes a prediction by choosing a vector $\boldsymbol{p}_t = (p_{1,t}, \dots, p_{K,t})$ of nonnegative weights that sum to one. Then every expert k incurs a loss $\ell_{k,t} \in [a, b]$ and the learner's loss is $\widehat{\ell}_t = \boldsymbol{p}_t^\top \boldsymbol{\ell}_t = \sum_{k=1}^K p_{k,t}\ell_{k,t}$, where $\boldsymbol{\ell}_t = (\ell_{1,t}, \dots, \ell_{K,t})$. The goal of the learner is to control his cumulative loss, which he can do by controlling his regret $R_{k,T}$ against each expert k, where $R_{k,T} = \sum_{t \leqslant T} \big(\widehat{\ell}_t - \ell_{k,t}\big)$. In the worst case, the best bound on the standard regret $R_{k,T}$ that can be guaranteed is of order $\mathcal{O}\big(\sqrt{T \ln K}\big)$; see, e.g., Cesa-Bianchi and Lugosi [43], but this can be improved. For example, when losses take values in $[0, 1]$, $R_{k,T} = \mathcal{O}\big(\sqrt{L_{k,T} \ln K}\big)$, with $L_{k,T} = \sum_{t=1}^T \ell_{k,t}$, is also possible, which is better when the losses are small—hence the name *improvement for small losses* for this type of bounds [43].

Second-order bounds Cesa-Bianchi et al. [45] raised the question of whether it was possible to improve even further by proving second-order (variance-like) bounds on the regret. They could establish two types of bound, each with its own advantages. The first is of the form

$$R_{k,t} \leqslant \frac{\ln K}{\eta} + \eta \sum_{t=1}^T \ell_{k,t}^2 \tag{2.1}$$

for all experts k, where $\eta \leqslant 1/2$ is a parameter of the algorithm. If one could optimize η with hindsight knowledge of the losses, this would lead to the desired bound

$$R_{k,T} = \mathcal{O}\left(\sqrt{\ln K \sum_{t=1}^T \ell_{k,t}^2}\right), \tag{2.2}$$

but, unfortunately, no method is known that actually achieves (2.2) for all experts k simultaneously without such hindsight knowledge. As explained by Cesa-Bianchi et al. [45] and Hazan and Kale [89], the technical difficulty is that the optimal η would depend on $\sum_t \ell_{k_T^\star,t}^2$, where

$$k_T^\star \in \operatorname*{argmin}_{k=1,\dots,K} \left\{ \sum_{t=1}^T \ell_{k,t} + \sqrt{\ln K \sum_{t=1}^T \ell_{k,t}^2} \right\}.$$

But, because $k_T^\star$ can vary with T, the sequence of the $\sum \ell_{k_t^\star,t}^2$ is not monotonic and, as a consequence, standard tuning methods (like for example the doubling trick) cannot be applied directly on this sequence (only on the least non-decreasing sequence larger than it, which is then the key quantity in the regret bound though it is difficult to interpret).

This is why this issue — when hindsight bounds seem too good to be obtained in a sequential fashion — is sometimes referred to as the problem of *impossible tunings*. Improved bounds with respect to (2.1) have been obtained by Hazan and Kale [89] and Chiang et al. [49] but they suffer from the same impossible tuning issue.

The second type of bound distinguished by Cesa-Bianchi et al. [45] is of the form

$$R_{k,T} = \mathcal{O}\left(\sqrt{\ln K \sum_{t=1}^T v_t}\right), \tag{2.3}$$

uniformly over all experts k, where $v_t = \sum_{k \leqslant K} p_{k,t}\big(\widehat{\ell}_t - \ell_{k,t}\big)^2$ is the variance of the losses at instance t under distribution $\boldsymbol{p}_t$. It can be achieved by a variant of the exponentially weighted average forecaster using the appropriate tuning of a time-varying learning rate η_t [45, 58]. The bound (2.3) was shown in the mentioned references to have several interesting consequences (see Section 2.5). Its main drawback comes from its uniformity: it does not reflect that it is harder to compete with some experts than with other ones.

Excess losses Instead of uniform regret bounds like (2.3), we aim to get expert-dependent regret bounds. We see this result as paving the way for solving the open problem of impossible tuning stated in (2.2).

The key quantities in our analysis turn out to be the instantaneous *excess losses* $\ell_{k,t} - \widehat{\ell}_t$, and we provide in Sections 2.2 and 2.3 a new second-order bound of the form

$$R_{k,T} = \mathcal{O}\left(\sqrt{\ln K \sum_{t=1}^{T} (\widehat{\ell}_t - \ell_{k,t})^2}\right), \tag{2.4}$$

which holds for all experts k simultaneously. To achieve this bound, we develop a variant of the Prod algorithm of Cesa-Bianchi et al. [45] with two innovations: first we extend the analysis for Prod to multiple learning rates η_k (one for each expert) in the spirit of a variant of the Hedge algorithm with multiple learning rates proposed by Blum and Mansour [29]. Standard tuning techniques of the learning rates would then still lead to an additional $\mathcal{O}(\sqrt{K \ln T})$ multiplicative factor, so, secondly, we develop new techniques that bring this factor down to $\mathcal{O}(\ln \ln T)$, which we consider to be essentially a constant. Duchi et al. [63] also studied learning with multiple learning rates in a somewhat different context, namely, general online convex optimization; but the obtained regret bound is uniform over the experts.

The interest of the bound (2.4) is demonstrated in Sections 2.4 and 2.5, as well as in the recent paper by Wintenberger [146]. Section 2.4 considers the setting of prediction with experts that report their confidences as a number in the interval $[0, 1]$, which was first studied by Blum and Mansour [29]. Our general bound (2.4) leads to the first bound on the confidence regret that scales optimally with the confidences of each expert. Section 2.5 returns to the standard setting described at the beginning of this paper: we show an improvement for small excess losses, which supersedes the basic improvement for small losses described at the beginning of the introduction. Also, we prove that in the special case of independent, identically distributed losses, our bound leads to a constant regret. Finally, Wintenberger [146] shows that bounds of the form (2.4) entail regret bounds on the cumulative predictive risks of the associated strategy without any assumption on the underlying stochastic process (in particular, without the usual dependency assumptions).

2.2. A new regret bound in the standard setting

We extend the Prod algorithm of Cesa-Bianchi et al. [45] to work with multiple learning rates.

Theorem 2.1. *For all sequences of loss vectors $\boldsymbol{\ell}_t \in [0, 1]^K$, the cumulative loss of Algorithm 1*

Algorithm 1: Prod with multiple learning rates (ML-Prod)

Parameters: a vector $\boldsymbol{\eta} = (\eta_1, \dots, \eta_K)$ of positive learning rates
Initialization: a vector $\boldsymbol{w}_0 = (w_{1,0}, \dots, w_{K,0})$ of nonnegative weights that sum to 1
For each round $t = 1, 2, \dots$

1. form the mixture $\boldsymbol{p}_t$ defined component-wise by
$$p_{k,t} = \frac{\eta_k w_{k,t-1}}{\sum_{j=1}^K \eta_j w_{j,t-1}}$$
2. observe the loss vector $\boldsymbol{\ell}_t$ and incur loss $\widehat{\ell}_t = \boldsymbol{p}_t^\top \boldsymbol{\ell}_t$P
3. for each expert k perform the update
$$w_{k,t} = w_{k,t-1}\big(1 + \eta_k\big(\widehat{\ell}_t - \ell_{k,t}\big)\big)$$

run with learning rates $\eta_k \leqslant 1/2$ is bounded by

$$\sum_{t=1}^T \widehat{\ell}_t \leqslant \min_{1\leqslant k\leqslant K} \left\{ \sum_{t=1}^T \ell_{k,t} + \frac{1}{\eta_k} \ln \frac{1}{w_{k,0}} + \eta_k \sum_{t=1}^T \big(\widehat{\ell}_t - \ell_{k,t}\big)^2 \right\}.$$

If we could optimize the bound of the theorem with respect to η_k, we would obtain the desired result:

$$\sum_{t=1}^T \widehat{\ell}_t \leqslant \min_{1\leqslant k\leqslant K} \left\{ \sum_{t=1}^T \ell_{k,t} + 2\sqrt{\sum_{t=1}^T V_{k,t} \ln \frac{1}{w_{k,0}}} \right\} \tag{2.5}$$

where $V_{k,t} = \big(\widehat{\ell}_t - \ell_{k,t}\big)^2$. The question is therefore how to get the optimized bound (2.5) in a fully sequential way. Working in regimes (resorting to some doubling trick) seems suboptimal, since K quantities $\sum_t V_{k,t}$ need to be controlled simultaneously and new regimes will start as soon as one of these quantities is larger than some dyadic threshold. This would lead to an additional $\mathcal{O}(\sqrt{K \ln T})$ multiplicative factor in the bound. We propose in Section 2.3 a finer scheme, based on time-varying learning rates $\eta_{k,t}$, which only costs a multiplicative $\mathcal{O}(\ln \ln T)$ factor in the regret bounds. Though the analysis of a single time-varying parameter is rather standard since the paper by Auer et al. [20], the analysis of multiple such parameters is challenging and does not follow from a routine calculation. That the "impossible tuning" issue does not arise here was quite surprising to us.

Empirical variance of the excess losses A consequence of (2.5) is the following bound, which is in terms of the empirical variance of the excess losses $\ell_{k,t} - \widehat{\ell}_t$:

$$\sum_{t=1}^T \widehat{\ell}_t \leqslant \min_{1\leqslant k\leqslant K} \left\{ \sum_{t=1}^T \ell_{k,t} + 4 \ln \frac{1}{w_{k,0}} + 2\sqrt{\sum_{t=1}^T \left(\widehat{\ell}_t - \ell_{k,t} - \frac{R_{k,T}}{T}\right)^2 \ln \frac{1}{w_{k,0}}} \right\}. \tag{2.6}$$

Proposition 2.2. *Suppose losses take values in $[0,1]$. If (2.5) holds, then (2.6) holds.*

Proof A bias-variance decomposition indicates that, for each k,

$$\sum_{t=1}^{T} V_{k,t} = \sum_{t=1}^{T} \big(\widehat{\ell}_t - \ell_{k,t}\big)^2 = \sum_{t=1}^{T} \big(\widehat{\ell}_t - \ell_{k,t} - R_{k,T}/T\big)^2 + T\left(R_{k,T}/T\right)^2. \tag{2.7}$$

It is sufficient to prove the result when the minimum is restricted to k such that $R_{k,T} \geqslant 0$. For such k, (2.5) implies that $R_{k,T}^2 \leqslant 4T \ln(1/w_{k,0})$. Substituting this into the rightmost term of (2.7), the result into (2.5), and using that $\sqrt{x+y} \leqslant \sqrt{x} + \sqrt{y}$ for $x, y \geqslant 0$ concludes the proof. ∎

Proof (of Theorem 2.1) The proof follows from a simple adaptation of Lemma 2 in Cesa-Bianchi et al. [45] and takes some inspiration from Section 6 of Blum and Mansour [29].

For $t \geqslant 0$, we denote by $\boldsymbol{r}_t \in [-1,1]^K$ the instantaneous regret vector defined component-wise by $r_{k,t} = \widehat{\ell}_t - \ell_{k,t}$ and we define $W_t = \sum_{k=1}^K w_{k,t}$. We bound $\ln W_T$ from above and from below.

On the one hand, using the inequality $\ln(1+x) \geqslant x - x^2$ for all $x \geqslant -1/2$ (stated as Lemma 1 in Cesa-Bianchi et al. [45]), we have, for all experts k, that

$$\ln W_T \geqslant \ln w_{k,T} = \ln w_{k,0} + \sum_{t=1}^{T} \ln\big(1 + \eta_k r_{k,t}\big) \geqslant \ln w_{k,0} + \eta_k \sum_{t=1}^{T} r_{k,t} - \eta_k^2 \sum_{t=1}^{T} r_{k,t}^2 \,.$$

The last inequality holds because, by assumption, $\eta_k \leqslant 1/2$ and hence $\eta_k\big(\widehat{\ell}_t - \ell_{k,t}\big) \leqslant 1/2$ as well.

We now show by induction that, on the other hand, $W_T = W_0 = 1$ and thus that $\ln W_T = 0$. By definition of the weight update (step 3 of the algorithm), W_t equals

$$\sum_{k=1}^{K} w_{k,t} = \sum_{k=1}^{K} w_{k,t-1}\Big(1 + \eta_k r_{k,t}\Big) = W_{t-1} + \underbrace{\Big(\sum_{k=1}^{K} \eta_k w_{k,t-1}\Big)}_{=\boldsymbol{\eta}^\top \boldsymbol{w}_{t-1}} \widehat{\ell}_t - \sum_{k=1}^{K} \underbrace{\eta_k w_{k,t-1}}_{=\boldsymbol{\eta}^\top \boldsymbol{w}_{t-1}\, p_{k,t}} \ell_{k,t} \,.$$

Substituting the definition of $\boldsymbol{p}_t$ (step 1 of the algorithm), as indicated in the line above, the last two sums are seen to cancel out, leading to $W_t = W_{t-1}$. Combining the lower bound on $\ln W_T$ with its value 0 and rearranging concludes the proof. ∎

2.3. Algorithms and bound for parameters varying over time

To achieve the optimized bound (2.5), the learning parameters η_k must be tuned using preliminary knowledge of the sums $\sum_{t=1}^T \big(\widehat{\ell}_t - \ell_{k,t}\big)^2$. In this section we show how to remove this requirement, at the cost of a logarithmic factor $\ln \ln T$ only (unlike what would be obtained by working in regimes as mentioned above). We do so by having the learning rates $\eta_{k,t}$ for each expert vary with time.

2.3.1. Multiplicative updates (adaptive version of ML-Prod)

We generalize Algorithm 1 and Theorem 2.1 to Algorithm 2 and Theorem 2.3.

Theorem 2.3. *For all sequences of loss vectors $\boldsymbol{\ell}_t \in [0,1]^K$, for all rules prescribing sequences of learning rates $\eta_{k,t} \leqslant 1/2$ that, for each k, are nonincreasing in t, the cumulative loss $\sum_{t \leqslant T} \widehat{\ell}_t$ of*

Algorithm 2 is bounded by

$$\min_{1\leqslant k\leqslant K}\left\{\sum_{t=1}^{T}\ell_{k,t}+\frac{1}{\eta_{k,0}}\ln\frac{1}{w_{k,0}}+\sum_{t=1}^{T}\eta_{k,t-1}\big(\widehat{\ell}_t-\ell_{k,t}\big)^2+\frac{1}{\eta_{k,T}}\ln\left(1+\frac{1}{\mathrm{e}}\sum_{k'=1}^{K}\sum_{t=1}^{T}\left(\frac{\eta_{k',t-1}}{\eta_{k',t}}-1\right)\right)\right\}.$$

Algorithm 2: Prod with multiple adaptive learning rates (Adapt-ML-Prod)

Parameter: a rule to sequentially pick positive learning rates
Initialization: a vector $\boldsymbol{w}_0=(w_{1,0},\dots,w_{K,0})$ of nonnegative weights that sum to 1
For each round $t=1,2,\dots$
0. pick the learning rates $\eta_{k,t-1}>0$ according to the rule
1. form the mixture $\boldsymbol{p}_t$ defined component-wise by

$$p_{k,t}=\eta_{k,t-1}w_{k,t-1}\big/\boldsymbol{\eta}_{t-1}^{\top}\boldsymbol{w}_{t-1}$$

2. observe the loss vector $\boldsymbol{\ell}_t$ and incur loss $\widehat{\ell}_t=\boldsymbol{p}_t^{\top}\boldsymbol{\ell}_t$
3. for each expert k perform the update

$$w_{k,t}=\Big(w_{k,t-1}\Big(1+\eta_{k,t-1}\big(\widehat{\ell}_t-\ell_{k,t}\big)\Big)\Big)^{\frac{\eta_{k,t}}{\eta_{k,t-1}}}$$

Corollary 2.4. *With uniform initial weights $\boldsymbol{w}_0=(1/K,\dots,1/K)$ and learning rates, for $t\geqslant 1$,*

$$\eta_{k,t-1}=\min\left\{\frac{1}{2},\sqrt{\frac{\ln K}{1+\sum_{s=1}^{t-1}\big(\widehat{\ell}_s-\ell_{k,s}\big)^2}}\right\},$$

the cumulative loss of Algorithm 2 is bounded by

$$\min_{1\leqslant k\leqslant K}\left\{\sum_{t=1}^{T}\ell_{k,t}+\frac{C_{K,T}}{\sqrt{\ln K}}\sqrt{1+\sum_{t=1}^{T}\big(\widehat{\ell}_t-\ell_{k,t}\big)^2}+2C_{K,T}\right\},$$

where $C_{K,T}=3\ln K+\ln\Big(1+\dfrac{K}{2\mathrm{e}}\big(1+\ln(T+1)\big)\Big)=\mathcal{O}(\ln K+\ln\ln T)$.

This optimized corollary is the adaptive version of (2.5). Its proof is postponed to Section 2.A.3 of the additional material (and shows that meaningful bounds can be achieved as well with non-uniform initial weights). Here we only give the main ideas in the proof of Theorem 2.3. The complete argument is given in Section 2.A.2 of the additional material. We point out that the proof technique is not a routine adaptation of well-known tuning tricks such as, for example, the ones of Auer et al. [20].

Proof (sketch for Theorem 2.3) We follow the path of the proof of Theorem 2.1 and bound $\ln W_T$ from below and from above. The lower bound is easy to establish as it only relies on individual non-increasing sequences of rates, $(\eta_{k,t})_{t\geqslant 0}$ for a fixed k: the weight update (step 3 of the algorithm)

Algorithm 3: Polynomially weighted averages with multiple learning rates (ML-Poly)

Parameter: a rule to sequentially pick positive learning rates $\boldsymbol{\eta}_t = \big(\eta_{1,t}, \dots, \eta_{K,t}\big)$
Initialization: the vector of regrets with each expert $\boldsymbol{R}_0 = (0, \dots, 0)$
For each round $t = 1, 2, \dots$
0. pick the learning rates $\eta_{k,t-1}$ according to the rule
1. form the mixture $\boldsymbol{p}_t$ defined component-wise by

$$p_{k,t} = \frac{\eta_{k,t-1}\,(R_{k,t-1})_+}{\sum_{j=1}^K \eta_{j,t-1}\,(R_{j,t-1})_+}$$

where $\boldsymbol{x}_+$ denotes the vector of the nonnegative parts of the components of $\boldsymbol{x}$
2. observe the loss vector $\boldsymbol{\ell}_t$ and incur loss $\widehat{\ell}_t = \boldsymbol{p}_t^\top \boldsymbol{\ell}_t$
3. for each expert k update the regret:

$$R_{k,t} = R_{k,t-1} + \widehat{\ell}_t - \ell_{k,t}$$

was indeed tailored for it to go through. More precisely, by induction and still with the inequality $\ln(1+x) \geqslant x - x^2$ for $x \geqslant -1/2$, we get that

$$\ln W_T \;\geqslant\; \ln w_{k,T} \;\geqslant\; \frac{\eta_{k,T}}{\eta_{k,0}} \ln w_{k,0} + \eta_{k,T} \sum_{t=1}^T \big(r_{k,t} - \eta_{k,t-1} r_{k,t}^2\big)\,.$$

The difficulties arise in proving an upper bound. We proceed by induction again and aim at upper bounding W_t by W_{t-1} plus some small term. The core difficulty is that the powers $\eta_{k,t}/\eta_{k,t-1}$ in the weight update are different for each k. In the literature, time-varying parameters could previously be handled using Jensen's inequality for the function $x \mapsto x^{\alpha_t}$ with a parameter $\alpha_t = \eta_t/\eta_{t-1} \geqslant 1$ that was the same for all experts: this is, for instance, the core of the argument in the main proof of Auer et al. [20] as noticed by Györfi and Ottucsák [83] in their re-worked version of the proof. This needs to be adapted here as we have $\alpha_{k,t} = \eta_{k,t-1}/\eta_{k,t}$, which depends on k. We quantify the cost for the $\alpha_{k,t}$ not to be all equal to a single power α_t, say 1: we have $\alpha_{k,t} \geqslant 1$ but the gap to 1 should not be too large. This is why we may apply the inequality $x \leqslant x^{\alpha_{k,t}} + (\alpha_{k,t} - 1)/e$, valid for all $x > 0$ and $\alpha_{k,t} \geqslant 1$. We can then prove that

$$W_t \leqslant W_{t-1} + \frac{1}{e} \sum_{k=1}^K \left(\frac{\eta_{k,t-1}}{\eta_{k,t}} - 1 \right)\,,$$

where the second term on the right-hand side is precisely the price to pay for having different time-varying learning rates — and this price is measured by how much they vary. ∎

2.3.2. Polynomial potentials

As illustrated in Cesa-Bianchi and Lugosi [42], polynomial potentials are also useful to minimize the regret. We present here an algorithm based on them (with order $p = 2$ in the terminology of the indicated reference). Its bound has the same poor dependency on the number of experts K and on T as achieved by working in regimes (see the discussion in Section 2.2), but its analysis is simpler and

more elegant than that of Algorithm 2 (see Section 2.A.4 in the appendix; the analysis resembles the proof of Blackwell's approachability theorem). The right dependencies might be achieved by considering polynomial functions of arbitrary orders p as in Cesa-Bianchi and Lugosi [42] but we were unable to provide an accurate analysis for these values.

Theorem 2.5. *For all sequences of loss vectors $\boldsymbol{\ell}_t \in [0,1]^K$, the cumulative loss of Algorithm 3 run with learning rates*

$$\eta_{k,t-1} = \frac{1}{1 + \sum_{s=1}^{t-1} \big(\widehat{\ell}_s - \ell_{k,s}\big)^2}$$

is bounded by

$$\sum_{t=1}^{T} \widehat{\ell}_t \leqslant \min_{1 \leqslant k \leqslant K} \left\{ \sum_{t=1}^{T} \ell_{k,t} + \sqrt{K\big(1 + \ln(1+T)\big)\left(1 + \sum_{t=1}^{T} \big(\widehat{\ell}_t - \ell_{k,t}\big)^2\right)} \right\}.$$

2.4. First application: bounds with experts that report their confidences

We justify in this section why the second-order bounds exhibited in the previous sections are particularly adapted to the setting of prediction with experts that report their confidences, which was first considered* by Blum and Mansour [29]. It differs from the standard setting in that, at the start of every round t, each expert k expresses its confidence as a number $I_{k,t} \in [0,1]$. In particular, confidence $I_{k,t} = 0$ expresses that expert k is inactive (or sleeping) in round t. The learner now has to assign nonnegative weights $\boldsymbol{p}_t$, which sum up to 1, to the set $\mathcal{A}_t = \{k : I_{k,t} > 0\}$ of so-called active experts and suffers loss $\widehat{\ell}_t = \sum_{k \in \mathcal{A}_t} p_{k,t} \ell_{k,t}$. (It is assumed that, for any round t, there is at least one active expert k with $I_{k,t} > 0$, so that $\mathcal{A}_t$ is never empty.)

The main difference in prediction with confidences comes from the definition of the regret. The *confidence regret* with respect to expert k takes the numbers $I_{k,t}$ into account and is defined as

$$R^{\mathrm{c}}_{k,T} = \sum_{t=1}^{T} I_{k,t}\big(\widehat{\ell}_t - \ell_{k,t}\big)\,. \tag{2.8}$$

When $I_{k,t}$ is always 1, prediction with confidences reduces to regular prediction with expert advice, and when the confidences $I_{k,t}$ only take on the values 0 and 1, it reduces to prediction with sleeping (or specialized) experts as introduced by Blum [28] and Freund et al. [71].

Because the confidence regret scales linearly with $I_{k,t}$, one would therefore like to obtain bounds on the confidence regret that scale linearly as well. When confidences do not depend on k, this is achieved, e.g., by the bound (2.3). However, for confidences that do depend on k, the best available

* Technically, Blum and Mansour [29] decouple the confidences $I_{k,t}$, which they refer to as "time selection functions", from the experts, but as we explain in Appendix 2.C.2, the two settings are equivalent.

stated bound [29, Theorem 16] is

$$R^{\mathrm{c}}_{k,T} = \sum_{t=1}^{T} I_{k,t}\big(\widehat{\ell}_t - \ell_{k,t}\big) = O\left(\sqrt{\sum_{t\leqslant T} I_{k,t}\ell_{k,t}}\right). \tag{2.9}$$

(We rederive this bound in Appendix 2.C.2.) If, in this bound, all confidences $I_{k,t}$ are scaled down by a factor $\lambda_k \in [0,1]$, then we would like the bound to also scale down by λ_k, but instead it scales only by $\sqrt{\lambda_k}$. In the remainder of this section we will show how our new second-order bound (2.4) solves this issue via a generic reduction of the setting of prediction with confidences to the standard setting from Sections 2.1 and 2.2.

Remark 2.1. We consider the case of linear losses. The extension of our results to convex losses is immediate via the so-called gradient trick. The latter also applies in the setting of experts that report their confidences. The details were essentially provided by Devaine et al. [60] (we recall them in Section 2.C.1).

Generic reduction to the standard setting There exists a generic reduction from the setting of sleeping experts to the standard setting of prediction with expert advice [13, 103]. This reduction generalizes easily to the setting of experts that report their confidences, as we will now explain.

Given any algorithm designed for the standard setting, we run it on modified losses $\widetilde{\ell}_{k,s}$, which will be defined shortly. At round $t \geqslant 1$, the algorithm takes as inputs the past modified losses $\widetilde{\ell}_{k,s}$, where $s \leqslant t-1$, and outputs a weight vector $\widetilde{\boldsymbol{p}}_t$ on $\{1,\dots,K\}$. This vector is then used to form another weight vector $\boldsymbol{p}_t$, which has strictly positive weights only on $\mathcal{A}_t$:

$$p_{k,t} = \frac{I_{k,t}\,\widetilde{p}_{k,t}}{\sum_{k'=1}^{K} I_{k',t}\,\widetilde{p}_{k',t}} \qquad \text{for all } k. \tag{2.10}$$

This vector $\boldsymbol{p}_t$ is to be used with the experts that report their confidences. Then, the losses $\ell_{k,t}$ are observed and the modified losses are computed as follows: for all k,

$$\widetilde{\ell}_{k,t} = I_{k,t}\ell_{k,t} + (1 - I_{k,t})\widehat{\ell}_t \qquad \text{where} \quad \widehat{\ell}_t = \sum_{k\in\mathcal{A}_t} p_{k,t}\ell_{k,t}\,.$$

Proposition 2.6. *The induced confidence regret on the original losses $\ell_{k,t}$ equals the standard regret of the algorithm on the modified losses $\widetilde{\ell}_{k,t}$. In particular,*

$$I_{k,t}\big(\widehat{\ell}_t - \ell_{k,t}\big) = \sum_{i=1}^{K} \widetilde{p}_{i,t}\widetilde{\ell}_{i,t} - \widetilde{\ell}_{k,t}$$

for all rounds t and experts k.

Proof First we show that the loss in the standard setting (on the losses $\widetilde{\ell}_{k,t}$) is equal to the loss in the confidence regret setting (on the original losses $\ell_{k,t}$):

$$\sum_{k=1}^{K} \widetilde{p}_{k,t}\widetilde{\ell}_{k,t} = \sum_{k=1}^{K} \widetilde{p}_{k,t}\Big(I_{k,t}\ell_{k,t} + (1 - I_{k,t})\widehat{\ell}_t\Big)$$

$$= \sum_{k=1}^{K} \widetilde{p}_{k,t} I_{k,t} \ell_{k,t} + \widehat{\ell}_t - \left(\sum_{k=1}^{K} \widetilde{p}_{k,t} I_{k,t} \right) \widehat{\ell}_t$$
$$= \left(\sum_{k'=1}^{K} \widetilde{p}_{k',t} \, I_{k',t} \right) \sum_{k=1}^{K} p_{k,t} \ell_{k,t} + \widehat{\ell}_t - \left(\sum_{k=1}^{K} \widetilde{p}_{k,t} I_{k,t} \right) \widehat{\ell}_t = \widehat{\ell}_t.$$

The proposition now follows by subtracting $\widetilde{\ell}_{k,t}$ on both sides of the equality. ∎

Corollary 2.7. *An algorithm with a standard regret bound of the form*

$$R_{k,T} \leqslant \Xi_1 \sqrt{(\ln K) \sum_{t \leqslant T} \big(\widehat{\ell}_t - \ell_{k,t}\big)^2} + \Xi_2 \qquad \text{for all } k, \tag{2.11}$$

leads, via the generic reduction described above (and for losses $\ell_{k,t} \in [0,1]$), to an algorithm with a confidence regret bound of the form

$$R^c_{k,T} \leqslant \Xi_1 \sqrt{(\ln K) \sum_{t \leqslant T} I_{k,t}^2 \big(\widehat{\ell}_t - \ell_{k,t}\big)^2} + \Xi_2 \leqslant \Xi_1 \sqrt{(\ln K) \sum_{t \leqslant T} I_{k,t}^2} + \Xi_2 \qquad \text{for all } k. \tag{2.12}$$

We note that the second upper-bound, $\sqrt{\sum I_{k,t}^2}$, can be extracted from the proof of Theorem 11 in Chernov and Vovk [48]—but not the first one, which, combined with the techniques of Section 2.5.1, yields a bound on the confidence regret for small (excess) losses.

Comparison to the instantiation of other regret bounds We now discuss why (2.12) improves on the literature. Consider first the improved bound for small losses from the introduction, which takes the form $\Xi_3 \sqrt{\sum_t \ell_{k,t}} + \Xi_4$. This improvement does not survive the generic reduction, as the resulting confidence regret bound is

$$\Xi_3 \sqrt{\sum_{t=1}^{T} \widetilde{\ell}_{k,t}} + \Xi_4 = \Xi_3 \sqrt{\sum_{t=1}^{T} I_{k,t} \ell_{k,t} + \underbrace{\sum_{t=1}^{T} (1 - I_{k,t}) \widehat{\ell}_t}_{\text{undesirable}}} + \Xi_4,$$

which is no better than plain $\Xi'_3 \sqrt{T} + \Xi'_4$ bounds.

Alternatively, bounds (2.3) of Cesa-Bianchi et al. [45] and de Rooij et al. [58] are of the form

$$\Xi_5 \sqrt{\sum_{t=1}^{T} \sum_{k=1}^{K} p_{k,t} \big(\ell_{k,t} - \widehat{\ell}_t\big)^2} + \Xi_6,$$

uniformly over all experts k. These lead to a confidence regret bound against expert k of the form

$$\Xi_5 \sqrt{\sum_{t=1}^{T} \sum_{k=1}^{K} p_{k,t} \, I_{k,t}^2 \big(\widehat{\ell}_t - \ell_{k,t}\big)^2} + \Xi_6 \leqslant \Xi_5 \sqrt{\sum_{t=1}^{T} \sum_{k=1}^{K} p_{k,t} \, I_{k,t}^2} + \Xi_6,$$

which depends not just on the confidences of this expert k, but also on the confidences of the other experts. It therefore does not scale proportionally to the confidences of the expert k at hand.

We note that even bounds of the form (2.2), if they existed, would not be suitable either. They would indeed lead to

$$R^c_{k,T} = \mathcal{O}\left(\sqrt{\sum_{t=1}^{T}\Big(I_{k,t}\ell_{k,t} + (1-I_{k,t})\widehat{\ell}_t\Big)^2}\right),$$

which also does not scale linearly with the confidences of expert k.

2.5. Other applications: bounds in the standard setting

We now leave the setting of prediction with confidences, and detail other applications of our new second-order bound (2.4). First, in Section 2.5.1, we show that, like (2.1) and (2.3), our new bound implies an improvement over the standard bound $\mathcal{O}\big(\sqrt{\sum_t \ell_{k,t}\ln K}\big)$, which is itself already better than the worst-case bound if the losses of the reference expert are small. The key feature in our improvement is that excess losses $\ell_{k,t} - \widehat{\ell}_t$ can be considered instead of plain losses $\ell_{k,t}$. Then, in Section 2.5.2, we look at the non-adversarial setting in which losses are i.i.d., and show that our new bound implies constant regret of order $\mathcal{O}\big(\ln K\big)$.

2.5.1. Improvement for small excess losses

It is known [45, 58] that (2.3) implies a bound of the form

$$R_{k^*,T} = \mathcal{O}\left(\sqrt{\ln K \frac{L_{k^*,T}(T - L_{k^*,T})}{T}}\right), \tag{2.13}$$

where $k^* \in \arg\min_k L_{k,T}$ is the expert with smallest cumulative loss. This bound symmetrizes the standard bound for small losses described in the introduction, because it is small also if $L_{k^*,T}$ is close to T, which is useful when losses are defined in terms of gains [45].

However, if one is ready to lose symmetry, another way of improving the standard bound for small losses is to express it in terms of *excess* losses:

$$\sqrt{\ln K \sum_{t\colon \ell_{k,t}\geqslant\widehat{\ell}_t}\Big(\ell_{k,t} - \widehat{\ell}_t\Big)} \leqslant \sqrt{\ln K \sum_{t\leqslant T}\ell_{k,t}}\,,$$

where the inequality holds for nonnegative losses. As we show next, bounds of the form (2.4) indeed entail bounds of this form.

Theorem 2.8. *If the regret of an algorithm satisfies (2.11) for all sequences of loss vectors $\boldsymbol{\ell}_t \in [0,1]^K$, then it also satisfies*

$$R_{k,T} \leqslant 2\,\Xi_1\sqrt{\ln K \sum_{t\colon \ell_{k,t}\geqslant\widehat{\ell}_t}\Big(\ell_{k,t} - \widehat{\ell}_t\Big)} + \big(\Xi_2 + 2\,\Xi_1\sqrt{\Xi_2 \ln K} + 4\,\Xi_1^2 \ln K\big)\,. \tag{2.14}$$

In general, losses take values in the range $[a,b]$. To apply our methods, they therefore need to be translated by $-a$ and scaled by $1/(b-a)$ to fit the canonical range $[0,1]$. In the standard improvement for small losses, these operations remain visible in the regret bound, which becomes

$R_{k,T} = \mathcal{O}\big(\sqrt{(b-a)(L_{k,T} - Ta)\ln K}\big)$ in general. In particular, if $a < 0$, then no significant improvement over the worst-case bound $\mathcal{O}\big(\sqrt{T\ln K}\big)$ is realized. By contrast, our original second-order bound (2.11) and its corollary (2.14) both have the nice feature that translations do not affect the bound because $(\ell_{k,t} - a) - (\widehat{\ell}_t - a) = \ell_{k,t} - \widehat{\ell}_t$, so that our new improvement for small losses remains meaningful even for $a < 0$.

Proof We define the positive and the negative part of the regret with respect to an expert k by, respectively,

$$R^+_{k,T} = \sum_{t=1}^{T} \Big(\widehat{\ell}_t - \ell_{k,t}\Big)\, \mathbb{1}_{\{\ell_{k,t} \leqslant \widehat{\ell}_t\}} \qquad \text{and} \qquad R^-_{k,T} = \sum_{t=1}^{T} \Big(\ell_{k,t} - \widehat{\ell}_t\Big)\, \mathbb{1}_{\{\ell_{k,t} \geqslant \widehat{\ell}_t\}}\,.$$

The proof will rely on rephrasing the bound (2.11) in terms of $R^+_{k,T}$ and $R^-_{k,T}$ only. On the one hand, $R_{k,T} = R^+_{k,T} - R^-_{k,T}$, while, on the other hand,

$$\sqrt{\sum_{t\leqslant T}\big(\widehat{\ell}_t - \ell_{k,t}\big)^2} \leqslant \sqrt{\sum_{t\leqslant T}\Big|\widehat{\ell}_t - \ell_{k,t}\Big|} = \sqrt{R^+_{k,T} + R^-_{k,T}} \leqslant 2\sqrt{R^+_{k,T}}\,, \tag{2.15}$$

where we used $\ell_{k,t} \in [0,1]$ for the first inequality and where we assumed, with no loss of generality, that $R^+_{k,T} \geqslant R^-_{k,T}$. Indeed, if this was not the case, the regret would be negative and the bound would be true. Therefore for all experts k, substituting these (in)equalities in the initial inequality (2.11), we are left with the quadratic inequality

$$R^+_{k,T} - R^-_{k,T} \leqslant 2\Xi_1\sqrt{R^+_{k,T}\ln K} + \Xi_2\,. \tag{2.16}$$

Solving for $R^+_{k,T}$ using Lemma 2.9 below (whose proof can be found in Section 2.A.1) yields

$$\sqrt{R^+_{k,T}} \leqslant \sqrt{R^-_{k,T} + \Xi_2} + 2\Xi_1\sqrt{\ln K} \leqslant \sqrt{R^-_{k,T}} + \sqrt{\Xi_2} + 2\Xi_1\sqrt{\ln K}\,,$$

which leads to the stated bound after re-substitution into (2.16). ∎

Lemma 2.9. *Let $a, c \geqslant 0$. If $x \geqslant 0$ satisfies $x^2 \leqslant a + cx$, then $x \leqslant \sqrt{a} + c$.*

2.5.2. Stochastic (i.i.d.) losses

van Erven et al. [138] provide a specific algorithm that guarantees worst-case regret bounded by $\mathcal{O}\left(\sqrt{L_{k^\star,T}\ln K}\right)$, but at the same time is able to adapt to the non-adversarial setting with independent, identically distributed (i.i.d.) loss vectors, for which its regret is bounded by $\mathcal{O}(K)$. Theorem 2.8 already indicated that any algorithm satisfying a regret bound of the form (2.11) also achieves a worst-case bound that is at least as good as $\mathcal{O}\left(\sqrt{L_{k^\star,T}\ln K}\right)$. Here we consider i.i.d. losses that satisfy the same assumption as the one imposed by van Erven et al.:

Assumption 2.10. *The loss vectors $\boldsymbol{\ell}_t \in [0,1]^K$ are independent random variables such that there exists an action $k^\star$ and some $\alpha \in (0,1]$ for which the expected differences in loss satisfy*

$$\forall t \geqslant 1, \qquad \min_{k\neq k^\star} \mathbb{E}\big[\ell_{k,t} - \ell_{k^\star,t}\big] \geqslant \alpha\,.$$

As shown by the following theorem, any algorithm that satisfies our new second-order bound (with a constant Ξ_1 factor and a Ξ_2 factor of order $\ln K$) is guaranteed to achieve constant regret of order $\mathcal{O}(\ln K)$ under Assumption 2.10.

Theorem 2.11. *If a strategy achieves a regret bound of the form (2.11) and the loss vectors satisfy Assumption 2.10, then the expected regret for that strategy is bounded by a constant: for all T,*

$$\mathbb{E}[R_{k^\star,T}] \leqslant C(\Xi_1,\Xi_2,\alpha) \stackrel{\text{def}}{=} (\Xi_1^2 \ln K)/\alpha + \Xi_1\sqrt{(\Xi_2 \ln K)/\alpha} + \Xi_2\,;$$

while for any T and any $\delta \in (0,1)$, its regret is bounded with probability at least $1-\delta$ by

$$R_{k^\star,T} \leqslant C(\Xi_1,\Xi_2,\alpha) + \frac{6\,\Xi_1}{\alpha}\sqrt{\left(\ln\frac{1}{\delta} + \ln\left(1+\frac{1}{2\mathrm{e}}\ln\bigl(1+C(\Xi_1,\Xi_2,\alpha)/4\bigr)\right)\right)\ln K}\,.$$

By the law of large numbers, the cumulative loss of any action $k \neq k^\star$ will exceed the cumulative loss of $k^\star$ by a linear term in the order of αT, so that, for all sufficiently large T, the fact that $R_{k^\star,T}$ is bounded by a constant implies that the algorithm will have negative regret with respect to all other k.

Because we want to avoid using any special properties of the algorithm except for the fact that it satisfies (2.11), our proof of Theorem 2.11 requires a Bernstein-Freedman-type martingale concentration result [68] rather than basic applications of Hoeffding's inequality, which are sufficient in the proof of van Erven et al. [138]. However, this type of concentration inequalities is typically stated in terms of a deterministic bound M on the cumulative conditional variance $\sum V_t$. To bound the deviations by the (random) quantity $\sqrt{\sum V_t}$ instead of the deterministic $\sqrt{M}$, peeling techniques can be applied as in Cesa-Bianchi et al. [44, Corollary 16]; this leads to an additional $\sqrt{\ln T}$ factor (in case of an additive peeling) or $\sqrt{\ln\ln T}$ (in case of a geometric peeling). Here, we replace these non-constant factors by a term of order $\ln\ln\mathbb{E}\bigl[\sum V_t\bigr]$, which will be seen to be less than a constant in our case.

Theorem 2.12. *Let $(X_t)_{t\geqslant 1}$ be a martingale difference sequence with respect to some filtration $\mathcal{F}_0 \subseteq \mathcal{F}_1 \subseteq \mathcal{F}_2 \subseteq \ldots$ and let $V_t = \mathbb{E}\bigl[X_t^2 \,\big|\, \mathcal{F}_{t-1}\bigr]$ for $t \geqslant 1$. We assume that $X_t \leqslant 1$ a.s., for all $t \geqslant 1$. Then, for any $\delta \in (0,1)$ and any $T \geqslant 1$, with probability at least $1-\delta$,*

$$\sum_{t=1}^{T} X_t \leqslant 3\sqrt{\left(1+\sum_{t=1}^{T} V_t\right)\ln\frac{\gamma}{\delta} + \ln\frac{\gamma}{\delta}}\,,$$

where

$$\gamma = 1 + \frac{1}{2\mathrm{e}}\left(1+\ln\left(1+\mathbb{E}\left[\sum_{t=1}^{T} V_t\right]\right)\right).$$

Theorem 2.12 and its proof (see Section 2.A.5 for the latter) may be of independent interest, because our derivation uses new techniques that we originally developed for time-varying learning rates in the proof of Theorem 2.3. Instead of studying supermartingales of the form $\exp\bigl(\lambda\sum X_t - (\mathrm{e}-2)\lambda^2\sum V_t\bigr)$ for some constant value of λ, as is typical, we are able to consider (predictable) random

variables Λ_t, which in some sense play the role of the time-varying learning parameter η_t of the (ML-)Prod algorithm.

Proof (of Theorem 2.11) We recall the notation $r_{k,t} = \widehat{\ell}_t - \ell_{k,t}$ for the instantaneous regret. We start from $\mathcal{F}_0$, the trivial σ–algebra $\{\emptyset, \Omega\}$ (consisting of the empty set and the whole underlying probability space), and define by induction the following martingale difference sequence: for all $t \geqslant 1$,

$$Y_t = -r_{k^\star,t} + \mathbb{E}\big[r_{k^\star,t} \,\big|\, \mathcal{F}_{t-1}\big]$$

and $\mathcal{F}_t = \sigma(Y_1, \dots, Y_t)$ is the σ–algebra generated by the random variables $Y_1, \dots, Y_t$. We first bound the expectation of the regret. We note that

$$\mathbb{E}\big[r_{k^\star,t} \,\big|\, \mathcal{F}_{t-1}\big] = \sum_{k=1}^{K} p_{k,t}\, \mathbb{E}\big[\ell_{k,t} - \ell_{k^\star,t} \,\big|\, \mathcal{F}_{t-1}\big] = \sum_{k=1}^{K} p_{k,t}\, \mathbb{E}\big[\ell_{k,t} - \ell_{k^\star,t}\big] \;\geqslant \alpha(1 - p_{k^\star,t})\,, \tag{2.17}$$

while by convexity of $(\,\cdot\,)^2$,

$$r_{k^\star,t}^2 \leqslant \sum_{k=1}^{K} p_{k,t} \left(\ell_{k,t} - \ell_{k^\star,t}\right)^2 \leqslant 1 - p_{k^\star,t}\,, \tag{2.18}$$

thus,

$$W_t = \mathbb{E}\big[Y_t^2 \,\big|\, \mathcal{F}_{t-1}\big] \leqslant \mathbb{E}\big[r_{k^\star,t}^2 \,\big|\, \mathcal{F}_{t-1}\big] \leqslant 1 - p_{k^\star,t}\,. \tag{2.19}$$

Therefore, using that expectations of conditional expectations are unconditional expectations,

$$\mathbb{E}[R_{k^\star,T}] \geqslant \alpha\, \mathbb{E}[S_T] \quad \text{and} \quad \mathbb{E}\left[\sum_{t=1}^{T} r_{k^\star,t}^2\right] \leqslant \mathbb{E}[S_T] \tag{2.20}$$

where

$$S_T = \sum_{t=1}^{T} (1 - p_{k^\star,t})\,.$$

Substituting these inequalities in (2.11) using Jensen's inequality for $\sqrt{\cdot}$, we get

$$\mathbb{E}[S_T] \leqslant \frac{\Xi_1 \sqrt{\ln K}}{\alpha} \sqrt{\mathbb{E}[S_T]} + \frac{\Xi_2}{\alpha}\,.$$

Solving the quadratic inequality (see Lemma 2.9) yields $\mathbb{E}[S_T] \leqslant \big((\Xi_1\sqrt{\ln K})/\alpha + \sqrt{\Xi_2/\alpha}\big)^2$. By (2.20) this bounds $\mathbb{E}\Big[\sum_{t=1}^{T} r_{k^\star,t}^2\Big]$, which we substitute into (2.11), together with Jensen's inequality, to prove the claimed bound on the expected regret.

Now, to get the high-probability bound, we apply Theorem 2.12 to $X_t = Y_t/2 \leqslant 1$ a.s. and $V_t = W_t/4$ and use the bounds (2.17) and (2.19). We find that, with probability at least $1 - \delta$,

$$\begin{aligned} \alpha S_T &\leqslant R_{k^\star,T} + 3\sqrt{(4 + S_T)\ln(\gamma/\delta)} + 2\ln(\gamma/\delta) \\ &\leqslant R_{k^\star,T} + 3\sqrt{S_T \ln(\gamma/\delta)} + 8\ln(\gamma/\delta)\,, \end{aligned}$$

where

$$\gamma \leqslant 1 + (1/2\mathrm{e}) \left[1 + \ln\big(1 + \mathbb{E}[S_T]/4\big)\right],$$

and where we used $\sqrt{\ln(\gamma/\delta)} \geqslant 1$. Combining the bound (2.11) on the regret with (2.18) yields $R_{k^\star,T} \leqslant \Xi_1\sqrt{S_T \ln K} + \Xi_2$, so that, still with probability at least $1-\delta$,

$$\alpha S_T \leqslant \left(\Xi_1\sqrt{\ln K} + 3\sqrt{\ln(\gamma/\delta)}\right)\sqrt{S_T} + \left(8\ln(\gamma/\delta) + \Xi_2\right).$$

Solving for $\sqrt{S_T}$ with Lemma 2.9 and using that $\alpha \leqslant 1$, this implies

$$\begin{aligned}\sqrt{S_T} &\leqslant \frac{\Xi_1\sqrt{\ln K} + 3\sqrt{\ln(\gamma/\delta)}}{\alpha} + \frac{1}{\sqrt{\alpha}}\sqrt{8\ln(\gamma/\delta) + \Xi_2} \\ &\leqslant \frac{\Xi_1\sqrt{\ln K}}{\alpha} + \sqrt{\frac{\Xi_2}{\alpha}} + \frac{6}{\alpha}\sqrt{\ln\frac{\gamma}{\delta}}.\end{aligned}$$

Substitution into the (deterministic) regret bound $R_{k^\star,T} \leqslant \Xi_1\sqrt{S_T \ln K} + \Xi_2$ concludes the proof. ∎

2.6. Alternative comparison classes

We focused in the previous sections on improved regret bounds. However, the cumulative loss of an algorithm can be decomposed as the cumulative loss of the best element in a comparison class plus the regret with respect to that class. In this section we discuss alternative comparison classes, formed by combining the original base experts into meta-experts. We explain why, up to considering meta-experts, the study of these alternative classes falls under the umbrella of the setting described in Section 2.4. The meta-experts we introduce are based on partial rankings of the experts, which generalizes the *full ranking* setting introduced by Kleinberg et al. [99, 98], and their probabilistic extension to Markov chains in Section 2.6.2.

Remark 2.2. For the sake of readability we restrict our attention in this subsection to the case of sleeping experts, i.e., to the case where $I_{k,t} \in \{0,1\}$ (see Section 2.4). All definitions and results readily extend to the case of experts reporting general confidences $I_{k,t} \in [0,1]$.

Partial rankings A *partial ranking* $\sigma_{1:m} = (\sigma_1, \dots, \sigma_m)$ creates a chain of experts: if expert σ_1 is sleeping, then some other expert σ_2 is asked for advice; if the second expert is sleeping as well, then some third expert σ_3 (different from σ_1 and σ_2) is considered; and so on until expert σ_m. Since $\sigma_{1:m}$ corresponds to a ranking of a subset of size m of the experts, we call it a partial ranking. At round t, given the set of active experts $\mathcal{A}_t$, the partial ranking $\sigma_{1:m}$ is sleeping, $I_{\sigma_{1:m},t} = 0$, if $\{\sigma_1, \dots, \sigma_m\} \cap \mathcal{A}_t = \emptyset$. Otherwise, it picks the expert $k(\sigma_{1:m}, \mathcal{A}_t) = \sigma_{\min\{j\colon \sigma_j \in \mathcal{A}_t\}}$.

The interest in partial rankings lies in that they sleep less often than the base experts. This strengthens the comparison class, but also leads to increased running time for algorithms and a larger regret bound. Indeed, there are $K!/(K-m)! \leqslant K^m$ partial rankings of length m so that via the generic reduction described in Section 2.4, any standard algorithm will guarantee regret

$$\sum_{t=1}^{T} I_{\sigma_{1:m},t}\left(\widehat{\ell}_t - \ell_{k(\sigma_{1:m},\mathcal{A}_t)}\right) = \mathcal{O}\left(\sqrt{Tm\ln K}\right) \qquad \text{for all } \sigma_{1:m}.$$

A unification between two seemingly different approaches in the literature The comparison class introduced above puts under the same umbrella two approaches that previously were considered unrelated. The value $m = 1$ corresponds to the setting described in Section 2.4, while the

value $m = K$ amounts to considering full rankings of the experts as in Kleinberg et al. [99, 98]. Intermediate values of m provide a natural way to interpolate between the two settings, which trades off how well the full ranking setting is approximated against computational efficiency and magnitude of the regret.

Because of our assumption that there is always at least one active expert, all full rankings $\sigma_{1:K}$ are active in each round. This holds great appeal, but unfortunately no efficient algorithm has been found for learning full rankings, and a hardness result by Kleinberg et al. [98] rules out many of the plausible candidates. In particular, the algorithms for learning permutations by Helmbold and Warmuth [91] and Yasutake et al. [149] do not apply, because the loss of rankings cannot be expressed in the linear form they require, and Kanade et al. [96] require stochastic assumptions about the set $\mathcal{A}_t$ of active experts. It is therefore necessary to consider intermediate values of m.

2.6.1. Numerical experiments

We illustrate in this section that, indeed, considering partial rankings of not too large a length $m > 1$ leads to a reduction of the cumulative loss compared to $m = 1$. We do so in a qualitative way, by comparing the performance of two algorithms for values $m \in \{1, 2, 3, 4\}$. The cumulative (or average) losses first decrease when m increases to 2 or 3, and then level off. This is because of a trade-off between the approximation error (equal to the cumulative or average loss of the best partial ranking, which decreases as m increases) and the sequential estimation error (the regret, which increases with m). This trade-off should be reminiscent of the bias-variance trade-off in statistics.

More precisely, we consider an electricity forecasting data set analyzed by Devaine et al. [60]. The data set contains half-hourly measurements of the total electricity consumption in France from September 1, 2007 to August 31, 2008, together with several covariates, including temperature, cloud cover, wind, etc. The goal is to forecast the consumption one day ahead, for the time interval 12:00–12:30, with the help of 24 experts (consumption forecasting models) that are unavailable (sleeping) on some of the days, and prediction accuracy is measured in gigawatt (GW) by the root mean squared error (RMSE). For details about the data set we refer to Devaine et al. [60]. In the original data set the experts are too synchronized in terms of their sleeping behavior. We correct for this by introducing 6 deterministic sleeping scenarios that depend on the weather. The first two of these scenarios activate an expert only on the 70% of days with the highest (respectively lowest) cloud cover, which simulates specializing in cloudy (respectively sunny) days. The last four scenarios are constructed similarly from the attributes wind and temperature. Since we do not know the specializations of the 24 original experts, we assign each expert one of the sleeping scenarios at random, and, in addition to the expert's sleeping behavior in the original data set, we also put the expert to sleep as dictated by their assigned scenario.

We evaluate ML-Prod and the exponentially weighted average forecaster on pseudo-losses that come from the so-called gradient trick (see 2.C.1 in the supplementary material). The second forecaster is then called EG (exponentiated gradient). We did not attempt to optimize the learning rates η of these forecasters: the results below should be considered showing a qualitative behavior that we observed for all values of η we tried. Figure 2.1 shows the performance of the algorithms when run for partial rankings of lengths $m = 1, \ldots, 4$. To evaluate the effect of the randomness in the assignment of sleeping scenarios, we repeat the experiment 100 times and present a boxplot of the

resulting RMSE. We see that, for both algorithms, the RMSE follows the behavior described at the beginning of this subsection, decreasing with m till a certain value, e.g., $m = 3$ or $m = 4$ after which there is no benefit of increasing the size of the comparison class.

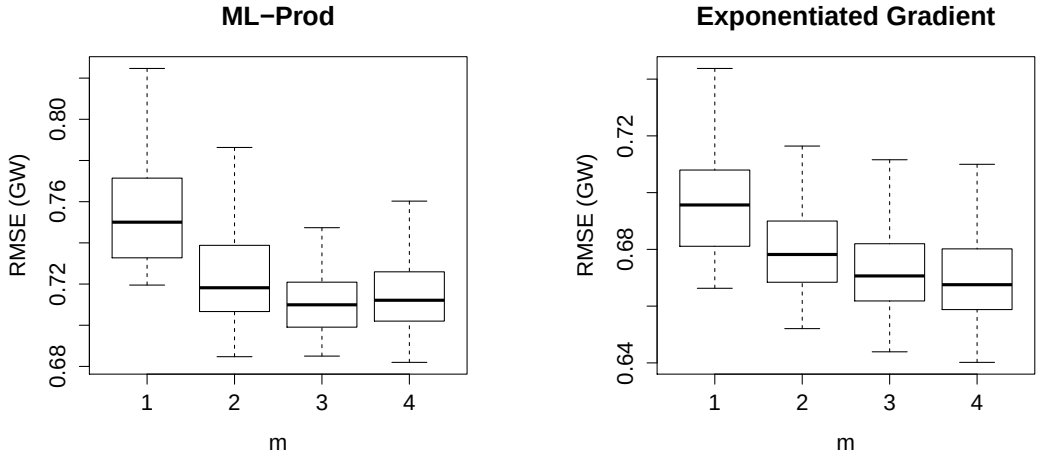


Figure 2.1.: Boxplots of the average prediction errors for two algorithms when run on partial rankings of different lengths m.

2.6.2. Markov chain modeling

Because of the chain idea indicated when introducing partial rankings it is natural to also think of alternative comparison classes based on Markov chains. The results below show that this model is equivalent to the one based on rankings. We parameterize a Markov chain by a pair $(\boldsymbol{q}, Q)$, where $\boldsymbol{q} = (q_1, \ldots, q_K)$ is a probability vector on experts and Q is (the transposition of) a $K \times K$ stochastic matrix. That is, for all experts i, the i-th column $(Q_{i,j})_j$ of Q is a probability vector. We use the Markov chain to generalize the redistribution of the prediction weights according to confidences stated in (2.10).

Let A_t denote the diagonal matrix with the vector $(I_{k,t})_k$ as its diagonal and let $\boldsymbol{I}_K$ denotes the $K \times K$ identity matrix. The matrices A_t and $\boldsymbol{I}_K - A_t$, when applied to some vector $\boldsymbol{q}$, select the components corresponding respectively to the active and sleeping experts. In general, the vector $A_t\boldsymbol{q}$ of masses on the active experts does not sum up to 1: there is some missing mass, which corresponds to the vector $(\boldsymbol{I}_K - A_t)\boldsymbol{q}$. We redistribute this missing mass using the transition matrix Q, getting in the second round $A_t\, Q(\boldsymbol{I}_K - A_t)\boldsymbol{q}$ as the redistributed mass on active experts and $(\boldsymbol{I}_K - A_t)\, Q(\boldsymbol{I}_K - A_t)\boldsymbol{q}$ as the vector of mass still not allocated. We can iterate the process for a given number r of repetitions. Depending on Q and r there may be missing mass at the end of the process, which we correct for. (We will discuss in detail under which conditions there is no missing mass.)

More formally, the missing mass after $r \geqslant 0$ iterations equals $1 - \tau_t^r$, with

$$\tau_t^r \equiv \tau_t^r(\boldsymbol{q}, Q, A_t) = 1 - \mathbb{1}^\top(\boldsymbol{I}_K - A_t)\big(Q(\boldsymbol{I}_K - A_t)\big)^r\boldsymbol{q} \ \in [0,1]\,, \tag{2.21}$$

where the vector $\mathbb{1} = (1, \ldots, 1)$ is used to sum the probabilities. The resulting distribution $\boldsymbol{q}_t^r \equiv$

$\boldsymbol{q}_t^r(\boldsymbol{q}, Q, A_t) = \big(q_{1,t}^r, \dots, q_{K,t}^r\big)$ is defined arbitrarily when $\tau_t^r = 0$ and otherwise as

$$\boldsymbol{q}_t^r = \frac{1}{\tau_t^r} \sum_{i=0}^{r} A_t\big(Q(\boldsymbol{I}_K - A_t)\big)^i \boldsymbol{q}\,. \tag{2.22}$$

In particular, $\boldsymbol{q}_t^r$ is a probability vector that is positive only on the set $\mathcal{A}_t$ of active experts.
The confidence regret against a Markov chain $(\boldsymbol{q}, Q)$ becomes

$$R^{\mathrm{c}}_{(\boldsymbol{q},Q),T} = \sum_{t=1}^{T} \tau_t^r \big(\widehat{\ell}_t - \ell_t^\top \boldsymbol{q}_t^r\big)\,.$$

Equivalence of competing with rankings and with Markov chains: finitely many repetitions The definition (2.8) corresponds to a Markov chain $(\boldsymbol{q}, Q)$ in which $\boldsymbol{q}$ is a point mass and there are no redistribution steps (i.e., $r = 0$), such that Q is not used. The following theorem shows that for any given $r \leqslant K-1$, competing with the class of all Markov chains $(\boldsymbol{q}, Q)$ with r repetitions is equally hard as competing with the class of all partial rankings of length $m \leqslant r+1$. Its proof can be found in the supplementary material.

Theorem 2.13. *Let $0 \leqslant r \leqslant K-1$. Then there exists a constant $C \equiv C(K,T)$ such that for all Markov chains $(\boldsymbol{q}, Q)$,*

$$\sum_{t=1}^{T} \tau_t^r \left(\widehat{\ell}_t - \ell_t^\top \boldsymbol{q}_t^r\right) \leqslant C\sqrt{\sum_{t=1}^{T} \tau_t^r} \tag{2.23}$$

if and only if

$$\sum_{t=1}^{T} I_{\sigma_{1:m},t} \left(\widehat{\ell}_t - \ell_{\sigma_{1:m},t}\right) \leqslant C\sqrt{\sum_{t=1}^{T} I_{\sigma_{1:m},t}} \tag{2.24}$$

for all partial rankings $\sigma_{1:m}$ of length $m \leqslant r+1$.

Extension to infinitely many repetitions In this paragraph, we restrict our attention to transition matrices Q that are irreducible. This means that, for all experts k and k', there should be a path $(s_1, s_2, \dots, s_v)$ of experts with $s_1 = k$ and $s_v = k'$ such that $Q_{s_{i+1},s_i} > 0$ for all steps $i = 1, \dots, v-1$ in the path. In this case, $\boldsymbol{I}_K - Q(\boldsymbol{I}_K - A_t)$ is invertible, with inverse

$$\big(\boldsymbol{I}_K - Q(\boldsymbol{I}_K - A_t)\big)^{-1} = \sum_{i=0}^{\infty} \big(Q(\boldsymbol{I}_K - A_t)\big)^i\,.$$

We can thus let $r \to \infty$ in (2.21) and (2.22), which leads to $\tau_t^\infty = 1$ (all the mass is redistributed now) and gives rise to the distribution $\boldsymbol{q}_t^\infty = A_t\big(\boldsymbol{I}_K - Q(\boldsymbol{I}_K - A_t)\big)^{-1}\boldsymbol{q}$.

Note that this distribution depends on Q in a non-convex way. Nevertheless, at first sight one might hope that some clever trick will allow competing with the distributions $\boldsymbol{q}_t^\infty$ anyway. The following theorem, proved in the supplementary material, shows that such a trick would have to be clever indeed, because an algorithm can compete with $\boldsymbol{q}_t^\infty$ if and only if it can compete with the comparison class of all full rankings, which we know to be computationally difficult, as discussed above.

Theorem 2.14. *There exists $C \equiv C(K,T)$ such that for all irreducible Markov chains $(\boldsymbol{q}, Q)$,*

$$\sum_{t=1}^{T} \left(\widehat{\ell}_t - \ell_t^\top \boldsymbol{q}_t^\infty \right) \leqslant C \qquad \text{if and only if} \qquad \sum_{t=1}^{T} \left(\widehat{\ell}_t - \ell_{\sigma_{1:K},t} \right) \leqslant C$$

for all full rankings $\sigma_{1:K}$.

Combining Theorem 2.14 with Theorem 2.13, we see that, essentially, the case of $r = K-1$ repetitions already captures the difficulty of dealing with $r = \infty$ repetitions.

Appendices for Chapter 2

2.A. Proofs

We gather in this appendix several facts and results whose proofs were omitted from the main body of the chapter.

2.A.1. Proof of Lemma 2.9

Solving $x^2 \leqslant a + cx$ for x, we find that

$$\frac{1}{2}c - \frac{1}{2}\sqrt{c^2+4a} \leqslant x \leqslant \frac{1}{2}c + \frac{1}{2}\sqrt{c^2+4a}\,.$$

In particular, focusing on the upper bound, we get $2x \leqslant c + \sqrt{c^2+4a} \leqslant c + \sqrt{c^2} + \sqrt{4a} = 2c + 2\sqrt{a}$, which was to be shown.

2.A.2. Proof of Theorem 2.3

The proof will rely on the following simple lemma.

Lemma 2.15. *For all $x > 0$ and all $\alpha \geqslant 1$, we have $x \leqslant x^\alpha + (\alpha-1)/\mathrm{e}$.*

Proof The inequality is straightforward when $x \geqslant 1$, so we restrict our attention to the case where $x < 1$. The function $\alpha \mapsto x^\alpha = \mathrm{e}^{\alpha \ln x}$ is convex and thus is above any tangent line. In particular, considering the value $x \ln x$ of the derivative function $\alpha \mapsto (\ln x)\,\mathrm{e}^{\alpha \ln x}$ at $\alpha = 1$, we get

$$\forall\, \alpha > 0, \qquad x^\alpha - x \geqslant (x \ln x)\,(\alpha - 1)\,.$$

Now, since we only consider $\alpha \geqslant 1$, it suffices to lower bound $x \ln x$ for the values of interest for x, namely, the ones in $(0,1)$ as indicated at the beginning of the proof. On this interval, the stated quantity is at least $-1/\mathrm{e}$, which concludes the proof. ∎

We now prove Theorem 2.3.

Proof (of Theorem 2.3) As in the proof of Theorem 2.1, we bound $\ln W_T$ from below and from above. For the lower bound, we start with $\ln W_T \geqslant \ln w_{k,T}$. We then show by induction that for all $t \geqslant 0$,

$$\ln w_{k,t} \geqslant \eta_{k,t} \sum_{s=1}^{t} \Big(r_{k,s} - \eta_{k,s-1} r_{k,s}^2 \Big) + \frac{\eta_{k,t}}{\eta_{k,0}} \ln w_{k,0} \,,$$

where $r_{k,s} = \widehat{\ell}_s - \ell_{k,s}$ denotes the instantaneous regret with respect to expert k. The inequality is trivial for $t = 0$. If it holds at a given round t, then by the weight update (step 3 of the algorithm),

$$\begin{aligned}
\ln w_{k,t+1} &= \frac{\eta_{k,t+1}}{\eta_{k,t}} \Big(\ln w_{k,t} + \ln\Big(1 + \eta_{k,t} r_{k,t+1}\Big) \Big) \\
&\geqslant \frac{\eta_{k,t+1}}{\eta_{k,t}} \left(\frac{\eta_{k,t}}{\eta_{k,0}} \ln w_{k,0} + \eta_{k,t} \sum_{s=1}^{t} \Big(r_{k,s} - \eta_{k,s-1} r_{k,s}^2 \Big) \right) + \frac{\eta_{k,t+1}}{\eta_{k,t}} \big(\eta_{k,t} r_{k,t+1} - \eta_{k,t}^2 r_{k,t+1}^2 \big) \\
&= \eta_{k,t+1} \sum_{s=1}^{t+1} \Big(r_{k,s} - \eta_{k,s-1} r_{k,s}^2 \Big) + \frac{\eta_{k,t+1}}{\eta_{k,0}} \ln w_{k,0} \,,
\end{aligned}$$

where the inequality comes from the induction hypothesis and from the inequality $\ln(1+x) \geqslant x - x^2$ for all $x \geqslant -1/2$ already used in the proof of Theorem 2.1.

We now bound from above $\ln W_T$, or equivalently, W_T itself. We show by induction that for all $t \geqslant 0$,

$$W_t \leqslant 1 + \frac{1}{\mathrm{e}} \sum_{k=1}^{K} \sum_{s=1}^{t} \left(\frac{\eta_{k,s-1}}{\eta_{k,s}} - 1 \right).$$

The inequality is trivial for $t = 0$. To show that if the property holds for some $t \geqslant 0$ it also holds for $t+1$, we prove that

$$W_{t+1} \leqslant W_t + \frac{1}{\mathrm{e}} \sum_{k=1}^{K} \left(\frac{\eta_{k,t}}{\eta_{k,t+1}} - 1 \right). \tag{2.25}$$

Indeed, since $x \leqslant x^\alpha + (\alpha - 1)/\mathrm{e}$ for all $x > 0$ and $\alpha \geqslant 1$ (see Lemma 2.15), we have, for each expert k,

$$w_{k,t+1} \leqslant \big(w_{k,t+1}\big)^{\frac{\eta_{k,t}}{\eta_{k,t+1}}} + \frac{1}{\mathrm{e}} \left(\frac{\eta_{k,t}}{\eta_{k,t+1}} - 1 \right); \tag{2.26}$$

we used here $x = w_{k,t+1}$ and $\alpha = \eta_{k,t}/\eta_{k,t+1}$, which is larger than 1 because of the assumption that the learning rates are nonincreasing in t for each k. Now, by definition of the weight update (step 3 of the algorithm),

$$\sum_{k=1}^{K} \big(w_{k,t+1}\big)^{\frac{\eta_{k,t}}{\eta_{k,t+1}}} = \sum_{k=1}^{K} w_{k,t} \big(1 + \eta_{k,t} r_{k,t+1}\big) = W_t \,,$$

where the second inequality follows from the same argument as in the last display of the proof of Theorem 2.1, by using that $\eta_{k,t} w_{k,t}$ is proportional to $p_{k,t+1}$. Summing (2.26) over k thus yields (2.25) as desired.

Finally, combining the upper and lower bounds on $\ln W_T$ and rearranging leads to the inequality of Theorem 2.3. ∎

2.A.3. Proof of Corollary 2.4

The following lemma will be useful.

Lemma 2.16. *Let $a_0 > 0$ and $a_1, \dots, a_m \in [0,1]$ be real numbers and let $f : (0, +\infty) \to [0, +\infty)$ be a nonincreasing function. Then*

$$\sum_{i=1}^{m} a_i \, f\big(a_0 + \ldots + a_{i-1}\big) \leqslant f(a_0) + \int_{a_0}^{a_0 + a_1 + \ldots + a_m} f(u) \, \mathrm{d}u \, .$$

Proof Abbreviating $s_i = a_0 + \ldots + a_i$ for $i = 0, \dots, m$, we find that

$$\begin{aligned} \sum_{i=1}^{m} a_i \, f(s_{i-1}) &= \sum_{i=1}^{m} a_i \, f(s_i) + \sum_{i=1}^{m} a_i \big(f(s_{i-1}) - f(s_i)\big) \\ &\leqslant \sum_{i=1}^{m} a_i \, f(s_i) + \sum_{i=1}^{m} \big(f(s_{i-1}) - f(s_i)\big) \leqslant \sum_{i=1}^{m} a_i \, f(s_i) + f(s_0), \end{aligned}$$

where the first inequality follows because $f(s_{i-1}) \geqslant f(s_i)$ and $a_i \leqslant 1$ for $i \geqslant 1$, while the second inequality stems from a telescoping argument together with the fact that $f(s_m) \geqslant 0$. Using that f is nonincreasing together with $s_i - s_{i-1} = a_i$ for $i \geqslant 1$, we further have

$$a_i \, f(s_i) = \int_{s_{i-1}}^{s_i} f(s_i) \, \mathrm{d}y \leqslant \int_{s_{i-1}}^{s_i} f(y) \, \mathrm{d}y \, .$$

Substituting this bound in the above inequality completes the proof. ∎

We will be slightly more general and take

$$\eta_{k,t} = \min \left\{ \frac{1}{2}, \, \sqrt{\frac{\gamma_k}{1 + \sum_{s=1}^{t} r_{k,s}^2}} \right\}$$

for some constant $\gamma_k > 0$ to be defined by the analysis.

Because of the choice of nonincreasing learning rates, the first inequality of Theorem 2.3 holds true, and the regret $R_{k,t}$ is upper-bounded by

$$\frac{1}{\eta_{k,0}} \ln \frac{1}{w_{k,0}} + \frac{1}{\eta_{k,T}} \ln \left(1 + \frac{1}{\mathrm{e}} \sum_{k'=1}^{K} \underbrace{\sum_{t=1}^{T} \left(\frac{\eta_{k',t-1}}{\eta_{k',t}} - 1 \right)}_{\text{first term}} \right) + \underbrace{\sum_{t=1}^{T} \eta_{k,t-1} r_{k,t}^2}_{\text{second term}} \, . \tag{2.27}$$

For the first term in (2.27), we note that for each k' and $t \geqslant 1$ one of three possibilities must hold, all depending on which of the inequalities in $\eta_{k',t} \leqslant \eta_{k',t-1} \leqslant 1/2$ are equalities or strict inequalities. More precisely, either $\eta_{k',t} = \eta_{k',t-1} = 1/2$; or

$$\sqrt{\frac{\gamma_{k'}}{1 + \sum_{s=1}^{t} r_{k',s}^2}} = \eta_{k',t} < \eta_{k',t-1} = \frac{1}{2} \leqslant \sqrt{\frac{\gamma_{k'}}{1 + \sum_{s=1}^{t-1} r_{k',s}^2}} \, ;$$

or $\eta_{k',t} \leqslant \eta_{k',t-1} < 1/2$. In all cases, the ratios $\eta_{k',t-1}/\eta_{k',t} - 1$ can be bounded as follows:

$$\sum_{t=1}^{T}\left(\frac{\eta_{k',t-1}}{\eta_{k',t}} - 1\right) \leqslant \sum_{t=1}^{T}\left(\sqrt{\frac{1+\sum_{s=1}^{t} r_{k',t}^2}{1+\sum_{s=1}^{t-1} r_{k',t}^2}} - 1\right)$$
$$= \sum_{t=1}^{T}\left(\sqrt{1+\frac{r_{k',t}^2}{1+\sum_{s=1}^{t-1} r_{k',s}^2}} - 1\right) \leqslant \frac{1}{2}\sum_{t=1}^{T}\frac{r_{k',t}^2}{1+\sum_{s=1}^{t-1} r_{k',s}^2}\,, \quad (2.28)$$

where we used, for the second inequality, that $g(1+z) \leqslant g(1) + z\,g'(1)$ for $z \geqslant 0$ for any concave function g, in particular the square root. We apply Lemma 2.16 with $f(x) = 1/x$ to further bound the sum in (2.28), which gives

$$\sum_{t=1}^{T}\frac{r_{k',t}^2}{1+\sum_{s=1}^{t-1} r_{k',s}^2} \leqslant 1 + \ln\left(1+\sum_{t=1}^{T} r_{k',t}^2\right) - \cancel{\ln(1)} \leqslant 1 + \ln(T+1)\,. \quad (2.29)$$

For the second term in (2.27), we write

$$\sum_{t=1}^{T}\eta_{k,t-1} r_{k,t}^2 \leqslant \sqrt{\gamma_k}\sum_{t=1}^{T}\frac{r_{k,t}^2}{\sqrt{1+\sum_{s=1}^{t-1} r_{k,s}^2}}\,.$$

We apply Lemma 2.16 again, with $f(x) = 1/\sqrt{x}$, and get

$$\sum_{t=1}^{T}\frac{r_{k,t}^2}{\sqrt{1+\sum_{s=1}^{t-1} r_{k,s}^2}} \leqslant \underbrace{1-2\sqrt{1}}_{\leqslant 0} + 2\sqrt{\left(1+\sum_{t=1}^{T} r_{k,t}^2\right)}\,. \quad (2.30)$$

We may now get back to (2.27). Substituting the obtained bounds on its first and second terms, and using $\eta_{k,0} \geqslant \eta_{k,T}$, we find it is no greater than

$$\frac{1}{\eta_{k,T}}\left(\ln\frac{1}{w_{k,0}} + B_{K,T}\right) + 2\sqrt{\gamma_k\left(1+\sum_{t=1}^{T} r_{k,t}^2\right)}\,, \quad (2.31)$$

where $B_{K,T} = \ln\Big(1 + \frac{K}{2\mathrm{e}}\big(1+\ln(T+1)\big)\Big)$.

Now if $\sqrt{1+\sum_{t=1}^{T} r_{k,t}^2} > 2\sqrt{\gamma_k}$ then $\eta_{k,T} < 1/2$ and (2.31) is bounded by

$$\sqrt{1+\sum_{t=1}^{T} r_{k,t}^2}\left(2\sqrt{\gamma_k} + \frac{\ln\frac{1}{w_{k,0}} + B_{K,T}}{\sqrt{\gamma_k}}\right).$$

Alternatively, if $\sqrt{1+\sum_{t=1}^{T} r_{k,t}^2} \leqslant 2\sqrt{\gamma_k}$, then $\eta_{k,T} = 1/2$ and (2.31) does not exceed

$$2\ln\frac{1}{w_{k,0}} + 2B_{K,T} + 4\gamma_k.$$

In either case, (2.31) is smaller than the sum of the latter two bounds, from which the corollary follows upon taking $\gamma_k = \ln(1/w_{k,0}) = \ln K$.

2.A.4. Proof of Theorem 2.5

The proof has a geometric flavor—the same as in the proof of the approachability theorem [27]. With a diagonal matrix $D = \operatorname{diag}(d_1, \dots, d_K)$, with positive on-diagonal elements d_i, we associate an inner product and a norm as follows:

$$\forall\, \boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^K, \qquad \langle \boldsymbol{x},\, \boldsymbol{y} \rangle_D = \boldsymbol{x}^\top D \boldsymbol{y} \qquad \text{and} \qquad \|\boldsymbol{x}\|_D = \sqrt{\boldsymbol{x}^\top D \boldsymbol{x}}\,.$$

We denote by π_D the projection on $\mathbb{R}^K_-$ under the norm $\|\cdot\|_D$. It turns out that this projection is independent of the considered matrix D satisfying the constraints described above: it equals

$$\forall \boldsymbol{x} \in \mathbb{R}^K, \qquad \pi_D(\boldsymbol{x}) = \boldsymbol{x} - \boldsymbol{x}_+\,,$$

where we recall that $\boldsymbol{x}_+$ denotes the vector whose components are the nonnegative parts of the components of $\boldsymbol{x}$. This entails that for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^K$

$$\|(\boldsymbol{x}+\boldsymbol{y})_+\|_D^2 = \|\boldsymbol{x}+\boldsymbol{y}-\pi_D(\boldsymbol{x}+\boldsymbol{y})\|_D^2 \leqslant \|\boldsymbol{x}+\boldsymbol{y}-\pi_D(\boldsymbol{x})\|_D^2 = \|\boldsymbol{x}_+ + \boldsymbol{y}\|_D^2\,. \tag{2.32}$$

Now, we consider, for each instance $t \geqslant 1$, the diagonal matrix $D_t = \operatorname{diag}(\eta_{1,t}, \dots, \eta_{K,t})$, with positive elements on the diagonal. As all sequences $(\eta_{k,t})_{t \geqslant 0}$ are non-increasing for a fixed k, we have, for all $t \geqslant 1$, that

$$\forall \boldsymbol{x} \in \mathbb{R}^K, \qquad \|\boldsymbol{x}\|_{D_t} \leqslant \|\boldsymbol{x}\|_{D_{t-1}}\,. \tag{2.33}$$

This entails that

$$\big\|(\boldsymbol{R}_t)_+\big\|_{D_t} \leqslant \big\|(\boldsymbol{R}_t)_+\big\|_{D_{t-1}} = \big\|(\boldsymbol{R}_{t-1}+\boldsymbol{r}_t)_+\big\|_{D_{t-1}} \leqslant \big\|(\boldsymbol{R}_{t-1})_+ + \boldsymbol{r}_t\big\|_{D_{t-1}}\,, \tag{2.34}$$

where we denoted by $\boldsymbol{r}_t$ the vector $(r_{k,t})_{1 \leqslant k \leqslant K}$ of the instantaneous regrets and where we applied (2.32). Taking squares and developing the squared norm, we get

$$\big\|(\boldsymbol{R}_t)_+\big\|_{D_t}^2 \leqslant \big\|(\boldsymbol{R}_{t-1})_+\big\|_{D_{t-1}}^2 + \|\boldsymbol{r}_t\|_{D_{t-1}}^2 + 2\,\boldsymbol{r}_t^\top D_{t-1}\,(\boldsymbol{R}_{t-1})_+\,. \tag{2.35}$$

But the inner product equals

$$2\,\boldsymbol{r}_t^\top D_{t-1}\,(\boldsymbol{R}_{t-1})_+ = 2\sum_{k=1}^K \eta_{k,t-1}\,(R_{k,t-1})_+\,r_{k,t} = 2\,\boldsymbol{\eta}_{t-1}^\top\,(\boldsymbol{R}_{t-1})_+ \underbrace{\sum_{k=1}^K p_{k,t} r_{k,t}}_{=0} = 0\,,$$

where the last but one equality follows from step 1 of the algorithm.

Hence (2.35) entails $\big\|(\boldsymbol{R}_t)_+\big\|_{D_t}^2 - \big\|(\boldsymbol{R}_{t-1})_+\big\|_{D_{t-1}}^2 \leqslant \|\boldsymbol{r}_t\|_{D_{t-1}}^2$, which, summing over all rounds $t \geqslant 1$, leads to

$$\big\|(\boldsymbol{R}_T)_+\big\|_{D_T}^2 - \cancel{\big\|(\boldsymbol{R}_0)_+\big\|_{D_0}^2} \leqslant \sum_{t=1}^T \|\boldsymbol{r}_t\|_{D_{t-1}}^2 = \sum_{t=1}^T \sum_{k=1}^K \eta_{k,t-1} r_{k,t}^2$$

$$= \sum_{k=1}^{K} \sum_{t=1}^{T} \frac{r_{k,t}^2}{1 + \sum_{s=1}^{t-1} r_{k,s}^2} \leqslant K\big(1 + \ln(1+T)\big)\,, \quad (2.36)$$

where the last equality follows from substituting the value of $\eta_{k,t-1}$ and the last inequality was proved in (2.29). Finally, (2.36) implies that, for any expert $k = 1, \dots, K$,

$$\eta_{k,T}\,(R_{k,T})_+^2 \leqslant \big\|\,(\boldsymbol{R}_T)_+\,\big\|_{D_T}^2 \leqslant K\big(1 + \ln(1+T)\big)\,,$$

so that

$$R_{k,T} \leqslant \sqrt{K\big(1 + \ln(1+T)\big)\,\eta_{k,T}^{-1}}\,.$$

The proof is concluded by substituting the value of $\eta_{k,T}$.

2.A.5. Proof of Theorem 2.12 (variation on the Bernstein–Freedman inequality)

Let $\varphi : \mathbb{R} \to \mathbb{R}$ and $\varphi : \mathbb{R} \to \mathbb{R}$ be defined by $\varphi(\lambda) = \mathrm{e}^\lambda - \lambda - 1$ on the one hand, $\varphi(0) = 1/2$ and $\varphi(\lambda) = \varphi(\lambda)/\lambda^2$ on the other hand. The following lemma is due to Freedman [68, Lemmas 1.3a and 3.1]. Note that we are only proving a one-sided inequality and do not require the lower bound on X imposed in the mentioned reference.

Lemma 2.17. — Freedman [68]. *The function φ is increasing. As a consequence, for all bounded random variables $X \leqslant 1$ a.s., for all σ–algebras $\mathcal{F}$ such that $\mathbb{E}[X \,|\, \mathcal{F}] = 0$ a.s., and for all nonnegative random variables $\Lambda \geqslant 0$ that are $\mathcal{F}$–measurable,*

$$\mathbb{E}\big[\,\exp(\Lambda X)\,\big|\,\mathcal{F}\big] \leqslant \exp\big(\varphi(\Lambda)\,V\big) \quad \text{a.s.} \qquad \textit{where} \qquad V = \mathbb{E}\big[X^2\,\big|\,\mathcal{F}\big] = \mathrm{Var}(X \,|\, \mathcal{F})\,.$$

Proof That φ is increasing follows from a function study. Using this, we get $\varphi(\Lambda X) \leqslant \varphi(\Lambda)$, which can be rewritten as

$$\mathrm{e}^{\Lambda X} - \Lambda X - 1 \leqslant \varphi(\Lambda)\,X^2\,.$$

By integrating both sides with respect to $\mathbb{E}[\,\cdot\,|\,\mathcal{F}]$ and by using that Λ is $\mathcal{F}$–measurable and that $\mathbb{E}[X \,|\, \mathcal{F}] = 0$ a.s., we get

$$\mathbb{E}\big[\,\exp(\Lambda X)\,\big|\,\mathcal{F}\big] \leqslant 1 + \varphi(\Lambda)\,V \qquad \text{a.s.}$$

The proof is concluded by the inequality $1 + u \leqslant \mathrm{e}^u$, valid for all $u \in \mathbb{R}$. ∎

Proof (of Theorem 2.12) We fix $x > 0$. The analysis relies on a non-increasing sequence of random variables $1 \geqslant \Lambda_1 \geqslant \Lambda_2 \geqslant \ldots > 0$ such that each Λ_t is $\mathcal{F}_{t-1}$–measurable. More precisely, we pick

$$\Lambda_t = \min\left\{1,\ \sqrt{\frac{x}{1 + \sum_{s=1}^{t-1} V_s}}\right\}$$

and choose, by convention, $\Lambda_0 = 1$. We define, for all $t \geqslant 1$,

$$H_t = \exp\left(\Lambda_t \sum_{s=1}^{t} \left(X_s - \frac{\varphi(\Lambda_s)}{\Lambda_s} V_s\right)\right)$$

and $H_0 = 1$. Below we will apply Markov's inequality to H_t and we therefore need to bound $\mathbb{E}[H_t]$. By Lemma 2.17,

$$\mathbb{E}\Big[\exp\big(\Lambda_t X_t - \varphi(\Lambda_t) V_t\big) \,\Big|\, \mathcal{F}_{t-1}\Big] \leqslant 1 \qquad \text{a.s.},$$

so that for all $t \geqslant 1$,

$$\begin{aligned}\mathbb{E}[H_t] &= \mathbb{E}\big[\mathbb{E}[H_t \,|\, \mathcal{F}_{t-1}]\big] \\ &= \mathbb{E}\left[\exp\left(\Lambda_t \sum_{s=1}^{t-1} \left(X_s - \frac{\varphi(\Lambda_s)}{\Lambda_s} V_s\right)\right) \mathbb{E}\Big[\exp\big(\Lambda_t X_t - \varphi(\Lambda_t) V_t\big) \,\Big|\, \mathcal{F}_{t-1}\Big]\right] \leqslant \mathbb{E}\Big[H_{t-1}^{\Lambda_t/\Lambda_{t-1}}\Big]\,.\end{aligned}$$

Applying Lemma 2.15 with $\alpha = \Lambda_{t-1}/\Lambda_t$, this can be further bounded as

$$\mathbb{E}[H_t] \leqslant \mathbb{E}\Big[H_{t-1}^{\Lambda_t/\Lambda_{t-1}}\Big] \leqslant \mathbb{E}[H_{t-1}] + \frac{1}{\mathrm{e}}\,\mathbb{E}\left[\frac{\Lambda_{t-1}}{\Lambda_t} - 1\right].$$

Proceeding by induction and given that $H_0 = 1$, we get, for all $T \geqslant 1$,

$$\mathbb{E}[H_T] \leqslant 1 + \frac{1}{\mathrm{e}} \sum_{t=1}^{T} \mathbb{E}\left[\frac{\Lambda_{t-1}}{\Lambda_t} - 1\right].$$

The same argument and calculations as in (2.28) and (2.29) finally show that

$$\mathbb{E}[H_T] \leqslant \underbrace{1 + \frac{1}{2\mathrm{e}}\,\mathbb{E}\left[1 + \ln\left(1 + \sum_{t=1}^{T} V_t\right)\right]}_{\leqslant \gamma};$$

that the left-hand side is less than γ follows from Jensen's inequality for the logarithm. An application of Markov's inequality entails that

$$\mathbb{P}\left\{\sum_{t=1}^{T} X_t \geqslant \frac{x}{\Lambda_T} + \sum_{t=1}^{T} \frac{\varphi(\Lambda_t)}{\Lambda_t} V_t\right\} = \mathbb{P}\big\{\ln H_T \geqslant x\big\} = \mathbb{P}\big\{H_T \geqslant \mathrm{e}^x\big\} \leqslant \mathbb{E}[H_T]\,\mathrm{e}^{-x}\,.$$

To conclude the proof, it thus suffices to take x such that

$$\mathbb{E}[H_T]\,\mathrm{e}^{-x} \leqslant \delta\,, \qquad \text{e.g.,} \qquad x = \ln\frac{\gamma}{\delta}$$

and to show that

$$\frac{x}{\Lambda_T} + \sum_{t=1}^{T} \frac{\varphi(\Lambda_t)}{\Lambda_t} V_t \leqslant 3\sqrt{x}\sqrt{1 + \sum_{t=1}^{T} V_t} + x\,, \tag{2.37}$$

which we do next.

Because $\Lambda_t \leqslant 1$ and φ is increasing, we have $\varphi(\Lambda_t) = \varphi(\Lambda_t)/\Lambda_t^2 \leqslant \varphi(1) = \mathrm{e} - 2 \leqslant 1$. Therefore,

$$\sum_{t=1}^{T} \frac{\varphi(\Lambda_t)}{\Lambda_t} V_t \leqslant \sum_{t=1}^{T} \Lambda_t V_t \leqslant \sum_{t=1}^{T} \sqrt{\frac{x}{1+\sum_{s=1}^{t-1} V_s}\, V_t} \leqslant 2\sqrt{x}\sqrt{1+\sum_{t=1}^{T} V_t}\,,$$

where we used for the second inequality the definition of Λ_t as a minimum and applied the same argument as in (2.30) for the third one. It only remains to bound x/Λ_T, for which we use the upper bound (again, following from the definition of Λ_T as a minimum)

$$\frac{x}{\Lambda_T} \leqslant x + \sqrt{x}\sqrt{1+\sum_{t=1}^{T-1} V_t} \leqslant x + \sqrt{x}\sqrt{1+\sum_{t=1}^{T} V_t}\,.$$

Putting things together, we proved (2.37), which concludes this proof. ∎

2.B. Proof of Theorems 2.13 and 2.14

We recall that for the sake of simplicity, the theorem was stated and will thus be proved only in the case when the confidences are binary, $I_{k,t} \in \{0,1\}$.

Proof (of Theorem 2.13) We prove first that (2.23) implies (2.24). To this end, let $\sigma_{1:m}$ be any partial ranking of length $m \leqslant r+1$. Now choose $\boldsymbol{q}$ and Q such that $q_{\sigma_1} = 1$ one the one hand, $Q_{\sigma_{i+1}\sigma_i} = 1$ for $i = 1, \dots, m-1$ and $Q_{\sigma_m\sigma_m} = 1$ on the other hand. (The columns of Q not indexed by a σ_i can be chosen arbitrarily.) If one of the experts in $\sigma_{1:m}$ is active, we then have that $\tau_r^t = 1$ and, denoting by σ_i the first active expert in the sequence $(\sigma_1, \dots, \sigma_m)$, that $\boldsymbol{q}_t^r$ is a point mass on σ_i. Alternatively, if none of the experts in $\sigma_{1:m}$ is active, we have $\tau_r^t = 0$. So in both cases $\tau_t^r = I_{\sigma_{1:m},t}$ and when the latter equals 1, the losses are equal, $\ell_{(\boldsymbol{q},Q),t} = \ell_{\sigma_{1:m},t}$. The implication (2.23) $\Rightarrow$ (2.24) follows from these equalities.

For the converse implication, we first note that partial rankings of lengths $m \leqslant r+1$ can be re-parameterized in a redundant way, in terms of paths $s_{0:r} = (s_0, \dots, s_r)$ of length $r+1$. Here each step $s_i \in \{1, \dots, K\}$ in the path indicates an expert, with repetitions allowed (unlike in the case of partial rankings). A path $s_{0:r}$ induces canonically a partial ranking $\sigma(s_{0:r})$ by deleting all repetitions, i.e., all steps s_j such that $s_j = s_i$ for some $i < j$. This partial ranking $\sigma(s_{0:r})$ is of length $m \leqslant r+1$.

Now we set the confidence $I_{s_{0:r},t}$ and, in the case when the latter equals 1, the action $k(s_{0:r}, A_t)$ and the loss $\ell_{s_{0:r},t}$ of the path $s_{0:r}$ as, respectively, the confidence, the action and the loss of the induced partial ranking $\sigma(s_{0:r})$. The bound (2.24) entails that

$$\sum_{t=1}^{T} I_{s_{0:r},t} \left(\widehat{\ell}_t - \ell_{s_{0:r},t}\right) \leqslant C\sqrt{\sum_{t=1}^{T} I_{s_{0:r},t}} \qquad \text{for all paths } s_{0:r}. \tag{2.38}$$

We now introduce convex weights $M_{s_{0:r}}$ on the paths of length $r+1$ that induce weights on partial rankings which, when aggregated (and normalized), lead to the convex weights (2.22) on the experts.

Namely, we consider the weights

$$M_{s_{0:r}} = \left(\prod_{j=1}^{r} Q_{s_{j+1},s_j} \right) q_{s_0} \, ; \tag{2.39}$$

they form a convex weight vector on all paths of length $r+1$. By (2.38) and Jensen's inequality

$$\begin{aligned} \sum_{s_{0:r}} M_{s_{0:r}} \sum_{t=1}^{T} I_{s_{0:r},t} \left(\widehat{\ell}_t - \ell_{s_{0:r},t} \right) &\leqslant C \sum_{s_{0:r}} M_{s_{0:r}} \sqrt{\sum_{t=1}^{T} I_{s_{0:r},t}} \\ &\leqslant C \sqrt{\sum_{t=1}^{T} \sum_{s_{0:r}} M_{s_{0:r}} I_{s_{0:r},t}} \, . \end{aligned} \tag{2.40}$$

We now explain why for all experts k' and all A_t,

$$\sum_{s_{0:r}} M_{s_{0:r}} \, I_{s_{0:r},t} \, \mathbb{1}_{\{k(s_{0:r},A_t)=k'\}} = \tau_t^r \, q_{k',t}^r \, . \tag{2.41}$$

This will ensure that both the left and right-hand sides of (2.40) equal the ones of (2.23). Indeed, the very definition (2.22) indicates that for all k,

$$\tau_t^r q_{k,t}^r = \mathbb{1}_{\{k \in A_t\}} \left(q_k + \sum_{i=1}^{r} \sum_{b_1,\ldots,b_i \notin A_t} Q_{k,b_i} \left(\prod_{j=1}^{i-1} Q_{b_{j+1},b_j} \right) q_{b_1} \right) \, ;$$

or, in words, we look at all paths starting from some sleeping expert b_1, going through a chain $b_1, \ldots, b_i$ of sleeping experts, and ending up with k as the first active expert. This is exactly what we have as well in (2.39) and (2.41). ∎

Proof (of Theorem 2.14) We first show that irreducible Markov chains induce all full rankings. Indeed, let $\pi_{1:K} = (\pi_1, \ldots, \pi_K)$ be a ranking of the experts. Then if we take $q_{\pi_1} = 1$ and any Q such that $Q_{\pi_{i+1}\pi_i} = 1$ for $i = 1, \ldots, K-1$, we find that $\boldsymbol{q}_t^\infty$ is a point mass on the first active expert in $\pi_{1:K}$, so that the loss of $(\boldsymbol{q}, Q)$ is equal to the loss of $\pi_{1:K}$ for all rounds t. This implies that any uniform bound $C(T)$ on the regret for irreducible Markov chains $(\boldsymbol{q}, Q)$ implies the same bound on the regret for full rankings.

For the converse implication, we note that any irreducible Markov chain $(\boldsymbol{q}, Q)$ induces a distribution on full rankings via a construction similar to the construction in the proof of Theorem 2.13. ∎

2.C. Additional material for Section 2.4

2.C.1. The gradient trick — how to deal with convex losses via a reduction to the linear case

Freund et al. [71] consider the case of convex aggregation in the context of sleeping experts and design several strategies, each specific to a convex loss function. Devaine *et al.* explain in Section 2.2 of Devaine et al. [60] how to reduce the problem of convex aggregation to linear losses, via the

standard gradient trick (see, e.g., Section 2.5 of Cesa-Bianchi and Lugosi [43]), and could exhibit a unified analysis of all the strategies of Freund et al. [71].

We briefly recall this reduction here and note that it also holds for the generalization from sleeping experts to experts that report their confidences.

Setting and notation (see Freund et al. [71]). Suppose the experts predict by choosing an element $x_{k,t}$ from a convex set $\mathcal{X} \subseteq \mathbb{R}^d$ of possible predictions, and that their losses at round t are determined by a convex and differentiable function f_t, such that $\ell_{k,t} = f_t(x_{k,t})$. At each step, the forecaster chooses a weight vector $\boldsymbol{p}_t$ over $\mathcal{A}_t$ and aggregates the expert forecasts as $\widehat{x}_t = \sum_{k\in\mathcal{A}_t} p_{k,t} x_{k,t}$, with resulting loss $\widehat{\ell}_t = f_t\big(\widehat{x}_t\big)$.

Instead of competing with the best expert, we may wish to compete with the best fixed convex combination of experts in the following way. At round t, a weight vector $\boldsymbol{q}$ with nonnegative components that sum to 1 aggregates the forecasts according to

$$x_{\boldsymbol{q},t} = \sum_{k\in\mathcal{A}_t} \frac{q_k I_{k,t}}{Q_t(\boldsymbol{q})} x_{k,t} \quad \text{where} \quad Q_t(\boldsymbol{q}) = \sum_{k'\in\mathcal{A}_t} q_{k'} I_{k',t}\,;$$

the resulting loss equals $f_t(x_{\boldsymbol{q},t})$.

The regret with respect to a given $\boldsymbol{q}$ is then defined as

$$R^{\mathrm{c}}_{\boldsymbol{q},T} = \sum_{t=1}^{T} Q_t(\boldsymbol{q})\big(\widehat{\ell}_t - f_t(x_{\boldsymbol{q},t})\big),$$

which reduces to the confidence regret of Section 2.4 if $\boldsymbol{q}$ is a point-mass.

The reduction to linear losses. We may now reduce this problem to the case of linear losses considered in Sections 2.1 and 2.4. We do so by resorting to the so-called gradient trick. We denote by ∇f_t the gradient of f_t and introduce pseudo-losses $\ell'_{k,t} = \nabla f_t(\widehat{x}_t)^\top x_{k,t}$ for all experts k. We denote by $\boldsymbol{\ell}'_t$ the vector of the pseudo-losses. Because of the convexity inequality

$$f_t(y) \geqslant f_t(x) + \nabla f_t(x)^\top (y - x) \qquad \forall\, x, y \in \mathcal{X},$$

we have

$$\begin{aligned}\max_{\boldsymbol{q}} R^{\mathrm{c}}_{\boldsymbol{q},T} &= \max_{\boldsymbol{q}} \sum_{t=1}^{T} Q_t(\boldsymbol{q})\Big(f_t\big(\widehat{x}_t\big) - f_t(x_{\boldsymbol{q},t})\Big) \leqslant \max_{\boldsymbol{q}} \sum_{t=1}^{T} Q_t(\boldsymbol{q})\Big(\nabla f_t\big(\widehat{x}_t\big)^\top\big(\widehat{x}_t - x_{\boldsymbol{q},t}\big)\Big) \\ &= \max_{\boldsymbol{q}} \sum_{t=1}^{T} Q_t(\boldsymbol{q}) \left(\boldsymbol{p}_t^\top \boldsymbol{\ell}'_t - \sum_{k\in\mathcal{A}_t} \frac{q_k I_{k,t}}{Q_t(\boldsymbol{q})} \ell'_{k,t}\right).\end{aligned}$$

Substituting the definition of $Q_t(\boldsymbol{q})$, we get that $\max_{\boldsymbol{q}} R^{\mathrm{c}}_{\boldsymbol{q},T}$ is upper-bounded by

$$\max_{\boldsymbol{q}} \sum_{t=1}^{T} \left(\sum_{k'\in\mathcal{A}_t} q_{k'} I_{k',t} \boldsymbol{p}_t^\top \boldsymbol{\ell}'_t - \sum_{k\in\mathcal{A}_t} q_k I_{k,t} \ell'_{k,t}\right) = \max_{\boldsymbol{q}} \sum_{k=1}^{K} q_k \underbrace{\sum_{t=1}^{T} I_{k,t}\big(\boldsymbol{p}_t^\top \boldsymbol{\ell}'_t - \ell'_{k,t}\big)}_{R^{\mathrm{c}}_{k,T}} = \max_k R^{\mathrm{c}}_{k,T}\,,$$

where the first equality is because $I_{k,t} = 0$ for $k \notin \mathcal{A}_t$, and the last equality follows by linearity of the expression in $\boldsymbol{q}$.

Therefore, any regret bound for the linear prediction setting with losses $\ell'_{k,t}$ implies a bound for competing with the best convex combination of expert predictions in the original convex setting with losses $\ell_{k,t}$.

2.C.2. Hedge with multiple learning rates for experts that report their confidences

In this section, we discuss another algorithm with multiple learning rates, which was proposed by Blum and Mansour [29]. We slightly adjust its presentation so that it fits the setting of this chapter: Blum and Mansour always consider all combinations of K experts and M confidences $\mathcal{M} = \{I_{1,t}, \dots, I_{M,t}\}$, which they refer to as "time selection functions." These enter as $\sqrt{\ln(KM)}$ in their Theorem 16. To recover their setting, we can consider M copies of each expert, one for each "time selection function", so that our effective number of experts becomes KM and we also obtain a $\sqrt{\ln(KM)}$ factor in our bounds. Converse, to couple time selection functions and experts, like we do, Blum and Mansour (see their Section 6) simply take $\mathcal{M} = \{I_{1,t}, \dots, I_{K,t}\}$, so that $M = K$ and hence they obtain $\sqrt{\ln(KM)} = \sqrt{2 \ln K}$, which is the same as our $\sqrt{\ln K}$ up to a factor $\sqrt{2}$. Thus the two settings are essentially equivalent.

Theorem 2.18. — Adapted from Blum and Mansour [29]. *For all K-tuples $\boldsymbol{\eta}$ of positive learning rates in $[0,1]^K$, for all sequences of loss vectors $\boldsymbol{\ell}_t \in [0,1]^K$ and of confidences $(I_{1,t}, \dots, I_{K,t}) \in [0,1]^K$, the confidence regret of Algorithm 4 is bounded as follows: for all experts $k \in \{1, \dots, K\}$,*

$$R^c_{k,t} = \sum_{t=1}^{T} I_{k,t}\big(\widehat{\ell}_t - \ell_{k,t}\big) \leqslant \frac{\ln(1/w_{k,0})}{\eta_k} + (\mathrm{e}-1)\eta_k \sum_{t=1}^{T} I_{k,t}\ell_{k,t} + (\mathrm{e}-1)\ln(1/w_{k,0})\,. \tag{2.42}$$

Optimizing (2.42) with respect to η_k, for all $k \in \{1, \dots, K\}$ we obtain

$$R^c_{k,t} \leqslant \quad 2\sqrt{(\mathrm{e}-1)\sum_{t=1}^{T} I_{k,t}\ell_{k,t}\ln(1/w_{k,0}) + (\mathrm{e}-1)\ln(1/w_{k,0})}\,,$$

as indicated in (2.9).

Remark 2.3. Although, in practice, we cannot optimize (2.42) with respect to η_k, it is possible to tune the parameters $\eta_{k,t}$ of MLC-Hedge sequentially using a similar approach as in the proof of Theorem 2.3, at the same small $\mathcal{O}(\ln \ln T)$ cost. (We believe that there is some cost here for this tuning; the bound stated in Section 6 of Blum and Mansour [29] only considers the case of an optimization in hindsight and alludes to the possibility of some online tuning, not working out the details.)

The analysis of MLC-Hedge suggests that its bound can probably not be obtained in a two-step procedure, by first exhibiting a bound in the standard setting for some ML-Hedge algorithm and then applying the generic reduction from Section 2.4 to get an algorithm suited for experts that report their confidences. Thus, the approach taken in the main body of this chapter seems more general.

Algorithm 4: Hedge with multiple learning rates for experts reporting confidences (MLC-Hedge)

Parameters: a vector $\boldsymbol{\eta} = (\eta_1, \dots, \eta_K)$ of learning rates
Initialization: a vector $\boldsymbol{w}_0 = (w_{1,0}, \dots, w_{K,0})$ of nonnegative weights that sum to 1
For each round $t = 1, 2, \dots$

1. form the mixture $\boldsymbol{p}_t$ defined by

$$p_{k,t} = \frac{I_{k,t}\big(1 - \mathrm{e}^{-\eta_k}\big) w_{k,t-1}}{\sum_{k'=1}^{K} I_{k',t}\big(1 - \mathrm{e}^{-\eta_{k'}}\big) w_{k',t-1}}$$

2. observe the loss vector $\boldsymbol{\ell}_t$ and incur loss $\widehat{\ell}_t = \boldsymbol{p}_t^\top \boldsymbol{\ell}_t$
3. for each expert k perform the update

$$w_{k,t} = w_{k,t-1} \exp\Big(\eta_k\, I_{k,t}\big(\mathrm{e}^{-\eta_k}\widehat{\ell}_t - \ell_{k,t}\big)\Big)$$

Proof (of Theorem 2.18) As in the proof of Theorem 2.1, we upper and lower bound $\ln W_T$. For all k, the lower bound $W_T \geqslant w_{k,T}$ together with the fact that

$$w_{k,T} = w_{k,0} \exp\left(\eta_k \sum_{t=1}^{T} I_{k,t}\big(\mathrm{e}^{-\eta_k}\widehat{\ell}_t - \ell_{k,t}\big)\right),$$

yields

$$\sum_{t=1}^{T} I_{k,t}\big(\mathrm{e}^{-\eta_k}\widehat{\ell}_t - \ell_{k,t}\big) \leqslant \frac{\ln W_T + \ln(1/w_{k,0})}{\eta_k},$$

which entails

$$\sum_{t=1}^{T} I_{k,t}\widehat{\ell}_t \leqslant \left(\sum_{t=1}^{T} I_{k,t}\ell_{k,t} + \frac{\ln W_T + \ln(1/w_{k,0})}{\eta_k}\right) \mathrm{e}^{\eta_k}. \tag{2.43}$$

We now upper-bound W_T by $W_0 = 1$. To do so, we show that $W_{t+1} \leqslant W_t$ for all $t \geqslant 0$. By the weight update (step 3 of the algorithm), $W_t = \sum_{k=1}^{K} w_{k,t}$ equals

$$\sum_{k=1}^{K} w_{k,t-1} \exp\Big(\eta_k\, I_{k,t}\big(\mathrm{e}^{-\eta_k}\widehat{\ell}_t - \ell_{k,t}\big)\Big) = \sum_{k=1}^{K} w_{k,t-1} \exp\big(-\eta_k I_{k,t}\ell_{k,t}\big) \exp\big(\eta_k \mathrm{e}^{-\eta_k} I_{k,t}\widehat{\ell}_t\big). \tag{2.44}$$

For all $\eta \in \mathbb{R}$, the function $x \in [0,1] \mapsto \mathrm{e}^{\eta x}$ is convex, and therefore,

$$\mathrm{e}^{\eta x} \leqslant (1-x)\mathrm{e}^0 + x\,\mathrm{e}^{\eta} = 1 - \big(1 - \mathrm{e}^{\eta}\big)x.$$

In particular for all $\eta > 0$ and for all $x \in [0,1]$

$$\mathrm{e}^{-\eta x} \leqslant 1 - \big(1 - \mathrm{e}^{-\eta}\big)x \qquad \text{and} \qquad \mathrm{e}^{\eta x} \leqslant 1 + \big(1 - \mathrm{e}^{-\eta}\big)\mathrm{e}^{\eta} x. \tag{2.45}$$

Bounding (2.44) further with the two inequalities stated above, we get

$$
\begin{aligned}
W_t &\leqslant \sum_{k=1}^{K} w_{k,t-1}\Big(1-\big(1-\mathrm{e}^{-\eta_k}\big)I_{k,t}\ell_{k,t}\Big)\Big(1+\big(1-\mathrm{e}^{-\eta_k}\big)\cancel{\mathrm{e}^{\eta_k}\mathrm{e}^{-\eta_k}}I_{k,t}\widehat{\ell}_t\Big)\\
&\leqslant \sum_{k=1}^{K} w_{k,t-1}\Big(1+\big(1-\mathrm{e}^{-\eta_k}\big)I_{k,t}\big(\widehat{\ell}_t-\ell_{k,t}\big)\Big)\\
&= W_{t-1}+\sum_{k=1}^{K}\underbrace{w_{k,t-1}\big(1-\mathrm{e}^{-\eta_k}\big)I_{k,t}}_{=Z_t\,p_{k,t}}\big(\widehat{\ell}_t-\ell_{k,t}\big)\\
&= W_{t-1}+Z_t\underbrace{\left(\widehat{\ell}_t-\sum_{k=1}^{K}p_{k,t}\ell_{k,t}\right)}_{=0} = W_{t-1}\,,
\end{aligned}
$$

where $Z_t=\sum_{k'=1}^{K} w_{k',t-1}\big(1-\mathrm{e}^{-\eta_{k'}}\big)I_{k',t}$ and the first equality is by the definition of $\boldsymbol{p}_t$ (step 1 of the algorithm). This concludes the induction.

We then get from (2.43)

$$
\sum_{t=1}^{T} I_{k,t}\widehat{\ell}_t \leqslant \left(\sum_{t=1}^{T} I_{k,t}\ell_{k,t}+\frac{\ln(1/w_{k,0})}{\eta_k}\right)\mathrm{e}^{\eta_k}\,.
$$

The claim of the theorem follows by the upper bound $\mathrm{e}^{\eta_k}\leqslant 1+(\mathrm{e}-1)\eta_k$ for $\eta_k\in[0,1]$, which is a special case of (2.45). ∎

3

Mirror descent meets fixed share (and feels no regret)

Mirror descent with an entropic regularizer is known to achieve shifting regret bounds that are logarithmic in the dimension. This is done using either a carefully designed projection or by a weight sharing technique. Via a novel unified analysis, we show that these two approaches deliver essentially equivalent bounds on a notion of regret generalizing shifting, adaptive, discounted, and other related regrets. Our analysis also captures and extends the generalized weight sharing technique of Bousquet and Warmuth, and can be refined in several ways, including improvements for small losses and adaptive tuning of parameters.

Contents

NOTA: This chapter is a joint work with Nicolò Cesa-Bianchi, Gábor Lugosi, and Gilles Stoltz. It is based on the conference paper [3] presented at Nips 2012.

3.1. Introduction

Online convex optimization is a sequential prediction paradigm in which, at each time step, the learner chooses an element from a fixed convex set $\mathcal{S}$ and then is given access to a convex loss function defined on the same set. The value of the function on the chosen element is the learner's loss. Many problems such as prediction with expert advice, sequential investment, and online regression/classification can be viewed as special cases of this general framework. Online learning algorithms are designed to minimize the regret. The standard notion of regret is the difference between the learner's cumulative loss and the cumulative loss of the single best element in $\mathcal{S}$. A much harder criterion to minimize is shifting regret, which is defined as the difference between the learner's cumulative loss and the cumulative loss of an arbitrary sequence of elements in $\mathcal{S}$. Shifting regret bounds are typically expressed in terms of the *shift*, a notion of regularity measuring the length of the trajectory in $\mathcal{S}$ described by the comparison sequence (i.e., the sequence of elements against which the regret is evaluated). In online convex optimization, shifting regret bounds for convex subsets $\mathcal{S} \subseteq \mathbb{R}^d$ are obtained for the projected online mirror descent (or follow-the-regularized-leader) algorithm. In this case the shift is typically computed in terms of the p-norm of the difference of consecutive elements in the comparison sequence —see Herbster and Warmuth [93], Zinkevich [151] and Cesa-Bianchi and Lugosi [43].

We focus on the important special case when $\mathcal{S}$ is the simplex. In Herbster and Warmuth [93] shifting bounds are shown for projected mirror descent with entropic regularizers using a 1-norm to measure the shift.* When the comparison sequence is restricted to the corners of the simplex (which is the setting of prediction with expert advice), then the shift is naturally defined to be the number of times the trajectory moves to a different corner. This problem is often called "tracking the best expert" —see, e.g., Herbster and Warmuth [92], Vovk [141], Herbster and Warmuth [93], Bousquet and Warmuth [32], Duchi et al. [62], and it is well known that exponential weights with weight sharing, which corresponds to the fixed-share algorithm of Herbster and Warmuth [92], achieves a good shifting bound in this setting. In Bousquet and Warmuth [32] the authors introduce a generalization of the fixed-share algorithm, and prove various shifting bounds for any trajectory in the simplex. However, their bounds are expressed using a quantity that corresponds to a proper shift only for trajectories on the simplex corners.

In this chapter we offer a unified analysis of mirror descent, fixed share, and the generalized fixed share of Bousquet and Warmuth [32] for the setting of online convex optimization in the simplex. Our bounds are expressed in terms of a notion of shift based on the total variation distance. Our analysis relies on a generalized notion of shifting regret which includes, as special cases, related notions of regret such as adaptive regret, discounted regret, and regret with time-selection functions. Perhaps surprisingly, we show that projected mirror descent and fixed share achieve essentially the same generalized regret bound. Finally, we show that widespread techniques in online learning, such as improvements for small losses and adaptive tuning of parameters, are all easily captured by our analysis.

*Similar 1-norm shifting bounds can also be proven using the analysis of Zinkevich [151]. However, without using entropic regularizers it is not clear how to achieve a logarithmic dependence on the dimension, which is one of the advantages of working in the simplex.

3.2. Preliminaries

For simplicity, we derive our results in the setting of online linear optimization. As we show in the supplementary material, these results can be easily extended to the more general setting of online convex optimization through a standard linearization step.

Online linear optimization may be cast as a repeated game between the *forecaster* and the *environment* as follows. We use Δ_d to denote the simplex $\big\{\boldsymbol{q} \in [0,1]^d \,:\, \|\boldsymbol{q}\|_1 = 1\big\}$.

For each round $t = 1, \ldots, T$,
1. Forecaster chooses $\widehat{\boldsymbol{p}}_t = (\widehat{p}_{1,t}, \ldots, \widehat{p}_{d,t}) \in \Delta_d$
2. Environment chooses a loss vector $\ell_t = (\ell_{1,t}, \ldots, \ell_{d,t}) \in [0,1]^d$
3. Forecaster suffers loss $\widehat{\boldsymbol{p}}_t^\top \ell_t$.

Figure 3.1.: Online linear optimization in the simplex.

The goal of the forecaster is to minimize the accumulated loss, e.g., $\widehat{L}_T = \sum_{t=1}^T \widehat{\boldsymbol{p}}_t^\top \ell_t$. In the now classical problem of prediction with expert advice, the goal of the forecaster is to compete with the best fixed component (often called "expert") chosen in hindsight, that is, with $\min_{i=1,\ldots,T} \sum_{t=1}^T \ell_{i,t}$; or even to compete with a richer class of *sequences* of components. In Section 3.3 we state more specifically the goals considered in this chapter.

We start by introducing our main algorithmic tool, described in Figure 5, a share algorithm whose formulation generalizes the seemingly unrelated formulations of the algorithms studied in Herbster and Warmuth [92, 93], Bousquet and Warmuth [32]. It is parameterized by the "mixing functions" $\psi_t : [0,1]^{td} \to \Delta_d$ for $t \geqslant 2$ that assign probabilities to past "pre-weights" as defined below. In all examples discussed in this chapter, these mixing functions are quite simple, but working with such a general model makes the main ideas more transparent. We then provide a simple lemma that serves as the starting point[†] for analyzing different instances of this generalized share algorithm.

Algorithm 5: The generalized share algorithm.

Parameters: learning rate $\eta > 0$ and mixing functions ψ_t for $t \geqslant 2$

Initialization: $\widehat{\boldsymbol{p}}_1 = \mathbf{v}_1 = (1/d, \ldots, 1/d)$

For each round $t = 1, \ldots, T$,
1. Predict $\widehat{\boldsymbol{p}}_t$;
2. Observe loss $\ell_t \in [0,1]^d$;
3. [loss update] **For** each $j = 1, \ldots, d$ define

$$v_{j,t+1} = \frac{\widehat{p}_{j,t}\, e^{-\eta\, \ell_{j,t}}}{\sum_{i=1}^d \widehat{p}_{i,t}\, e^{-\eta\, \ell_{i,t}}} \quad \text{the current pre-weights,} \quad \text{and } \boldsymbol{v}_{t+1} = (v_{1,t+1}, \ldots, v_{d,t+1});$$

$$\underline{V}_{t+1} = \big[v_{i,s}\big]_{1 \leqslant i \leqslant d,\, 1 \leqslant s \leqslant t+1} \quad \text{the } d \times (t+1) \text{ matrix of all past and current pre-weights;}$$

4. [shared update] Define $\widehat{\boldsymbol{p}}_{t+1} = \psi_{t+1}\big(\underline{V}_{t+1}\big)$.

[†]We only deal with linear losses in this chapter. However, it is straightforward that for sequences of η–exp-concave loss functions, the additional term $\eta/8$ in the bound is no longer needed.

Lemma 3.1. *For all $t \geqslant 1$ and for all $\boldsymbol{q}_t \in \Delta_d$, Algorithm 5 satisfies*

$$\big(\widehat{\boldsymbol{p}}_t - \boldsymbol{q}_t\big)^\top \ell_t \leqslant \frac{1}{\eta} \sum_{i=1}^{d} q_{i,t} \ln \frac{v_{i,t+1}}{\widehat{p}_{i,t}} + \frac{\eta}{8} \,.$$

Proof By Hoeffding's inequality (see, e.g., Cesa-Bianchi and Lugosi [43, Section A.1.1]),

$$\sum_{j=1}^{d} \widehat{p}_{j,t}\, \ell_{j,t} \leqslant -\frac{1}{\eta} \ln \left(\sum_{j=1}^{d} \widehat{p}_{j,t}\, e^{-\eta\, \ell_{j,t}} \right) + \frac{\eta}{8} \,. \tag{3.1}$$

By definition of $v_{i,t+1}$, for all $i = 1, \dots, d$ we then have $\sum_{j=1}^{d} \widehat{p}_{j,t}\, e^{-\eta\, \ell_{j,t}} = \widehat{p}_{i,t}\, e^{-\eta\, \ell_{i,t}} / v_{i,t+1}$, which implies $\widehat{\boldsymbol{p}}_t^\top \ell_t \leqslant \ell_{i,t} + (1/\eta) \ln(v_{i,t+1}/\widehat{p}_{i,t}) + \eta/8$. The proof is concluded by taking a convex aggregation with respect to $\boldsymbol{q}_t$. ∎

3.3. A generalized shifting regret for the simplex

We now introduce a generalized notion of shifting regret which unifies and generalizes the notions of discounted regret (see Cesa-Bianchi and Lugosi [43, Section 2.11]), adaptive regret (see Hazan and Seshadhri [90]), and shifting regret (see Zinkevich [151]). For a fixed horizon T, a sequence of discount factors $\beta_{t,T} \geqslant 0$ for $t = 1, \dots, T$ assigns varying weights to the instantaneous losses suffered at each round. We compare the total loss of the forecaster with the loss of an arbitrary sequence of vectors $\boldsymbol{q}_1, \dots, \boldsymbol{q}_T$ in the simplex Δ_d. Our goal is to bound the regret

$$\sum_{t=1}^{T} \beta_{t,T}\, \widehat{\boldsymbol{p}}_t^\top \ell_t - \sum_{t=1}^{T} \beta_{t,T}\, \boldsymbol{q}_t^\top \ell_t$$

in terms of the "regularity" of the comparison sequence $\boldsymbol{q}_1, \dots, \boldsymbol{q}_T$ and of the variations of the discounting weights $\beta_{t,T}$. By setting $\boldsymbol{u}_t = \beta_{t,T}\, \boldsymbol{q}_t^\top \in \mathbb{R}_+^d$, we can rephrase the above regret as

$$\sum_{t=1}^{T} \|\boldsymbol{u}_t\|_1\, \widehat{\boldsymbol{p}}_t^\top \ell_t - \sum_{t=1}^{T} \boldsymbol{u}_t^\top \ell_t \,. \tag{3.2}$$

In the literature on tracking the best expert Herbster and Warmuth [92], Vovk [141], Herbster and Warmuth [93], Bousquet and Warmuth [32], the regularity of the sequence $\boldsymbol{u}_1, \dots, \boldsymbol{u}_T$ is measured as the number of times $\boldsymbol{u}_t \neq \boldsymbol{u}_{t+1}$. We introduce the following regularity measure

$$m(\boldsymbol{u}_1^T) = \sum_{t=2}^{T} D_{\mathrm{TV}}(\boldsymbol{u}_t, \boldsymbol{u}_{t-1}) \tag{3.3}$$

where for $\boldsymbol{x} = (x_1, \dots, x_d), \boldsymbol{y} = (y_1, \dots, y_d) \in \mathbb{R}_+^d$, we define $D_{\mathrm{TV}}(\boldsymbol{x}, \boldsymbol{y}) = \sum_{x_i \geqslant y_i} (x_i - y_i)$. Note that when $\boldsymbol{x}, \boldsymbol{y} \in \Delta_d$, we recover the total variation distance $D_{\mathrm{TV}}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{2} \|\boldsymbol{x} - \boldsymbol{y}\|_1$, while for general $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}_+^d$, the quantity $D_{\mathrm{TV}}(\boldsymbol{x}, \boldsymbol{y})$ is not necessarily symmetric and is always bounded by $\|\boldsymbol{x} - \boldsymbol{y}\|_1$. The traditional shifting regret of Herbster and Warmuth [92], Vovk [141], Herbster and Warmuth [93], Bousquet and Warmuth [32] is obtained from (3.2) when all $\boldsymbol{u}_t$ are such that $\|\boldsymbol{u}_t\|_1 = 1$.

3.4. Projected update

The shifting variant of the EG algorithm analyzed in Herbster and Warmuth [93] is a special case of the generalized share algorithm in which the function ψ_{t+1} performs a projection of the pre-weights on the convex set $\Delta_d^\alpha = [\alpha/d, 1]^d \cap \Delta_d$. Here $\alpha \in (0,1)$ is a fixed parameter. We can prove (using techniques similar to the ones shown in the next section—see the supplementary material) the following bound which generalizes Herbster and Warmuth [93, Theorem 16].

Theorem 3.2. *For all $T \geqslant 1$, for all sequences $\ell_1, \dots, \ell_t \in [0,1]^d$ of loss vectors, and for all $\boldsymbol{u}_1, \dots, \boldsymbol{u}_T \in \mathbb{R}_+^d$, if Algorithm 1 is run with the above update, then*

$$\sum_{t=1}^{T} \|\boldsymbol{u}_t\|_1 \, \widehat{\boldsymbol{p}}_t^\top \ell_t - \sum_{t=1}^{T} \boldsymbol{u}_t^\top \ell_t \leqslant \frac{\|\boldsymbol{u}_1\|_1 \ln d}{\eta} + \frac{m(\boldsymbol{u}_1^T)}{\eta} \ln \frac{d}{\alpha} + \Big(\frac{\eta}{8} + \alpha\Big) \sum_{t=1}^{T} \|\boldsymbol{u}_t\|_1 \ . \tag{3.4}$$

This bound can be optimized by a proper tuning of α and η parameters. We show a similarly tuned (and slightly better) bound in Corollary 3.4.

3.5. Fixed-share update

Next, we consider a different instance of the generalized share algorithm corresponding to the update

$$\widehat{p}_{j,t+1} = \sum_{i=1}^{d} \Big(\frac{\alpha}{d} + (1-\alpha)\mathbb{1}_{i=j}\Big) v_{i,t+1} = \frac{\alpha}{d} + (1-\alpha) v_{j,t+1} \,, \qquad 0 \leqslant \alpha \leqslant 1 \tag{3.5}$$

Despite seemingly different statements, this update in Algorithm 5 can be seen to lead *exactly* to the fixed-share algorithm of Herbster and Warmuth [92] for prediction with expert advice. We now show that this update delivers a bound on the regret almost equivalent to (though slightly better than) that achieved by projection on the subset Δ_d^α of the simplex.

Theorem 3.3. *With the above update, for all $T \geqslant 1$, for all sequences $\ell_1, \dots, \ell_T$ of loss vectors $\ell_t \in [0,1]^d$, and for all $\boldsymbol{u}_1, \dots, \boldsymbol{u}_T \in \mathbb{R}_+^d$,*

$$\sum_{t=1}^{T} \|\boldsymbol{u}_t\|_1 \, \widehat{\boldsymbol{p}}_t^\top \ell_t - \sum_{t=1}^{T} \boldsymbol{u}_t^\top \ell_t \leqslant \frac{\|\boldsymbol{u}_1\|_1 \ln d}{\eta} + \frac{\eta}{8} \sum_{t=1}^{T} \|\boldsymbol{u}_t\|_1 + \frac{m(\boldsymbol{u}_1^T)}{\eta} \ln \frac{d}{\alpha} + \frac{\sum_{t=2}^{T} \|\boldsymbol{u}_t\|_1 - m(\boldsymbol{u}_1^T)}{\eta} \ln \frac{1}{1-\alpha} \ .$$

Note that if we only consider vectors of the form $\boldsymbol{u}_t = \boldsymbol{q}_t = (0, \dots, 0, 1, 0, \dots, 0)$ then $m(\boldsymbol{q}_1^T)$ corresponds to the number of times $\boldsymbol{q}_{t+1} \neq \boldsymbol{q}_t$ in the sequence $\boldsymbol{q}_1^T$. We thus recover Herbster and Warmuth [92, Theorem 1] and Bousquet and Warmuth [32, Lemma 6] from the much more general Theorem 3.3.

The fixed-share forecaster does not need to "know" anything in advance about the sequence of the norms $\|\boldsymbol{u}_t\|$ for the bound above to be valid. Of course, in order to minimize the obtained upper

bound, the tuning parameters α, η need to be optimized and their values will depend on the maximal values of $m(\boldsymbol{u}_1^T)$ and $\sum_{t=1}^T \|\boldsymbol{u}_t\|_1$ for the sequences one wishes to compete against. This is illustrated in the following corollary, whose proof is omitted. Therein, $h(x) = -x\ln x - (1-x)\ln(1-x)$ denotes the binary entropy function for $x \in [0,1]$. We recall[‡] that $h(x) \leqslant x\ln(e/x)$ for $x \in [0,1]$.

Corollary 3.4. *Suppose Algorithm 5 is run with the update (3.5). Let $m_0 > 0$ and $U_0 > 0$. For all $T \geqslant 1$, for all sequences $\ell_1, \dots, \ell_T$ of loss vectors $\ell_t \in [0,1]^d$, and for all sequences $\boldsymbol{u}_1, \dots, \boldsymbol{u}_T \in \mathbb{R}_+^d$ with $\|\boldsymbol{u}_1\|_1 + m(\boldsymbol{u}_1^T) \leqslant m_0$ and $\sum_{t=1}^T \|\boldsymbol{u}_t\|_1 \leqslant U_0$,*

$$\sum_{t=1}^T \|\boldsymbol{u}_t\|_1 \, \widehat{\boldsymbol{p}}_t^\top \ell_t - \sum_{t=1}^T \boldsymbol{u}_t^\top \ell_t \leqslant \sqrt{\frac{U_0}{2}\left(m_0 \ln d + U_0\, h\!\left(\frac{m_0}{U_0}\right)\right)} \leqslant \sqrt{\frac{U_0\, m_0}{2}\left(\ln d + \ln\left(\frac{e\, U_0}{m_0}\right)\right)}$$

whenever η and α are optimally chosen in terms of m_0 and U_0.

Proof (of Theorem 3.3) Applying Lemma 3.1 with $\boldsymbol{q}_t = \boldsymbol{u}_t / \|\boldsymbol{u}_t\|_1$, and multiplying by $\|\boldsymbol{u}_t\|_1$, we get for all $t \geqslant 1$ and $\boldsymbol{u}_t \in \mathbb{R}_+^d$

$$\|\boldsymbol{u}_t\|_1 \, \widehat{\boldsymbol{p}}_t^\top \ell_t - \boldsymbol{u}_t^\top \ell_t \leqslant \frac{1}{\eta} \sum_{i=1}^d u_{i,t} \ln \frac{v_{i,t+1}}{\widehat{p}_{i,t}} + \frac{\eta}{8} \|\boldsymbol{u}_t\|_1 \ . \tag{3.6}$$

We now examine

$$\sum_{i=1}^d u_{i,t} \ln \frac{v_{i,t+1}}{\widehat{p}_{i,t}} = \sum_{i=1}^d \left(u_{i,t} \ln \frac{1}{\widehat{p}_{i,t}} - u_{i,t-1} \ln \frac{1}{v_{i,t}}\right) + \sum_{i=1}^d \left(u_{i,t-1} \ln \frac{1}{v_{i,t}} - u_{i,t} \ln \frac{1}{v_{i,t+1}}\right) \ . \tag{3.7}$$

For the first term on the right-hand side, we have

$$\begin{aligned} \sum_{i=1}^d \left(u_{i,t} \ln \frac{1}{\widehat{p}_{i,t}} - u_{i,t-1} \ln \frac{1}{v_{i,t}}\right) &= \sum_{i\,:\,u_{i,t} \geqslant u_{i,t-1}} \left((u_{i,t} - u_{i,t-1}) \ln \frac{1}{\widehat{p}_{i,t}} + u_{i,t-1} \ln \frac{v_{i,t}}{\widehat{p}_{i,t}}\right) \\ &\quad + \sum_{i\,:\,u_{i,t} < u_{i,t-1}} \Bigg(\underbrace{(u_{i,t} - u_{i,t-1}) \ln \frac{1}{v_{i,t}}}_{\leqslant 0} + u_{i,t} \ln \frac{v_{i,t}}{\widehat{p}_{i,t}}\Bigg). \end{aligned} \tag{3.8}$$

In view of the update (3.5), we have $1/\widehat{p}_{i,t} \leqslant d/\alpha$ and $v_{i,t}/\widehat{p}_{i,t} \leqslant 1/(1-\alpha)$. Substituting in (3.8), we get

$$\begin{aligned} &\sum_{i=1}^d \left(u_{i,t} \ln \frac{1}{\widehat{p}_{i,t}} - u_{i,t-1} \ln \frac{1}{v_{i,t}}\right) \\ &\qquad \leqslant \sum_{i\,:\,u_{i,t} \geqslant u_{i,t-1}} (u_{i,t} - u_{i,t-1}) \ln \frac{d}{\alpha} + \left(\sum_{i:\ u_{i,t} \geqslant u_{i,t-1}} u_{i,t-1} + \sum_{i:\ u_{i,t} < u_{i,t-1}} u_{i,t}\right) \ln \frac{1}{1-\alpha} \end{aligned}$$

[‡]As can be seen by noting that $\ln\big(1/(1-x)\big) < x/(1-x)$

$$= D_{\mathrm{TV}}(\boldsymbol{u}_t, \boldsymbol{u}_{t-1}) \ln \frac{d}{\alpha} + \underbrace{\left(\sum_{i=1}^{d} u_{i,t} - \sum_{i \,:\, u_{i,t} \geqslant u_{i,t-1}} (u_{i,t} - u_{i,t-1}) \right)}_{= \|\boldsymbol{u}_t\|_1 - D_{\mathrm{TV}}(\boldsymbol{u}_t, \boldsymbol{u}_{t-1})} \ln \frac{1}{1-\alpha} \,.$$

The sum of the second term in (3.7) telescopes. Substituting the obtained bounds in the first sum of the right-hand side in (3.7), and summing over $t = 2, \dots, T$, leads to

$$\sum_{t=2}^{T} \sum_{i=1}^{d} u_{i,t} \ln \frac{v_{i,t+1}}{\widehat{p}_{i,t}} \leqslant m(\boldsymbol{u}_1^T) \ln \frac{d}{\alpha} + \left(\sum_{t=2}^{T} \|\boldsymbol{u}_t\|_1 - m(\boldsymbol{u}_1^T) \right) \ln \frac{1}{1-\alpha} + \sum_{i=1}^{d} u_{i,1} \ln \frac{1}{v_{i,2}} - \underbrace{u_{i,T} \ln \frac{1}{v_{i,T+1}}}_{\leqslant 0} \,.$$

We hence get from (3.6), which we use in particular for $t = 1$,

$$\sum_{t=1}^{T} \|\boldsymbol{u}_t\|_1 \, \widehat{\boldsymbol{p}}_t^{\top} \ell_t - \boldsymbol{u}_t^{\top} \ell_t \leqslant \frac{1}{\eta} \sum_{i=1}^{d} u_{i,1} \ln \frac{1}{\widehat{p}_{i,1}} + \frac{\eta}{8} \sum_{t=1}^{T} \|\boldsymbol{u}_t\|_1 + \frac{m(\boldsymbol{u}_1^T)}{\eta} \ln \frac{d}{\alpha} + \frac{\sum_{t=2}^{T} \|\boldsymbol{u}_t\|_1 \, m(\boldsymbol{u}_1^T)}{\eta} \ln \frac{1}{1-\alpha} \,.$$

3.6. Applications

We now show how our regret bounds can be specialized to obtain bounds on adaptive and discounted regret, and on regret with time-selection functions. We show regret bounds only for the specific instance of the generalized share algorithm using update (3.5); but the discussion below also holds up to minor modifications for the forecaster studied in Theorem 3.2.

Adaptive regret was introduced by Hazan and Seshadhri [90] and can be viewed as a variant of discounted regret where the monotonicity assumption is dropped. For $\tau_0 \in \{1, \dots, T\}$, the τ_0-*adaptive regret* of a forecaster is defined by

$$\mathcal{R}_T^{\tau_0-\mathrm{adapt}} = \max_{\substack{[r,s] \subset [1,T] \\ s+1-r \leqslant \tau_0}} \left\{ \sum_{t=r}^{s} \widehat{\boldsymbol{p}}_t^{\top} \ell_t - \min_{\boldsymbol{q} \in \Delta_d} \sum_{t=r}^{s} \boldsymbol{q}^{\top} \ell_t \right\} . \tag{3.9}$$

The fact that this is a special case of (3.2) clearly emerges from the proof of Corollary 3.5 below here.

Adaptive regret is an alternative way to measure the performance of a forecaster against a changing environment. It is a straightforward observation that adaptive regret bounds also lead to shifting regret bounds (in terms of hard shifts). In this chapter we note that these two notions of regret share an even tighter connection, as they can be both viewed as instances of the same *alma mater* notion of regret, i.e., the generalized shifting regret introduced in Section 3.3. The work Hazan and Seshadhri

[90] essentially considered the case of online convex optimization with exp-concave loss function; in case of general convex functions, they also mentioned that the greedy projection forecaster of Zinkevich [151] enjoys adaptive regret guarantees. This is obtained in much the same way as we obtain an adaptive regret bound for the fixed-share forecaster in the next result.

Corollary 3.5. *Suppose that Algorithm 5 is run with the shared update (3.5). Then for all $T \geqslant 1$, for all sequences $\ell_1, \dots, \ell_T$ of loss vectors $\ell_t \in [0,1]^d$, and for all $\tau_0 \in \{1, \dots, T\}$,*

$$\mathcal{R}_T^{\tau_0-\mathrm{adapt}} \leqslant \sqrt{\frac{\tau_0}{2}\left(\tau_0\, h\left(\frac{1}{\tau_0}\right) + \ln d\right)} \leqslant \sqrt{\frac{\tau_0}{2}\,\ln(ed\tau_0)}$$

whenever η and α are chosen optimally (depending on τ_0 and T).

As mentioned in Hazan and Seshadhri [90], standard lower bounds on the regret show that the obtained bound is optimal up to the logarithmic factors.

Proof For $1 \leqslant r \leqslant s \leqslant T$ and $\boldsymbol{q} \in \Delta_d$, the regret in the right-hand side of (3.9) equals the regret considered in Theorem 3.3 against the sequence $\boldsymbol{u}_1^T$ defined as $\boldsymbol{u}_t = \boldsymbol{q}$ for $t = r, \dots, s$ and $\mathbf{0} = (0, \dots, 0)$ for the remaining t. When $r \geqslant 2$, this sequence is such that $D_{\mathrm{TV}}(\boldsymbol{u}_r, \boldsymbol{u}_{r-1}) = D_{\mathrm{TV}}(\boldsymbol{q}, \mathbf{0}) = 1$ and $D_{\mathrm{TV}}(\boldsymbol{u}_{s+1}, \boldsymbol{u}_s) = D_{\mathrm{TV}}(\mathbf{0}, \boldsymbol{q}) = 0$ so that $m(\boldsymbol{u}_1^T) = 1$, while $\|\boldsymbol{u}_1\|_1 = 0$. When $r = 1$, we have $\|\boldsymbol{u}_1\|_1 = 1$ and $m(\boldsymbol{u}_1^T) = 0$. In all cases, $m(\boldsymbol{u}_1^T) + \|\boldsymbol{u}_1\|_1 = 1$, that is, $m_0 = 1$. Specializing the bound of Theorem 3.3 with the additional choice $U_0 = \tau_0$ gives the result. ∎

Discounted regret was introduced in Cesa-Bianchi and Lugosi [43, Section 2.11] and is defined by

$$\max_{\boldsymbol{q} \in \Delta_d} \sum_{t=1}^{T} \beta_{t,T}\left(\widehat{\boldsymbol{p}}_t^{\top} \ell_t - \boldsymbol{q}^{\top} \ell_t\right) . \tag{3.10}$$

The discount factors $\beta_{t,T}$ measure the relative importance of more recent losses to older losses. For instance, for a given horizon T, the discounts $\beta_{t,T}$ may be larger as t is closer to T. On the contrary, in a game-theoretic setting, the earlier losses may matter more then the more recent ones (because of interest rates), in which case $\beta_{t,T}$ would be smaller as t gets closer to T. We mostly consider below monotonic sequences of discounts (both non-decreasing and non-increasing). Up to a normalization, we assume that all discounts $\beta_{t,T}$ are in $[0,1]$. As shown in Cesa-Bianchi and Lugosi [43], a minimal requirement to get non-trivial bounds is that the sum of the discounts satisfies $U_T = \sum_{t \leqslant T} \beta_{t,T} \to \infty$ as $T \to \infty$.

A natural objective is to show that the quantity in (3.10) is $o(U_T)$, for instance, by bounding it by something of the order of $\sqrt{U_T}$. We claim that Corollary 3.4 does so, at least whenever the sequences $(\beta_{t,T})$ are monotonic for all T. To support this claim, we only need to show that $m_0 = 1$ is a suitable value to deal with (3.10). Indeed, for all $T \geqslant 1$ and for all $\boldsymbol{q} \in \Delta_d$, the measure of regularity involved in the corollary satisfies

$$\|\beta_{1,T}\boldsymbol{q}\|_1 + m\big((\beta_{t,T}\boldsymbol{q})_{t \leqslant T}\big) = \beta_{1,T} + \sum_{t=2}^{T}\big(\beta_{t,T} - \beta_{t-1,T}\big)_+ = \max\big\{\beta_{1,T},\, \beta_{T,T}\big\} \leqslant 1\,,$$

where the second equality follows from the monotonicity assumption on the discounts.

The values of the discounts for all t and T are usually known in advance. However, the horizon T is not. Hence, a calibration issue may arise. The online tuning of the parameters α and η shown in Section 3.7.3 entails a forecaster that can get discounted regret bounds of the order $\sqrt{U_T}$ for all T. The fundamental reason for this is that the discounts only come in the definition of the fixed-share forecaster via their sums. In contrast, the forecaster discussed in Cesa-Bianchi and Lugosi [43, Section 2.11] weighs each instance t directly with $\beta_{t,T}$ (i.e., in the very definition of the forecaster) and enjoys therefore no regret guarantees for horizons other than T (neither before T nor after T). Therein, the knowledge of the horizon T is so crucial that it cannot be dealt with easily, not even with online calibration of the parameters or with a doubling trick. We insist that for the fixed-share forecaster, much flexibility is gained as some of the discounts $\beta_{t,T}$ can change in a drastic manner for a round T to values $\beta_{t,T+1}$ for the next round. However we must admit that the bound of Cesa-Bianchi and Lugosi [43, Section 2.11] is smaller than the one obtained above, as it of the order of $\sqrt{\sum_{t\leqslant T}\beta_{t,T}^2}$, in contrast to our $\sqrt{\sum_{t\leqslant T}\beta_{t,T}}$ bound. Again, this improvement was made possible because of the knowledge of the time horizon.

As for the comparison to the setting of discounted losses of Chernov and Zhdanov [47], we note that the latter can be cast as a special case of our setting (since the discounting weights take the special form $\beta_{t,T} = \gamma_t \dots \gamma_{T-1}$ therein, for some sequence γ_s of positive numbers). In particular, the fixed-share forecaster can satisfy the bound stated in Chernov and Zhdanov [47, Theorem 2], for instance, by using the online tuning techniques of Section 3.7.3. A final reference to mention is the setting of time-selection functions of Blum and Mansour [29, Section 6], which basically corresponds to knowing in advance the weights $\|\boldsymbol{u}_t\|_1$ of the comparison sequence $\boldsymbol{u}_1, \dots, \boldsymbol{u}_T$ the forecaster will be evaluated against. We thus generalize their results as well.

3.7. Refinements and extensions

We now show that techniques for refining the standard online analysis can be easily applied to our framework. We focus on the following: improvement for small losses, sparse target sequences, and dynamic tuning of parameters. Not all of them where within reach of previous analyses.

3.7.1. Improvement for small losses

The regret bounds of the fixed-share forecaster can be significantly improved when the cumulative loss of the best sequence of experts is small. The next result improves on Corollary 3.4 whenever $L_0 \ll U_0$. For concreteness, we focus on the fixed-share update (3.5).

Corollary 3.6. *Suppose Algorithm 5 is run with the update (3.5). Let $m_0 > 0$, $U_0 > 0$, and $L_0 > 0$. For all $T \geqslant 1$, for all sequences $\ell_1, \dots, \ell_T$ of loss vectors $\ell_t \in [0,1]^d$, and for all sequences $\boldsymbol{u}_1, \dots, \boldsymbol{u}_T \in \mathbb{R}_+^d$ with $\|\boldsymbol{u}_1\|_1 + m(\boldsymbol{u}_1^T) \leqslant m_0$, $\sum_{t=1}^T \|\boldsymbol{u}_t\|_1 \leqslant U_0$, and $\sum_{t=1}^T \boldsymbol{u}_t^\top \ell_t \leqslant L_0$,*

$$\sum_{t=1}^T \|\boldsymbol{u}_t\|_1 \, \widehat{\boldsymbol{p}}_t^\top \ell_t - \sum_{t=1}^T \boldsymbol{u}_t^\top \ell_t \leqslant \sqrt{L_0 \, m_0 \left(\ln d + \ln\left(\frac{e\,U_0}{m_0}\right)\right)} + \ln d + \ln\left(\frac{e\,U_0}{m_0}\right)$$

whenever η and α are optimally chosen in terms of m_0, U_0, and L_0.

Here again, the parameters α and η may be tuned online using the techniques shown in Section 3.7.3. The above refinement is obtained by mimicking the analysis of Hedge forecasters for small losses (see, e.g., Cesa-Bianchi and Lugosi [43, Section 2.4]). In particular, one should substitute Lemma 3.1 with the following lemma in the analysis carried out in Section 3.5; its proof follows from the mere replacement of Hoeffding's inequality by Cesa-Bianchi and Lugosi [43, Lemma A.3], which states that for all $\eta \in \mathbb{R}$ and for all random variables X taking values in $[0,1]$, one has $\ln \mathbb{E}[e^{-\eta X}] \leqslant (e^{-\eta} - 1)\mathbb{E}X$.

Lemma 3.7. *Algorithm 5 satisfies*

$$\frac{1 - e^{-\eta}}{\eta} \widehat{\boldsymbol{p}}_t^\top \ell_t - \boldsymbol{q}_t^\top \ell_t \leqslant \frac{1}{\eta} \sum_{i=1}^{d} q_{i,t} \ln \left(\frac{v_{i,t}}{\widehat{p}_{i,t+1}} \right) ,$$

for all $\boldsymbol{q}_t \in \Delta_d$.

3.7.2. Sparse target sequences

The work Bousquet and Warmuth [32] introduced forecasters that are able to efficiently compete with the best sequence of experts among all those sequences that only switch a bounded number of times and also take a small number of different values. Such "sparse" sequences of experts appear naturally in many applications. In this section we show that their algorithms in fact work very well in comparison with a much larger class of sequences $\boldsymbol{u}_1, \dots, \boldsymbol{u}_T$ that are "regular"—that is, $m(\boldsymbol{u}_1^T)$, defined in (3.3) is small—and "sparse" in the sense that the quantity $n(\boldsymbol{u}_1^T) = \sum_{i=1}^d \max_{t=1,\dots,T} u_{i,t}$ is small. Note that when $\boldsymbol{q}_t \in \Delta_d$ for all t, then two interesting upper bounds can be provided. First, denoting the union of the supports of these convex combinations by $S \subseteq \{1, \dots, d\}$, we have $n(\boldsymbol{q}_1^T) \leqslant |S|$, the cardinality of S. Also, $n(\boldsymbol{q}_1^T) \leqslant \big|\{\boldsymbol{q}_t, \quad t = 1, \dots, T\}\big|$, the cardinality of the pool of convex combinations. Thus, $n(\boldsymbol{u}_1^T)$ generalizes the notion of sparsity of Bousquet and Warmuth [32].

Here we consider a family of shared updates of the form

$$\widehat{p}_{j,t} = (1 - \alpha) v_{j,t} + \alpha \frac{w_{j,t}}{Z_t} , \qquad 0 \leqslant \alpha \leqslant 1 , \tag{3.11}$$

where the $w_{j,t}$ are nonnegative weights that may depend on past and current pre-weights and $Z_t = \sum_{i=1}^d w_{i,t}$ is a normalization constant. Shared updates of this form were proposed by Bousquet and Warmuth [32, Sections 3 and 5.2]. Apart from generalizing the regret bounds of Bousquet and Warmuth [32], we believe that the analysis given below is significantly simpler and more transparent. We are also able to slightly improve their original bounds.

We focus on choices of the weights $w_{j,t}$ that satisfy the following conditions: there exists a constant $C \geqslant 1$ such that for all $j = 1, \dots, d$ and $t = 1, \dots, T$,

$$v_{j,t} \leqslant w_{j,t} \leqslant 1 \qquad \text{and} \qquad C\, w_{j,t+1} \geqslant w_{j,t} . \tag{3.12}$$

The next result improves on Theorem 3.3 when $T \ll d$ and $n(\boldsymbol{u}_1^T) \ll m(\boldsymbol{u}_1^T)$, that is, when the dimension (or number of experts) d is large but the sequence $\boldsymbol{u}_1^T$ is sparse. Its proof can be found in the supplementary material; it is a variation on the proof of Theorem 3.3.

Theorem 3.8. *Suppose Algorithm 5 is run with the shared update (3.11) with weights satisfying the conditions (3.12). Then for all $T \geqslant 1$, for all sequences $\ell_1, \dots, \ell_T$ of loss vectors $\ell_t \in [0,1]^d$, and for all sequences $\boldsymbol{u}_1, \dots, \boldsymbol{u}_T \in \mathbb{R}_+^d$,*

$$\sum_{t=1}^T \|\boldsymbol{u}_t\|_1 \, \widehat{\boldsymbol{p}}_t^\top \ell_t - \sum_{t=1}^T \boldsymbol{u}_t^\top \ell_t \leqslant \frac{n(\boldsymbol{u}_1^T) \ln d}{\eta} + \frac{n(\boldsymbol{u}_1^T)\, T \ln C}{\eta} + \frac{\eta}{8} \sum_{t=1}^T \|\boldsymbol{u}_t\|_1 + \frac{m(\boldsymbol{u}_1^T)}{\eta} \ln \frac{\max_{t \leqslant T} Z_t}{\alpha} + \frac{\sum_{t=2}^T \|\boldsymbol{u}_t\|_1 - m(\boldsymbol{u}_1^T)}{\eta} \ln \frac{1}{1-\alpha} \, .$$

Corollaries 8 and 9 of Bousquet and Warmuth [32] can now be generalized (and even improved); we do so—in the supplementary material—by showing two specific instances of the generic update (3.11) that satisfy (3.12).

3.7.3. Online tuning of the parameters

The forecasters studied above need their parameters η and α to be tuned according to various quantities, including the time horizon T. We show here how the trick of Auer et al. [20] of having these parameters vary over time can be extended to our setting. For the sake of concreteness we focus on the fixed-share update, i.e., Algorithm 5 run with the update (3.5). We respectively replace steps 3 and 4 of its description by the loss and shared updates

$$v_{j,t+1} = \frac{\widehat{p}_{j,t}^{\frac{\eta_t}{\eta_{t-1}}} e^{-\eta_t \ell_{j,t}}}{\sum_{i=1}^d \widehat{p}_{i,t}^{\frac{\eta_t}{\eta_{t-1}}} e^{-\eta_t \ell_{i,t}}} \qquad \text{and} \qquad p_{j,t+1} = \frac{\alpha_t}{d} + (1-\alpha_t)\, v_{j,t+1} \, , \tag{3.13}$$

for all $t \geqslant 1$ and all $j \in \{1, \dots, d\}$, where (η_τ) and (α_τ) are two sequences of positive numbers, indexed by $\tau \geqslant 1$. We also conventionally define $\eta_0 = \eta_1$. Theorem 3.3 is then adapted in the following way (when $\eta_t \equiv \eta$ and $\alpha_t \equiv \alpha$, Theorem 3.3 is exactly recovered).

Theorem 3.9. *The forecaster based on the updates (3.13) is such that whenever $\eta_t \leqslant \eta_{t-1}$ and $\alpha_t \leqslant \alpha_{t-1}$ for all $t \geqslant 1$, the following performance bound is achieved. For all $T \geqslant 1$, for all sequences $\ell_1, \dots, \ell_T$ of loss vectors $\ell_t \in [0,1]^d$, and for all $\boldsymbol{u}_1, \dots, \boldsymbol{u}_T \in \mathbb{R}_+^d$,*

$$\sum_{t=1}^T \|\boldsymbol{u}_t\|_1 \, \widehat{\boldsymbol{p}}_t^\top \ell_t - \sum_{t=1}^T \boldsymbol{u}_t^\top \ell_t \leqslant \left(\frac{\|\boldsymbol{u}_t\|_1}{\eta_1} + \sum_{t=2}^T \|\boldsymbol{u}_t\|_1 \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}} \right) \right) \ln d + \frac{m(\boldsymbol{u}_1^T)}{\eta_T} \ln \frac{d(1-\alpha_T)}{\alpha_T} + \sum_{t=2}^T \frac{\|\boldsymbol{u}_t\|_1}{\eta_{t-1}} \ln \frac{1}{1-\alpha_t} + \sum_{t=1}^T \frac{\eta_{t-1}}{8} \, \|\boldsymbol{u}_t\|_1 \, .$$

We provide an illustration of this bound in the supplementary material.

Appendices for Chapter 3

3.A. Online convex optimization on the simplex

By using a standard reduction, the results of the main body of the chapter (for linear optimization on the simplex) can be applied to online convex optimization on the simplex. In this setting, at each step t the forecaster chooses $\widehat{\boldsymbol{p}}_t \in \Delta_d$ and then is given access to a convex loss $\ell_t : \Delta_d \to [0,1]$. Now, using Algorithm 5 with the loss vector $\ell_t \in \partial \ell_t(\widehat{\boldsymbol{p}}_t)$ given by a subgradient of ℓ_t leads to the desired bounds. Indeed, by the convexity of ℓ_t, the regret at each time t with respect to any vector $\boldsymbol{u}_t \in \mathbb{R}_+^d$ with $\|\boldsymbol{u}_t\|_1 > 0$ is then bounded as

$$\|\boldsymbol{u}_t\|_1 \left(\ell_t(\widehat{\boldsymbol{p}}_t) - \ell_t\!\left(\frac{\boldsymbol{u}_t}{\|\boldsymbol{u}_t\|_1} \right) \right) \leqslant \left(\|\boldsymbol{u}_t\|_1 \, \widehat{\boldsymbol{p}}_t - \boldsymbol{u}_t \right)^\top \ell_t \,.$$

3.B. Proofs

3.B.1. Proof of Theorem 3.8; application of the bound to two different updates

Proof The beginning and the end of the proof are similar to the one of Theorem 3.3, as they do not depend on the specific weight update. In particular, inequalities (3.6) and (3.7) remain the same. The proof is modified after (3.8), which this time we upper bound using the first condition in (3.12),

$$\sum_{i=1}^d \left(u_{i,t} \ln \frac{1}{\widehat{p}_{i,t}} - u_{i,t-1} \ln \frac{1}{v_{i,t}} \right) = \sum_{i \,:\, u_{i,t} \geqslant u_{i,t-1}} (u_{i,t} - u_{i,t-1}) \ln \frac{1}{\widehat{p}_{i,t}} + u_{i,t-1} \ln \frac{v_{i,t}}{\widehat{p}_{i,t}}$$
$$+ \sum_{i \,:\, u_{i,t} < u_{i,t-1}} \underbrace{(u_{i,t} - u_{i,t-1})}_{\leqslant 0} \underbrace{\ln \frac{1}{v_{i,t}}}_{\geqslant \ln(1/w_{i,t})} + u_{i,t} \ln \frac{v_{i,t}}{\widehat{p}_{i,t}} \,. \quad (3.14)$$

By definition of the shared update (3.11), we have $1/\widehat{p}_{i,t} \leqslant Z_t/(\alpha\, w_{i,t})$ and $v_{i,t}/\widehat{p}_{i,t} \leqslant 1/(1-\alpha)$. We then upper bound the quantity at hand in (3.14) by

$$\sum_{i \,:\, u_{i,t} \geqslant u_{i,t-1}} (u_{i,t} - u_{i,t-1}) \ln \left(\frac{Z_t}{\alpha\, w_{i,t}} \right) + \left(\sum_{i \,:\, u_{i,t} \geqslant u_{i,t-1}} u_{i,t-1} + \sum_{i \,:\, u_{i,t} < u_{i,t-1}} u_{i,t} \right) \ln \frac{1}{1-\alpha}$$
$$+ \sum_{i \,:\, u_{i,t} < u_{i,t-1}} (u_{i,t} - u_{i,t-1}) \ln \frac{1}{w_{i,t}}$$
$$= D_{\mathrm{TV}}(\boldsymbol{u}_t, \boldsymbol{u}_{t-1}) \ln \frac{Z_t}{\alpha} + \left(\|\boldsymbol{u}_t\|_1 - D_{\mathrm{TV}}(\boldsymbol{u}_t, \boldsymbol{u}_{t-1}) \right) \ln \frac{1}{1-\alpha} + \sum_{i=1}^d (u_{i,t} - u_{i,t-1}) \ln \frac{1}{w_{i,t}} \,.$$

Proceeding as in the end of the proof of Theorem 3.3, we then get the claimed bound, provided that we can show that

$$\sum_{t=2}^{T}\sum_{i=1}^{d}(u_{i,t}-u_{i,t-1})\ln\frac{1}{w_{i,t}} \leqslant n(\boldsymbol{u}_1^T)\,(\ln d + T\ln C) - \|\boldsymbol{u}_1\|_1 \ln d\,,$$

which we do next. Indeed, the left-hand side can be rewritten as

$$\begin{aligned}
&\sum_{t=2}^{T}\sum_{i=1}^{d}\left(u_{i,t}\ln\frac{1}{w_{i,t}} - u_{i,t}\ln\frac{1}{w_{i,t+1}}\right) + \sum_{t=2}^{T}\sum_{i=1}^{d}\left(u_{i,t}\ln\frac{1}{w_{i,t+1}} - u_{i,t-1}\ln\frac{1}{w_{i,t}}\right)\\
&\qquad\leqslant \left(\sum_{t=2}^{T}\sum_{i=1}^{d}u_{i,t}\ln\frac{C\,w_{i,t+1}}{w_{i,t}}\right) + \left(\sum_{i=1}^{d}u_{i,T}\ln\frac{1}{w_{i,T+1}} - \sum_{i=1}^{d}u_{i,1}\ln\frac{1}{w_{i,2}}\right)\\
&\qquad\leqslant \left(\sum_{i=1}^{d}\left(\max_{t=1,\dots,T}u_{i,t}\right)\sum_{t=2}^{T}\ln\frac{C\,w_{i,t+1}}{w_{i,t}}\right)\\
&\qquad\qquad + \left(\sum_{i=1}^{d}\left(\max_{t=1,\dots,T}u_{i,t}\right)\ln\frac{1}{w_{i,T+1}} - \sum_{i=1}^{d}u_{i,1}\ln\frac{1}{w_{i,2}}\right)\\
&\qquad= \sum_{i=1}^{d}\left(\max_{t=1,\dots,T}u_{i,t}\right)\left((T-1)\ln C + \ln\frac{1}{w_{i,2}}\right) - \sum_{i=1}^{d}u_{i,1}\ln\frac{1}{w_{i,2}}\,,
\end{aligned}$$

where we used $C \geqslant 1$ for the first inequality and the second condition in (3.12) for the second inequality. The proof is concluded by noting that (3.12) entails $w_{i,2} \geqslant (1/C)w_{i,1} \geqslant (1/C)v_{i,1} = 1/(dC)$ and that the coefficient $\max_{t=1,\dots,T}u_{i,t} - u_{i,1}$ in front of $\ln(1/w_{i,2})$ is nonnegative. ∎

The first update uses $w_{j,t} = \max_{s\leqslant t} v_{j,s}$. Then (3.12) is satisfied with $C = 1$. Moreover, since a sum of maxima of nonnegative elements is smaller than the sum of the sums, $Z_t \leqslant \min\{d, t\} \leqslant T$. This immediately gives the following result.

Corollary 3.10. *Suppose Algorithm 5 is run with the update (3.11) with $w_{j,t} = \max_{s\leqslant t} v_{j,s}$. For all $T \geqslant 1$, for all sequences $\ell_1, \dots, \ell_T$ of loss vectors $\ell_t \in [0,1]^d$, and for all $\boldsymbol{q}_1, \dots, \boldsymbol{q}_T \in \Delta_d$,*

$$\sum_{t=1}^{T}\widehat{\boldsymbol{p}}_t^{\top}\ell_t - \sum_{t=1}^{T}\boldsymbol{q}_t^{\top}\ell_t \leqslant \frac{n(\boldsymbol{q}_1^T)\ln d}{\eta} + \frac{\eta}{8}T + \frac{m(\boldsymbol{q}_1^T)}{\eta}\ln\frac{T}{\alpha} + \frac{T - m(\boldsymbol{q}_1^T) - 1}{\eta}\ln\frac{1}{1-\alpha}\,.$$

The second update we discuss uses $w_{j,t} = \max_{s\leqslant t} e^{\gamma(s-t)}v_{j,s}$ in (3.11) for some $\gamma > 0$. Both conditions in (3.12) are satisfied with $C = e^{\gamma}$. One also has that

$$Z_t \leqslant d \qquad \text{and} \qquad Z_t \leqslant \sum_{\tau\geqslant 0} e^{-\gamma\tau} = \frac{1}{1-e^{-\gamma}} \leqslant \frac{1}{\gamma}$$

as $e^x \geqslant 1 + x$ for all real x. The bound of Theorem 3.8 then instantiates as

$$\frac{n(\boldsymbol{q}_1^T)\ln d}{\eta} + \frac{n(\boldsymbol{q}_1^T)\,T\gamma}{\eta} + \frac{\eta}{8}T + \frac{m(\boldsymbol{q}_1^T)}{\eta}\ln\frac{\min\{d,\,1/\gamma\}}{\alpha} + \frac{T - m(\boldsymbol{q}_1^T) - 1}{\eta}\ln\frac{1}{1-\alpha}$$

when sequences $\boldsymbol{u}_t = \boldsymbol{q}_t \in \Delta_d$ are considered. This bound is best understood when γ is tuned

optimally based on T and on two bounds m_0 and n_0 over the quantities $m(\boldsymbol{q}_1^T)$ and $n(\boldsymbol{q}_1^T)$. Indeed, by optimizing $n_0 T\gamma + m_0 \ln(1/\gamma)$, i.e., by choosing $\gamma = m_0/(n_0\, T)$, one gets a bound that improves on the one of the previous corollary:

Corollary 3.11. *Let m_0, $n_0 > 0$. Suppose Algorithm 5 is run with the update*

$$w_{j,t} = \max_{s \leqslant t} e^{\gamma(s-t)} v_{j,s}$$

where $\gamma = m_0/(n_0\, T)$. For all $T \geqslant 1$, for all sequences $\ell_1, \dots, \ell_T$ of loss vectors $\ell_t \in [0,1]^d$, and for all $\boldsymbol{q}_1, \dots, \boldsymbol{q}_T \in \Delta_d$ such that $m(\boldsymbol{q}_1^T) \leqslant m_0$ and $n(\boldsymbol{q}_1^T) \leqslant n_0$, we have

$$\sum_{t=1}^T \widehat{\boldsymbol{p}}_t^\top \ell_t - \sum_{t=1}^T \boldsymbol{q}_t^\top \ell_t \;\leqslant\; \frac{n_0 \ln d}{\eta} + \frac{m_0}{\eta}\left(1 + \ln \min\left\{d,\, \frac{n_0\, T}{m_0}\right\}\right) + \frac{\eta}{8} T + \frac{m_0}{\eta} \ln \frac{1}{\alpha} + \frac{T - m_0 - 1}{\eta} \ln \frac{1}{1-\alpha}\,.$$

As the factors $e^{-\gamma t}$ cancel out in the numerator and denominator of the ratio in (3.11), there is a straightforward implementation of the algorithm (not requiring the knowledge of T) that needs to maintain only d weights.

In contrast, the corresponding algorithm of Bousquet and Warmuth [32], using the updates $\widehat{p}_{j,t} = (1-\alpha) v_{j,t} + \alpha S_t^{-1} \sum_{s \leqslant t-1} (s-t)^{-1} v_{j,s}$ or $\widehat{p}_{j,t} = (1-\alpha) v_{j,t} + \alpha S_t^{-1} \max_{s \leqslant t-1} (s-t)^{-1} v_{j,s}$, where S_t denote normalization factors, needs to maintain $\mathcal{O}(dT)$ weights with a naive implementation, and $\mathcal{O}(d \ln T)$ weights with a more sophisticated one. In addition, the obtained bounds are slightly worse than the one stated above in Corollary 3.11 as an additional factor of $m_0 \ln(1 + \ln T)$ is present in Bousquet and Warmuth [32, Corollary 9].

3.B.2. Proof of Theorem 3.9; illustration of the obtained bound

We first adapt Lemma 3.1.

Lemma 3.12. *The forecaster based on the loss and shared updates (3.13) satisfies, for all $t \geqslant 1$ and for all $\boldsymbol{q}_t \in \Delta_d$,*

$$\big(\widehat{\boldsymbol{p}}_t - \boldsymbol{q}_t\big)^\top \ell_t \leqslant \sum_{i=1}^d q_{i,t} \left(\frac{1}{\eta_{t-1}} \ln \frac{1}{\widehat{p}_{i,t}} - \frac{1}{\eta_t} \ln \frac{1}{v_{i,t+1}}\right) + \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right) \ln d + \frac{\eta_{t-1}}{8}\,,$$

whenever $\eta_t \leqslant \eta_{t-1}$.

Proof By Hoeffding's inequality,

$$\sum_{j=1}^d \widehat{p}_{j,t}\, \ell_{j,t} \leqslant -\frac{1}{\eta_{t-1}} \ln \left(\sum_{j=1}^d \widehat{p}_{j,t}\, e^{-\eta_{t-1}\, \ell_{j,t}}\right) + \frac{\eta_{t-1}}{8}\,.$$

By Jensen's inequality, since $\eta_t \leqslant \eta_{t-1}$ and thus $x \mapsto x^{\frac{\eta_{t-1}}{\eta_t}}$ is convex,

$$\frac{1}{d}\sum_{j=1}^{d} \widehat{p}_{j,t}\, e^{-\eta_{t-1}\ell_{j,t}} = \frac{1}{d}\sum_{j=1}^{d} \left(\widehat{p}_{j,t}^{\frac{\eta_t}{\eta_{t-1}}} e^{-\eta_t \ell_{j,t}}\right)^{\frac{\eta_{t-1}}{\eta_t}} \geqslant \left(\frac{1}{d}\sum_{j=1}^{d} \widehat{p}_{j,t}^{\frac{\eta_t}{\eta_{t-1}}} e^{-\eta_t \ell_{j,t}}\right)^{\frac{\eta_{t-1}}{\eta_t}} .$$

Substituting in Hoeffding's bound we get

$$\widehat{\boldsymbol{p}}_t^\top \ell_t \leqslant -\frac{1}{\eta_t} \ln \left(\sum_{j=1}^{d} \widehat{p}_{j,t}^{\frac{\eta_t}{\eta_{t-1}}} e^{-\eta_t \ell_{j,t}}\right) + \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right)\ln d + \frac{\eta_{t-1}}{8} .$$

Now, by definition of the loss update in (3.13), for all $i \in \{1, \dots, d\}$,

$$\sum_{j=1}^{d} \widehat{p}_{j,t}^{\frac{\eta_t}{\eta_{t-1}}} e^{-\eta_t \ell_{j,t}} = \frac{1}{v_{i,t+1}} \widehat{p}_{i,t}^{\frac{\eta_t}{\eta_{t-1}}} e^{-\eta_t \ell_{i,t}} ,$$

which, after substitution in the previous bound leads to the inequality

$$\widehat{\boldsymbol{p}}_t^\top \ell_t \leqslant \ell_{i,t} + \frac{1}{\eta_{t-1}} \ln \frac{1}{\widehat{p}_{i,t}} - \frac{1}{\eta_t} \ln \frac{1}{v_{i,t+1}} + \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right)\ln d + \frac{\eta_{t-1}}{8} ,$$

valid for all $i \in \{1, \dots, d\}$. The proof is concluded by taking a convex aggregation over i with respect to $\boldsymbol{q}_t$. ∎

The proof of Theorem 3.9 follows the steps of the one of Theorem 3.3; we sketch it below.

Proof (of Theorem 3.9) Applying Lemma 3.12 with $\boldsymbol{q}_t = \boldsymbol{u}_t / \|\boldsymbol{u}_t\|_1$, and multiplying by $\|\boldsymbol{u}_t\|_1$, we get for all $t \geqslant 1$ and $\boldsymbol{u}_t \in \mathbb{R}_+^d$,

$$\|\boldsymbol{u}_t\|_1\, \widehat{\boldsymbol{p}}_t^\top \ell_t - \boldsymbol{u}_t^\top \ell_t \leqslant \frac{1}{\eta_{t-1}} \sum_{i=1}^{d} u_{i,t} \ln \frac{1}{\widehat{p}_{i,t}} - \frac{1}{\eta_t} \sum_{i=1}^{d} u_{i,t} \ln \frac{1}{v_{i,t+1}} + \|\boldsymbol{u}_t\|_1 \left(\frac{1}{\eta_t} - \frac{1}{\eta_{t-1}}\right)\ln d + \frac{\eta_{t-1}}{8}\, \|\boldsymbol{u}_t\|_1 . \quad (3.15)$$

We will sum these bounds over $t \geqslant 1$ to get the desired result but need to perform first some additional boundings for $t \geqslant 2$; in particular, we examine

$$\frac{1}{\eta_{t-1}} \sum_{i=1}^{d} u_{i,t} \ln \frac{1}{\widehat{p}_{i,t}} - \frac{1}{\eta_t} \sum_{i=1}^{d} u_{i,t} \ln \frac{1}{v_{i,t+1}} = \frac{1}{\eta_{t-1}} \sum_{i=1}^{d} \left(u_{i,t} \ln \frac{1}{\widehat{p}_{i,t}} - u_{i,t-1} \ln \frac{1}{v_{i,t}}\right) + \sum_{i=1}^{d} \left(\frac{u_{i,t-1}}{\eta_{t-1}} \ln \frac{1}{v_{i,t}} - \frac{u_{i,t}}{\eta_t} \ln \frac{1}{v_{i,t+1}}\right) , \quad (3.16)$$

where the first difference in the right-hand side can be bounded as in (3.8) by

$$\sum_{i=1}^{d} \left(u_{i,t} \ln \frac{1}{\widehat{p}_{i,t}} - u_{i,t-1} \ln \frac{1}{v_{i,t}}\right)$$

$$
\begin{aligned}
&\leqslant \sum_{i\,:\,u_{i,t}\geqslant u_{i,t-1}} \left((u_{i,t}-u_{i,t-1}) \ln \frac{1}{\widehat{p}_{i,t}} + u_{i,t-1} \ln \frac{v_{i,t}}{\widehat{p}_{i,t}} \right) + \sum_{i\,:\,u_{i,t}<u_{i,t-1}} u_{i,t} \ln \frac{v_{i,t}}{\widehat{p}_{i,t}} \\
&\leqslant D_{TV}(\boldsymbol{u}_t, \boldsymbol{u}_{t-1}) \ln \frac{d}{\alpha_t} + \big(\|\boldsymbol{u}_t\|_1 - D_{TV}(\boldsymbol{u}_t, \boldsymbol{u}_{t-1}) \big) \ln \frac{1}{1-\alpha_t} \\
&\leqslant D_{TV}(\boldsymbol{u}_t, \boldsymbol{u}_{t-1}) \ln \frac{d(1-\alpha_T)}{\alpha_T} + \|\boldsymbol{u}_t\|_1 \ln \frac{1}{1-\alpha_t}\,, \qquad (3.17)
\end{aligned}
$$

where we used for the second inequality that the shared update in (3.13) is such that $1/\widehat{p}_{i,t} \leqslant d/\alpha_t$ and $v_{i,t}/\widehat{p}_{i,t} \leqslant 1/(1-\alpha_t)$, and for the third inequality, that $\alpha_t \geqslant \alpha_T$ and $x \mapsto (1-x)/x$ is increasing on $(0,1]$. Summing (3.16) over $t = 2, \dots, T$ using (3.17) and the fact that $\eta_t \geqslant \eta_T$, we get

$$
\begin{aligned}
&\sum_{t=2}^{T} \left(\frac{1}{\eta_{t-1}} \sum_{i=1}^{d} u_{i,t} \ln \frac{1}{\widehat{p}_{i,t}} - \frac{1}{\eta_t} \sum_{i=1}^{d} u_{i,t} \ln \frac{1}{v_{i,t+1}} \right) \\
&\quad\leqslant \frac{m(\boldsymbol{u}_1^T)}{\eta_T} \ln \frac{d(1-\alpha_T)}{\alpha_T} + \sum_{t=2}^{T} \frac{\|\boldsymbol{u}_t\|_1}{\eta_{t-1}} \ln \frac{1}{1-\alpha_t} + \sum_{i=1}^{d} \Bigg(\frac{u_{i,1}}{\eta_1} \ln \frac{1}{v_{i,2}} - \underbrace{\frac{u_{i,T}}{\eta_T} \ln \frac{1}{v_{i,T+1}}}_{\geqslant 0} \Bigg).
\end{aligned}
$$

An application of (3.15) —including for $t = 1$, for which we recall that $\widehat{p}_{i,1} = 1/d$ and $\eta_1 = \eta_0$ by convention— concludes the proof. ∎

We now instantiate the obtained bound to the case of, e.g., T–adaptive regret guarantees, when T is unknown and/or can increase without bounds.

Corollary 3.13. *The forecaster based on the updates discussed above with $\eta_t = \sqrt{\big(\ln(dt)\big)/t}$ for $t \geqslant 3$ and $\eta_0 = \eta_1 = \eta_2 = \eta_3$ on the one hand, $\alpha_t = 1/t$ on the other hand, is such that for all $T \geqslant 3$ and for all sequences $\ell_1, \dots, \ell_T$ of loss vectors $\ell_t \in [0,1]^d$,*

$$
\max_{[r,s]\subset[1,T]} \left\{ \sum_{t=r}^{s} \widehat{\boldsymbol{p}}_t^{\top} \ell_t - \min_{\boldsymbol{q}\in\Delta_d} \sum_{t=r}^{s} \boldsymbol{q}^{\top} \ell_t \right\} \leqslant \sqrt{2T \ln(dT)} + \sqrt{3 \ln(3d)}\,.
$$

Proof The sequence $n \mapsto \ln(n)/n$ is only non-increasing after round $n \geqslant 3$, so that the defined sequences of (α_t) and (η_t) are non-increasing, as desired. For a given pair (r,s) and a given $\boldsymbol{q} \in \Delta_d$, we consider the sequence ν_1^T defined in the proof of Corollary 3.5; it satisfies that $m(\boldsymbol{u}_1^T) \leqslant 1$ and $\|\boldsymbol{u}_t\|_1 \leqslant 1$ for all $t \geqslant 1$. Therefore, Theorem 3.9 ensures that

$$
\sum_{t=r}^{s} \widehat{\boldsymbol{p}}_t^{\top} \ell_t - \min_{\boldsymbol{q}\in\Delta_d} \sum_{t=r}^{s} \boldsymbol{q}^{\top} \ell_t \leqslant \frac{\ln d}{\eta_T} + \frac{1}{\eta_T} \ln \underbrace{\frac{d(1-\alpha_T)}{\alpha_T}}_{\leqslant dT} + \underbrace{\sum_{t=2}^{T} \frac{1}{\eta_{t-1}} \ln \frac{1}{1-\alpha_t}}_{\leqslant (1/\eta_T) \sum_{t=2}^{T} \ln(t/(t-1)) = (\ln T)/\eta_T} + \sum_{t=1}^{T} \frac{\eta_{t-1}}{8}\,.
$$

It only remains to substitute the proposed values of η_t and to note that

$$
\sum_{t=1}^{T} \eta_{t-1} \leqslant 3\eta_3 + \sum_{t=3}^{T-1} \frac{1}{\sqrt{t}} \sqrt{\ln(dT)} \leqslant 3\sqrt{\frac{\ln(3d)}{3}} + 2\sqrt{T}\,\sqrt{\ln(dT)}\,.
$$

∎

3.B.3. Proof of Theorem 3.2

We recall that the forecaster at hand is the one described in Algorithm 5, with the shared update $\widehat{\boldsymbol{p}}_{t+1} = \psi_{t+1}\big(\underline{V}_{t+1}\big)$ for

$$\psi_{t+1}\big(\underline{V}_{t+1}\big) \in \operatorname*{arg\,min}_{\boldsymbol{x}\in\Delta_d^\alpha} \mathcal{K}(\boldsymbol{x}, \boldsymbol{v}_{t+1})\,, \qquad \text{where} \quad \mathcal{K}(\boldsymbol{x}, \boldsymbol{v}_{t+1}) = \sum_{i=1}^d x_i \ln \frac{x_i}{v_{i,t+1}} \tag{3.18}$$

is the Kullback-Leibler divergence and $\Delta_d^\alpha = [\alpha/d, 1]^d \cap \Delta_d$ is the simplex of convex vectors with the constraint that each component be larger than α/d.

The proof of the performance bound starts with an extension of Lemma 3.1.

Lemma 3.14. *For all $t \geqslant 1$ and for all $\boldsymbol{q}_t \in \Delta_d^\alpha$, the generalized forecaster with the shared update (3.18) satisfies*

$$(\widehat{\boldsymbol{p}}_t - \boldsymbol{q}_t)^\top \ell_t \leqslant \frac{1}{\eta}\sum_{i=1}^d q_{i,t} \ln \frac{\widehat{p}_{i,t+1}}{\widehat{p}_{i,t}} + \frac{\eta}{8}\,.$$

Proof We rewrite the bound of Lemma 3.1 in terms of Kullback-Leibler divergences,

$$\begin{aligned}(\widehat{\boldsymbol{p}}_t - \boldsymbol{q}_t)^\top \ell_t &\leqslant \frac{1}{\eta}\sum_{i=1}^d q_{i,t} \ln \frac{v_{i,t+1}}{p_{i,t}} + \frac{\eta}{8} = \frac{\mathcal{K}(\boldsymbol{q}_t, \widehat{\boldsymbol{p}}_t) - \mathcal{K}(\boldsymbol{q}_t, \boldsymbol{v}_{t+1})}{\eta} + \frac{\eta}{8} \\ &\leqslant \frac{\mathcal{K}(\boldsymbol{q}_t, \widehat{\boldsymbol{p}}_t) - \mathcal{K}(\boldsymbol{q}_t, \widehat{\boldsymbol{p}}_{t+1})}{\eta} + \frac{\eta}{8} = \frac{1}{\eta}\sum_{i=1}^d q_{i,t} \ln \frac{\widehat{p}_{i,t+1}}{\widehat{p}_{i,t}} + \frac{\eta}{8}\,,\end{aligned}$$

where the last inequality holds by applying a generalized Pythagorean theorem for Bregman divergences (here, the Kullback-Leibler divergence) —see, e.g., Cesa-Bianchi and Lugosi [43, Lemma 11.3]. ∎

Proof Let $\boldsymbol{q}_t = \dfrac{\alpha}{d} + (1-\alpha)\dfrac{\boldsymbol{u}_t}{\|\boldsymbol{u}_t\|_1} \in \Delta_d^\alpha$. We have by rearranging the terms for all t,

$$\begin{aligned}\big(\|\boldsymbol{u}_t\|_1 \widehat{\boldsymbol{p}}_t - \boldsymbol{u}_t\big)^\top \ell_t &= \|\boldsymbol{u}_t\|_1 (\widehat{\boldsymbol{p}}_t - \boldsymbol{q}_t)^\top \ell_t + \Big(\frac{\alpha}{d}\|\boldsymbol{u}_t\|_1 - \alpha\boldsymbol{u}_t\Big)^\top \ell_t \\ &\leqslant \|\boldsymbol{u}_t\|_1 (\widehat{\boldsymbol{p}}_t - \boldsymbol{q}_t)^\top \ell_t + \alpha\|\boldsymbol{u}_t\|_1\,.\end{aligned}$$

Therefore, by applying Lemma 3.14 with $\boldsymbol{q}_t \in \Delta_d^\alpha$, we further upper bound the quantity of interest as

$$\big(\|\boldsymbol{u}_t\|_1 \widehat{\boldsymbol{p}}_t - \boldsymbol{u}_t\big)^\top \ell_t \leqslant \frac{\|\boldsymbol{u}_t\|_1}{\eta}\sum_{i=1}^d q_{i,t} \ln \frac{\widehat{p}_{i,t+1}}{\widehat{p}_{i,t}} + \frac{\eta}{8}\|\boldsymbol{u}_t\|_1 + \alpha\|\boldsymbol{u}_t\|_1\,.$$

The upper bound is rewritten by summing over t and applying an Abel transform to its first term,

$$\sum_{t=1}^T \frac{\|\boldsymbol{u}_t\|_1}{\eta}\sum_{i=1}^d q_{i,t} \ln \frac{\widehat{p}_{i,t+1}}{\widehat{p}_{i,t}} + \frac{\eta}{8}\|\boldsymbol{u}_t\|_1 + \alpha\|\boldsymbol{u}_t\|_1$$

$$
\begin{aligned}
&= \frac{\|\boldsymbol{u}_1\|_1 \ln d}{\eta} + \frac{\|\boldsymbol{u}_T\|_1}{\eta} \underbrace{\sum_{i=1}^{d} q_{i,T} \ln \widehat{p}_{i,T+1}}_{\leqslant 0} + \frac{1}{\eta} \sum_{t=2}^{T} \sum_{i=1}^{d} \underbrace{\big(\|\boldsymbol{u}_t\|_1 \, q_{i,t} - \|\boldsymbol{u}_{t-1}\| \, q_{i,t-1}\big)}_{=(1-\alpha)(u_{i,t}-u_{i,t-1})} \underbrace{\ln \frac{1}{\widehat{p}_{i,t}}}_{0 \leqslant \,\cdot\, \leqslant \ln \frac{d}{\alpha}} \\
&\quad + \Big(\frac{\eta}{8} + \alpha\Big) \sum_{t=1}^{T} \|\boldsymbol{u}_t\|_1 \\
&\leqslant \frac{\|\boldsymbol{u}_1\|_1 \ln d}{\eta} + \frac{1-\alpha}{\eta} \left(\sum_{t=2}^{T} D_{\mathrm{TV}}(\boldsymbol{u}_t, \boldsymbol{u}_{t-1}) \right) \ln \frac{d}{\alpha} + \Big(\frac{\eta}{8} + \alpha\Big) \sum_{t=1}^{T} \|\boldsymbol{u}_t\|_1 \ .
\end{aligned}
$$

4

Electricity load forecasting by aggregating experts; how to design an efficient set of experts

Short-term electricity forecasting has been studied for years at EDF and different forecasting models were developed from various fields of statistics or machine learning (functional data analysis, time series, nonparametric regression, boosting, bagging). We are interested in the forecasting of France's daily electricity consumption based on these different approaches. We investigate in this empirical study how to use them to improve prediction accuracy. First, we show how combining members of the original set of forecasts can lead to a significant improvement. Second, we explore how to build various and heterogeneous forecasts from these models and analyze how we can aggregate them to get even better predictions.

Contents

Nota: This chapter is based on the book chapter [2]. It is a joint work with Yannig Goude.

4.1. Introduction

Electricity consumption forecasting is a crucial matter for electricity providers like EDF to maintain the equilibrium between production and demand. Overestimating the consumption leads to overproduction, which has a negative environmental impact and implies unnecessary loss of benefits for the company. On the other hand, underestimating the consumption may cause a shortage of energy and black outs. In the past years EDF R&D has therefore developed several competitive forecasting models achieving around 1.4% error in MAPE (the average of percentage errors, see (4.3) for a formal definition) at the daily horizon. However the electrical scene in France is constantly evolving (nuclear power, electric cars, air conditioning are developing for instance) and the opening of the electricity market induces potential customer losses. Therefore the historical models have to be regularly reconsidered and challenged. As daily forecasts are the main inputs for optimizing the production units we consider in this chapter the goal of improving short-term (daily) forecasting of France's electricity consumption.

As the historical French electricity provider, EDF has investigated the issue of load forecasting for years and developed models from a wide range of statistical or machine learning methods. Among many, we consider in this study three approaches presented below. They were chosen for two main reasons. First, they have a good forecasting accuracy. Second, they are derived from quite different statistical frameworks, which results in a sort of heterogeneity. The first model is a nonparametric model based on regularized regression on spline basis (see Wood [147]). It will be referred to next as the generalized additive model (GAM). This model has performed well on France's electricity consumption signal (see Pierrot and Goude [121]), on EDF portfolio data (see Wood et al. [148]) and was proven to be a good competitor on US data (see Nedellec et al. [117]). The second model is based on curve linear regression (CLR) via dimension reduction. It is introduced and applied to electricity consumption forecasting in Cho et al. [50, 51]. The third and last model, kernel wavelet functional (KWF), is detailed in Antoniadis et al. [17, 18, 16]. It combines clustering functional data and detection of similar patterns in functional processes based on a wavelet distance.

These three approaches are based on extremely different insights and we expect it can induce different behaviors that an aggregation algorithm can take advantage of in some online fashion. The GAM model captures non-linear relationships between electricity load and the different covariates driving it (temperature, fare effects...) and provides smooth estimates of these transfer functions without any transformation of the original data. The CLR model performs a data-driven dimension reduction as well as a data transformation so that the relationship between the transformed data is linear and can be captured by simple multivariate regression models. The KWF approach is nonparametric and does not use any exogenous variable but the past consumption. It is particularly robust to special days (bank holidays, holiday seasons) and meteorological forecasts errors. In the GAM setting, observations (half-hourly electricity load and covariates) are considered as finite dimensional whereas in the CLR and the KWF approaches, daily electricity load is the realization of a functional process.

As we have at our disposal three forecasting models, a straightforward question is how to combine them to produce accurate forecasts. The art of combining forecasts has been extensively studied for the past four decades (see the review of Clemen [53]) and the empirical literature is voluminous. However, few real-world empirical studies consider the framework of individual sequences to design the aggregation rules. Some of them include for instance climate prediction (Monteleoni et al. [115]),

air-quality prediction (Mallet [110], Mallet et al. [111]), quantile prediction of daily call volumes entering call center (Biau and Patra [24]), or electricity consumption (Devaine et al. [60]). The vast majority of these studies focuses however on the aggregation rules and how to weight the experts. Little consideration goes into designing the set of experts to include in the combination. In their technical report Aiolfi et al. [14] studied the construction of a varied enough set of experts by considering the combination of linear autoregressive models with non-linear models (logistic smooth transition autoregressive and neural networks). They however did not consider the same aggregation rules as we do: because of the small length of their time series, none of their rules had time to learn the weights and the best results were obtained using uniform aggregation scheme.

We now describe the methodology followed in this study. We aim first at designing a set of base forecasting methods (henceforth referred to as experts) by using the three models described above. We show how an aggregation rule that sequentially outputs forecasts of the electricity consumption for the next instances can significantly improve upon these experts. The aggregation rules and the framework of prediction with expert advice is detailed in Section 4.2. Then, we propose different strategies to design a larger set of experts from the three initial experts and give a detailed analysis of the corresponding combined forecasts.

4.2. Sequential aggregation of experts

The content of this section reviews the framework of sequential prediction with expert advice, a setting which received considerable attention in the past twenty years (see the monograph by Cesa-Bianchi and Lugosi [43]). It considers an online learning scenario in which a forecaster has to guess element by element future values of an observed time series. To form its prediction it receives and combines before each instance the opinions of a finite set of experts. This framework makes possible to consider several stochastic models with extremely different assumptions in a single approach. To do so, it adopts the deterministic and robust point of view of the literature of individual sequences. It is thus particularly adapted to our application.

4.2.1. Mathematical context

We now present the mathematical setting of prediction with expert advice. We suppose that at each time instance $t = 1, ..., T$ the next outcome y_t of a sequence of observations $y_1, \dots, y_T$, like half-hourly electricity consumptions, is to be predicted. We assume that the observations are all bounded by some positive constant B, so that $y_t \in [0, B]$. Before each time instance t, a finite number K of experts provide forecasts $\boldsymbol{x}_t = (x_{1,t}, \dots, x_{K,t}) \in [0, B]^K$ of the next observation y_t. A forecaster is then asked to form its own prediction with knowledge of the past observations $y_1^{t-1} = y_1, \dots, y_{t-1}$ and of the past expert advice $\boldsymbol{x}_1^t = \boldsymbol{x}_1, \dots, \boldsymbol{x}_t$. Let denote by $\cdot$ the inner product in $\mathbb{R}^K$. Formally the forecaster forms a mixture $\widehat{\boldsymbol{p}}_t = (\widehat{p}_{1,t}, \dots, \widehat{p}_{K,t}) \in \mathbb{R}^K$ and predicts $\widehat{y}_t = \widehat{\boldsymbol{p}}_t \cdot \boldsymbol{x}_t = \sum_{k=1}^K p_{k,t} x_{k,t}$ by linearly combining the predictions of the experts.

The accuracy of a prediction x proposed by an expert or by the aggregation rule at time instance t for the outcome y_t is measured through a convex loss function ℓ_t. In this chapter, we consider the special case of the square loss $\ell_t(x) = (y_t - x)^2$. The analysis can however be easily extended to any convex loss function. On instance t, expert k suffers loss $\ell_t(x_{k,t}) = (y_t - x_{k,t})^2$ and the aggregation rule incurs loss $\ell_t(\widehat{y}_t) = (y_t - \widehat{y}_t)^2$. The goal of the forecaster is to design aggregation rules (that

is, applications $\mathcal{A} : (\boldsymbol{x}_1^t, y_1^{t-1}) \mapsto \widehat{\boldsymbol{p}}_t$) with small average error. The latter can be decomposed as

$$\frac{1}{T}\sum_{t=1}^{T}(y_t - \widehat{y}_t)^2 \quad \stackrel{\text{def}}{=} \quad \inf_{\boldsymbol{q}\in S}\left\{\frac{1}{T}\sum_{t=1}^{T}(y_t - \boldsymbol{q}\cdot\boldsymbol{x}_t)^2\right\} + R_T\,, \tag{4.1}$$

where S is some closed and bounded subset of $\mathbb{R}^K$; and this defines the regret R_T. As we explain next this decomposition highlights the well-known trade-off between approximation error and estimation error. Because these two terms add up to the error incurred by the aggregation rule they act as two opposing forces.

The first term in (4.1) is the error encountered by the best constant weight vector chosen in hindsight in a closed and bounded set $S \subset \mathbb{R}^K$. This best mixture is called an *oracle*. Its performance is the target that the aggregation rule intends to reach and is thus used as a benchmark value to be compared to the performance of an aggregation rule. Several oracles can be defined according to the set S the aggregation rule aims at competing with. We can list several oracles: the *best expert* oracle suffers $\min_{k=1,\dots,K}\sum_{t=1}^{T}(y_t - x_{k,t})^2$; the *best convex weight vector* corresponds to the best element in $S = \Delta_K \stackrel{\text{def}}{=} \{\boldsymbol{q} \in \mathbb{R}_+^K \;:\; \sum_i q_i = 1\}$; and finally the *best linear* oracle is defined by $S = B_K(r)$ the ball of radius r in $\mathbb{R}^K$. The larger the set S we aim at competing with, the smaller the first term in (4.1) is, but the harder it is for the aggregation rule to remain competitive. The second term grows in general. This approximation error is closely related to the expert forecasts. It decreases with increasing heterogeneity of the expert set.

The second term R_T is the estimation error. It evaluates the ability of the aggregation rule to retrieve online the oracle, i.e., the best possible mixture. If the aggregation rule is well designed, R_T will vanish to 0 as the length T of the experiment grows to infinity.

We assume in this chapter that we have an efficient aggregation rule and we focus on reducing the approximation error; indeed many efficient aggregation rules are already well-known— see Section 4.2.2, but the approximation error is often left out of the debate.

4.2.2. Aggregation rules

Experiments are performed by considering four different aggregation rules: the exponentially weighted average forecaster (EWA), the fixed share forecaster (FS), the ridge regression forecaster (Ridge), and the polynomially weighted average forecaster with multiple learning rates (ML-Poly). EWA, FS, and Ridge are described in the book of Cesa-Bianchi and Lugosi [43] for constant values of their learning parameters. Devaine et al. [60] already applied EWA and FS to short-term load forecasting. They suggested in Section 2.4 an empirical tuning of the learning parameters which comes with no theoretical guarantees but works empirically well. It consists of optimally choosing the learning parameters on adaptive finite grids. Except for ML-Poly which already comes with its own learning parameter calibration rule, the parameters are tuned online following the method of Devaine et al. [60].

The exponentially weighted average forecaster (EWA) is an online convex aggregation rule introduced in learning theory by Littlestone and Warmuth [107] and by Vovk [139]. At time instance t,

it assigns to expert k the weight

$$\widehat{p}_{k,t} = \frac{e^{-\eta \sum_{s=1}^{t-1} \ell_s(x_{k,s})}}{\sum_{i=1}^{K} e^{-\eta \sum_{s=1}^{t-1} \ell_s(x_{i,s})}},$$

which is exponentially small in the cumulative loss suffered so far by the expert. When the learning parameter η is properly tuned, it has a small average regret $R_T = \mathcal{O}\left(1/\sqrt{T}\right)$ with respect to the best fixed expert oracle— see Cesa-Bianchi and Lugosi [43].

The fixed share forecaster (FS) is due to Herbster and Warmuth [92]. It has the property to compete not only with the best fixed expert but with the best sequence of experts that may change a small number of times. It is particularly interesting when dealing with non stationary environments, in which the best expert should regularly be reconsidered. The fixed share forecaster considers a learning parameter η as well as a mixing parameter $\alpha \in [0, 1]$ that evaluates the number of changes in the oracle sequence of experts we are competing with.

We now provide a short mathematical description of the fixed share aggregation rule. The initial weight distribution is uniform $\widehat{p}_1 = (1/K, \ldots, 1/K)$. Then, at each instance t, the weights are updated twice. First, a *loss update* takes into account the new loss incurred by each expert,

$$\widehat{v}_{k,t} = \frac{\widehat{p}_{k,t-1} e^{-\eta \sum_{s=1}^{t-1} \ell_s(x_{k,s})}}{\sum_{i=1}^{K} \widehat{p}_{i,t-1} e^{-\eta \sum_{s=1}^{t-1} \ell_s(x_{i,s})}}.$$

Second a *mixing-update* ensures that each expert gets a minimal weight α/K by assigning

$$\widehat{p}_{k,t} = (1 - \alpha)\widehat{v}_{k,t} + \alpha/K.$$

This update captures the possibility that the best expert may have switched at time instance t. The fixed share forecaster was proven to have nice theoretical properties and vanishing average regret R_T with respect to sequences of experts with few shifts. For more details about the fixed share aggregation rule the reader is referred to Cesa-Bianchi and Lugosi [43, Section 5.2].

The polynomially weighted average forecaster with multiple learning rates (ML-Poly) is obtained via a version of the polynomially weighted average forecaster detailed in Cesa-Bianchi and Lugosi [42], see also Cesa-Bianchi and Lugosi [43, Section 2.1]. The multiple learning rate version was introduced in Chapter 2 whose implementation is recalled in Algorithm 6. In Chapter 2 we proved the regret bound $R_T = \mathcal{O}(1/\sqrt{T})$ with respect to the best fixed expert. ML-Poly is particularly interesting since despite the theoretical tuning of the learning parameters, it achieves as good performance as the other ones. It runs also much faster than the empirical tuning described by Devaine et al. [60] and used for the other rules which needs to run as many times the aggregation rule as the size of the parameter grid.

The ridge regression forecaster (Ridge) is presented in Algorithm 7. It was introduced in a stochastic setting by Hoerl and Kennard [94]. It forms at each instance the linear combination of experts minimizing a L_2-regularized least-square criterion on past data. It was first studied in the context of prediction with expert advice by Azoury and Warmuth [21] and Vovk [142] and was proved to enjoy nice theoretical properties, namely a regret bound $R_T = o(1)$ as $T \to \infty$ with respect to the best linear oracle.

Algorithm 6: The polynomially weighted average forecaster with multiple learning rates (ML-Poly)

Initialization: $\boldsymbol{p}_1 = (1/K, \ldots, 1/K)$ and $\boldsymbol{R}_0 = (0, \ldots, 0)$

For each instance $t = 1, 2, \ldots, T$

0. pick the learning rates

$$\eta_{k,t-1} = \left(1 + \sum_{s=1}^{t-1} \big(\ell_s(\widehat{y}_s) - \ell_s(x_{k,s})\big)^2\right)^{-1}$$

1. form the mixture $\widehat{\boldsymbol{p}}_t$ defined component-wise by

$$\widehat{p}_{k,t} = \frac{\eta_{k,t-1} \left(R_{k,t-1}\right)_+}{\sum_{j=1}^{K} \eta_{j,t-1} \left(R_{j,t-1}\right)_+}$$

where $(x)_+ = \max\{x, 0\}$.

2. output prediction $\widehat{y}_t = \widehat{\boldsymbol{p}}_t \cdot \boldsymbol{x}_t$
3. for each expert k update the regret

$$R_{k,t} = R_{k,t-1} + \ell_t(\widehat{y}_t) - \ell_t(x_{k,t})$$

Once again, the learning parameter λ of the ridge regression aggregation rule has to be calibrated online. This tuning can be done using the methodology detailed in Devaine et al. [60, Section 2.4].

Ridge forms linear mixtures. The weights may be negative and not sum to one, while the other three aggregation rules restrict themselves to convex combination of experts. In other words they only propose weight vectors $\widehat{\boldsymbol{p}}_t \in \Delta_K$ where $\Delta_K = \{\boldsymbol{x} \in \mathbb{R}_+^K \ : \ \sum_i x_i = 1\}$. While linear aggregation rules might have more flexibility to detect correlation between experts and therefore often reach better performance, convex aggregation offers easy interpretation and safe predictions. Indeed convex weight vectors only assign nonnegative weights to experts and their predictions always lie in the convex hull of experts predictions. Thus if all the experts are known to perform well, the aggregation rule will do so as well.

The gradient trick. In the versions described above, EWA, FS, and ML-Poly compete only with the best fixed expert oracle. In Equation (4.1) they cannot per se ensure vanishing average regret R_T with respect to the best fixed convex combination (i.e., $S = \Delta_K$). But it exists a standard reduction from the problem of competing with the best convex combination oracle to the goal of competing with the best fixed expert. This reduction is a well-known trick in the literature of individual sequences and is known as the *gradient trick*. The theoretical proof of this reduction is beyond the scope of this empirical research and is detailed in Cesa-Bianchi and Lugosi [43, Section 2.5].

We only provide a brief description of the gradient trick. For each time instance t, we denote by $f_t : \boldsymbol{p} \in \Delta_K \mapsto \ell_t(\boldsymbol{p} \cdot \boldsymbol{x}_t) \in \mathbb{R}_+$ the function which evaluates the losses incurred by the weight vectors at time instance t. When the loss functions ℓ_t are convex and (sub)differentiable, the functions f_t are convex and (sub)differentiable over Δ_K. That is the case for instance for the square loss. We

Algorithm 7: The ridge regression forecaster (Ridge)

Parameter: $\lambda > 0$
Initialization: $\widehat{\boldsymbol{p}}_0 = (1/K, \ldots, 1/K)$
For each instance $t = 1, 2, \ldots, T$
1. form the mixture $\widehat{\boldsymbol{p}}_t$ defined by

$$\widehat{\boldsymbol{p}}_t = \operatorname*{arg\,min}_{\boldsymbol{u} \in \mathbb{R}^K} \left\{ \sum_{s=1}^{t-1} \left(y_s - \boldsymbol{u} \cdot \boldsymbol{x}_s \right)^2 + \lambda \left\| \boldsymbol{u} - \boldsymbol{p}_0 \right\|_2^2 \right\}$$

2. output prediction $\widehat{y}_t = \widehat{\boldsymbol{p}}_t \cdot \boldsymbol{x}_t$

denote by ∇f_t the (sub)gradient function of f_t. The gradient trick relies then on not directly running the aggregation rule with the loss functions ℓ_t but with modified gradient loss functions $\widetilde{f}_t : \boldsymbol{p} \in \Delta_K \mapsto \nabla f_t(\widehat{\boldsymbol{p}}_t) \cdot \boldsymbol{p}$. In other words, the aggregation rules are run the same way by replacing the loss $\ell_t(\widehat{y}_t)$ incurred by the algorithm by $\widetilde{f}_t(\widehat{\boldsymbol{p}}_t)$ and the loss $\ell_t(x_{k,t})$ suffered by expert k by $\widetilde{f}_t(\delta_{k,t})$, where $\delta_k \in \Delta_K$ is the Dirac mass on k. Experiments of the next section are run using the gradient trick.

4.3. Application to electricity load forecasting

We now describe the data we are dealing with and how we intend to build new experts from the three forecasting models described in the introduction. We then report the results obtained by mixing the different sets of experts as well as the performance of three reference oracles (best experts, best convex combination, best linear combination). As explained in Section 4.2 the performance of these oracles corresponds to the one aggregation rules hope to reach. Remember that the fixed share aggregation rule does not only compete with the best fixed convex combination but has a more ambitious goal. It aims at coming close to the performance of the best sequence of convex combinations that vary slowly enough. The results obtained by this more complex oracle will however not be reported in this research and we will only compare the performance of the fixed share aggregation rule to the best fixed convex combination of experts.

4.3.1. Presentation of the data set

We consider an electricity forecasting data set which corresponds to an updated version of the one analyzed by Devaine et al. [60]. It contains half-hourly measurements of the total electricity consumption of the EDF market in France from January 1, 2008 to June 15, 2012, together with several covariates, including temperature, cloud cover, wind, etc. Our goal is to forecast the consumption every day at 12:00 for the next 24 hours; that is, for the next 48 time instances.

Atypical days are excluded from the data set. They correspond to public holidays as well as the days before and after them. Besides, the data set is cut into two subsets. A training set of 1 452 days from January 1, 2008 to August 31, 2011 is used to build the forecasting methods. The performance of the methods is then measured using the testing set of 244 days between September 1, 2011 to June 15, 2012. Prediction accuracy is measured in megawatts (MW) by the root mean squared

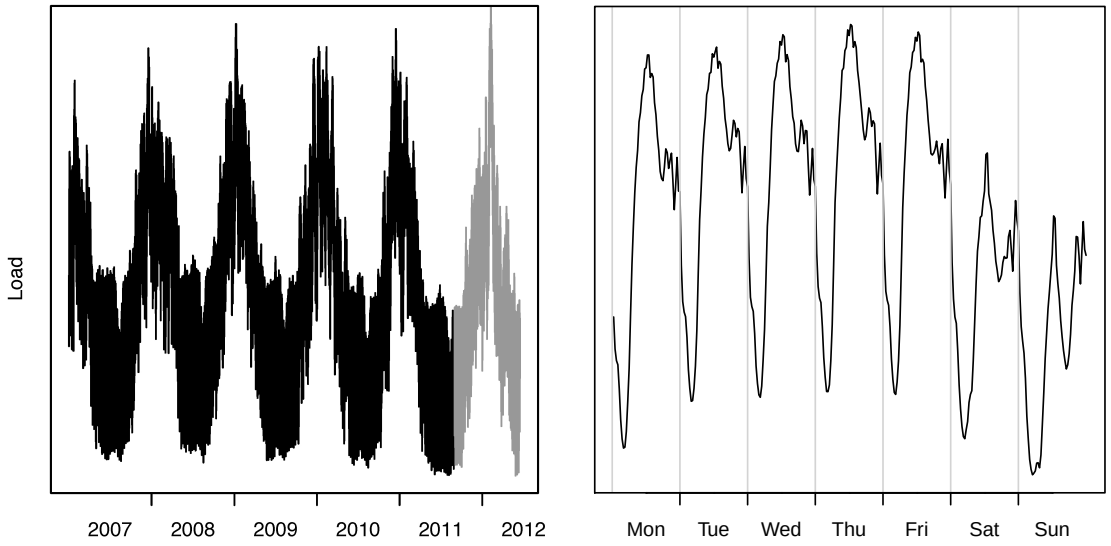


Figure 4.1.: [left] The observed half-hourly electricity consumptions between January 1, 2008 to June 15, 2012. An overall trend as well as a yearly seasonality can be pointed out in the data. The electrical heating in winter has a major impact in France on the electricity consumption. Approximately the last year is used to test the methods. [right] The observed half-hourly electricity consumptions during a typical week. A weekly pattern can be observed with a reduction of consumption during the week-end.

error (RMSE)

$$\sqrt{\frac{1}{T}\sum_{t=1}^{T}(y_t - \widehat{y}_t)^2} \tag{4.2}$$

and by the absolute percentage of error (MAPE)

$$\frac{1}{T}\sum_{t=1}^{T}\frac{|y_t - \widehat{y}_t|}{y_t}\,. \tag{4.3}$$

Operational forecasting purposes require the predictions to be made simultaneously at 12:00 for the next 24 hours (or equivalently for the next 48 half-hourly time instances). Aggregation rules can be adapted to this constraint via a generic extension detailed in Devaine et al. [60, Section 5.3].

4.3.2. Combining the three initial models

From each of the three forecasting models described in the introduction, one expert is obtained: one from the generalized additive model (GAM), one from the curve linear regression (CLR) and a last one from the kernel approach based on wavelets (KWF). The experts are trained using the total training set from January 1, 2008 to August 31, 2011 described in the previous section. We calibrate the methods as presented in Antoniadis et al. [16], Cho et al. [50], Pierrot and Goude [121]. This starting set of three experts is denoted in the rest of the chapter by E_0.

Table 4.1 reports the performance obtained by mixing the three experts in E_0. It describes also the reference results of the corresponding benchmark oracles: the best expert in E_0, the best convex combination and the best linear combination. The best convex combination and the best linear

combination obtain similar results with RMSEs of 629 MW. Due to confidentiality constraints, we cannot provide detailed characteristics of the observed electricity consumptions. The relative performance of the methods can be enjoyed by noting that MAPEs are around 1%. A significant improvement in performance can be noted in comparison to the best expert which obtains 744 MW. This motivates the necessity of mixing these models whose forecasts bring different information.

EWA, FS, and ML-Poly are designed to compete with the best convex combination of experts while Ridge aims at approaching the performance of the best linear combination. The latter suffer RMSEs between 624 MW and 638 MW, which corresponds to reductions of the RMSE of approximatively 15% compared to the best expert RMSE.

To quantify if our improvements are significant, we computed the dispersion of the errors among time instances of the aggregation rules and of the oracles— see technical report from Gaillard et al. [76, Section 1.2] for details. The dispersion is measured by the 95% standard error

$$\widehat{s}_t = \sqrt{\frac{\frac{1}{T}\sum_{t=1}^{T}\left((y_t-\widehat{y}_t)^2 - \frac{1}{T}\sum_{t=1}^{T}(y_t-\widehat{y}_t)^2\right)^2}{\frac{4}{T}\sum_{t=1}^{T}(y_t-\widehat{y}_t)^2}},$$

that is, the half-width of the 95% symmetric confidence interval of the error around the RMSEs reported in Tables 4.1 to 4.6. The 95% standard error of the RMSEs are around ten megawatts while the 95% standard error of the MAPE are approximatively 0.02%. Hence any reduction of the RMSE of more than 10 MW can be considered significant in the following.

Figure 4.2 reports the time evolution of the weights formed by ML-Poly and Ridge. The weight vectors created by Ridge converge but that is not obvious with ML-Poly. Stability is beneficial in an industrial context where weights have to be interpreted and understood by human beings. The weights formed by EWA and FS behave similarly to the ones of ML-Poly and are thus not reported here.

In the next section we will investigate how more experts can be designed based on these three models in order to improve further the predictions.

4.3.3. Creating new experts

We aim now at reducing the approximation error in Equation (4.1), i.e., at improving the performance of the oracles, by adding new experts to our initial set E_0. If the new experts are not different enough from the base ones, the approximation term will not decrease; and worse, the right-most term in (4.1), the sequential estimation error, may increase, as the aggregation rule will have to face more experts. Note that none of the newly constructed experts will significantly outperform the performance of the best expert in E_0, which achieves a RMSE of 744 MW and a MAPE of 1.29%. The benchmark performance of the best expert oracle thus remains the same for all considered extended sets of experts in this study.

Bagging

The first method that we investigate is inspired from bagging, a machine learning method that combines bootstrapping with aggregating. It was introduced by Breiman [34] in order to improve

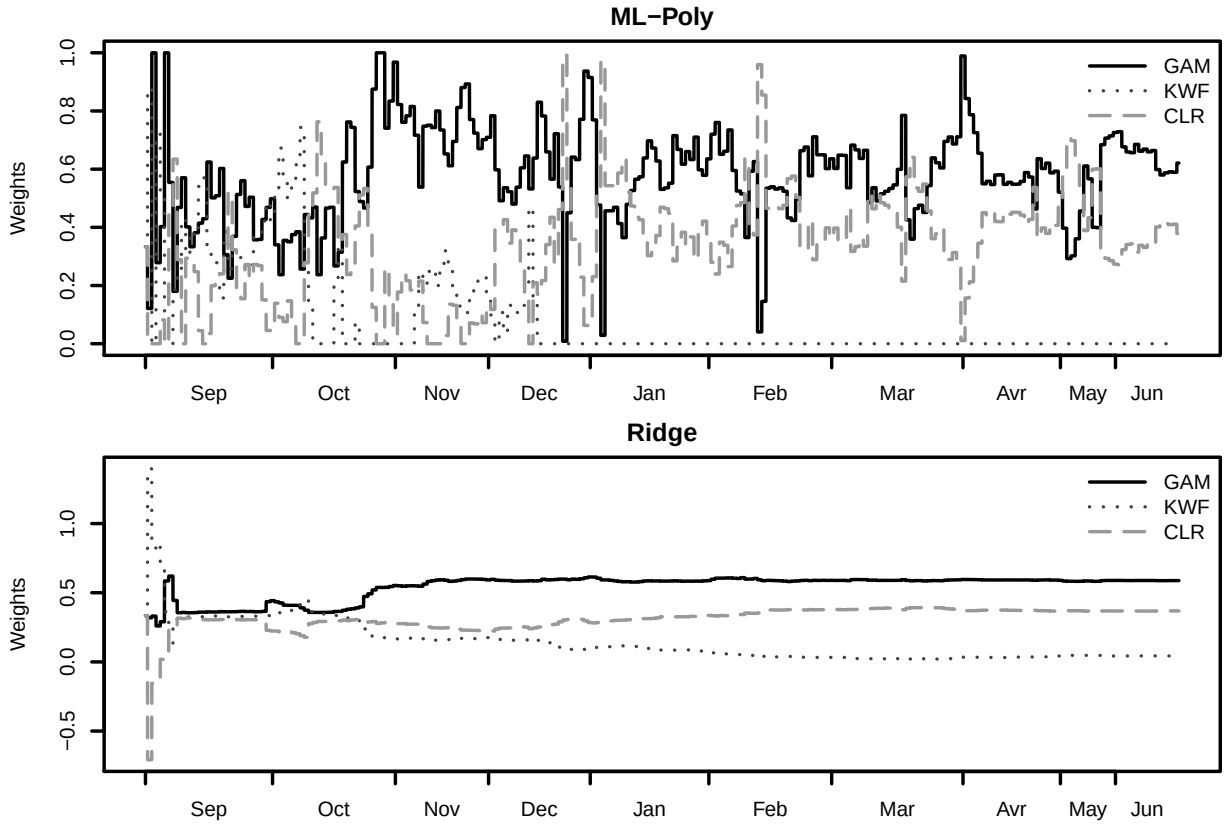


Figure 4.2.: Time evolution of the weight vectors formed by ML-Poly [top] and Ridge [bottom]. We remark that the weights assigned by ML-Poly are always nonnegative and sum to 1. Ridge can form negative weights.

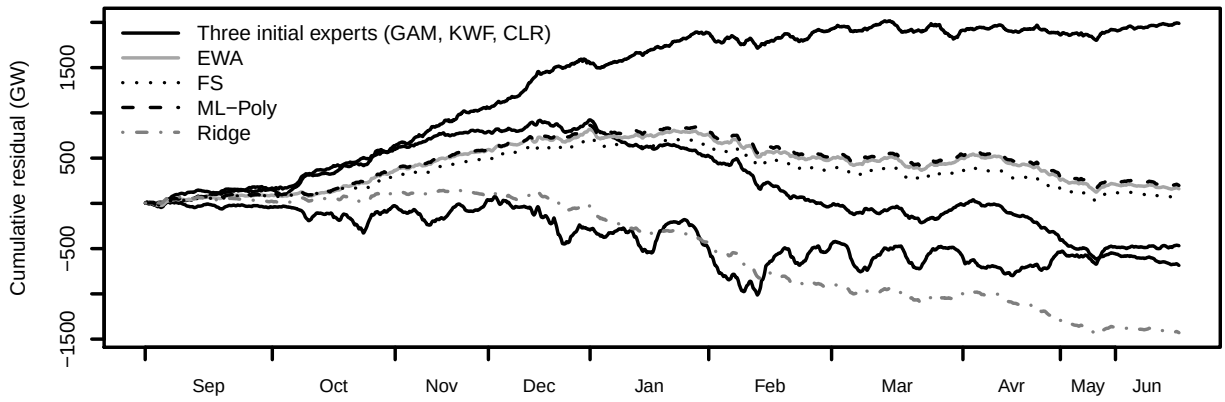


Figure 4.3.: Time evolution of cumulative residual of the three experts in E_0 and of the considered aggregation rules. The aggregation rules have smaller gradient in comparison to the experts. Besides it can be noticed that Ridge behaves very differently when compared to the other aggregation rules.

the stability and the accuracy of a forecasting model. As most averaging methods it is known to reduce the variance and to avoid over-fitting. We aim at creating new experts by bootstrapping and at averaging online the newly constructed set of experts by running the aggregation rules.

Given a forecasting model, a bootstrapped expert is obtained by estimating the model on a random

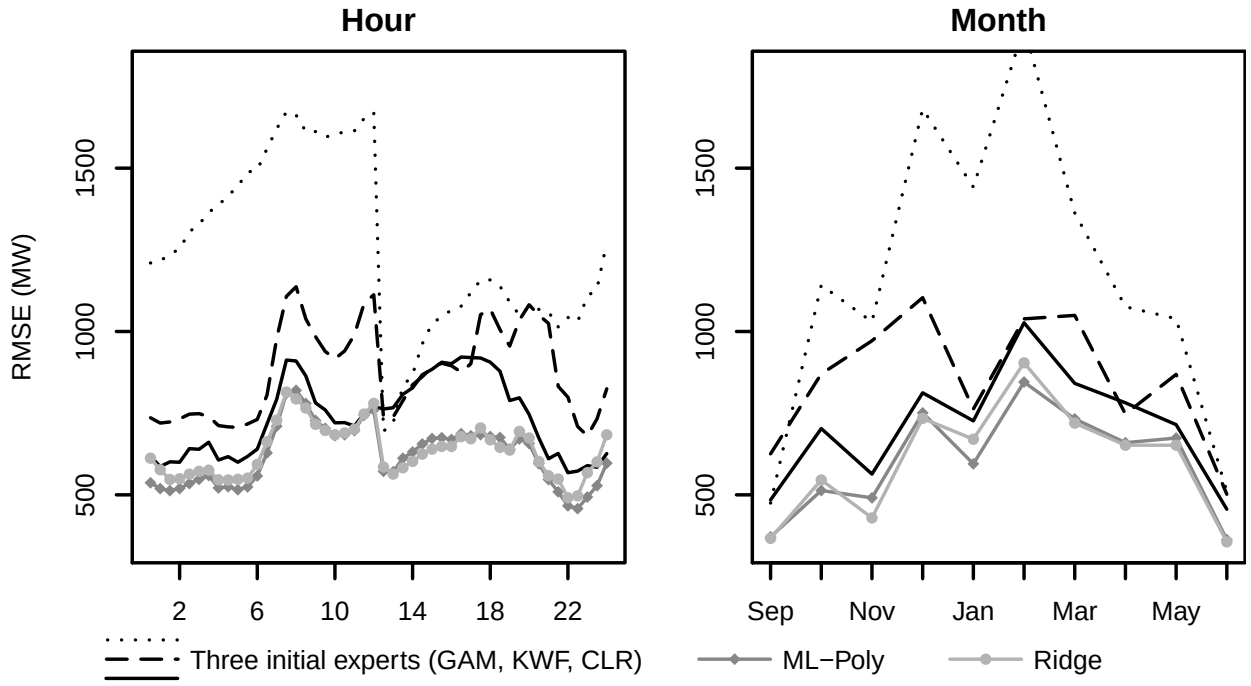


Figure 4.4.: Hourly and monthly RMSE of the first three experts and two aggregation rules described in Table 4.6. Because they obtain similar results to the ML-Poly aggregation rule, the EWA and the fixed share aggregation rules are not reported here.

training strict subset S_0' (that does not include the whole training set S_0 of $n = 1\,452$ days). The training set S_0' is generated by sampling n days from S_0 uniformly and with replacement. As the sampling is performed with replacement, some days may be present multiple times in S_0'. Breiman [34] pointed out that it leaves out $e^{-1} \approx 37\%$ of the days.

The bootstrap procedure is repeated 20 times using each of the three models at hand: GAM, CLR, and KWF. We name E_1 the set of 60 new experts, thus created. In our experiments we used 20 bootstrapped replicates of each model. This does not mean that more or fewer replicates would have led to worse performance. We wanted to add enough replicates to get sufficient variety but in the other hand we did not want to have too many bootstrapped experts in comparison to the experts we will build in the following sections. We tested several values and 20 expert replicates for each model seemed to be a reasonable trade off.

The performance of aggregation rules and oracles on $E_0 \cup E_1$ is reported in Table 4.2. The best linear combination oracle achieves a RMSE of 571 MW, which is a slightly better performance than the one of the best convex combination oracle, that equals 601 MW. This can be explained by two facts. First, the new experts might be biased. As their weights do not need to sum to one, linear mixtures correct better such bias. Second, as many experts are built using the same method, there are important correlations between them that can be better modeled using negative weights. However Ridge seems to have a hard time estimating the linear oracle and the performance is not much improved compared to Table 4.1. The empirical gain is about 10 MW for all aggregation rules. The improvement is thus not really significant.

Table 4.1.: Performance of oracles and aggregation rules using the set of experts E_0: GAM, CLR, and KWF.

Oracles and aggregation rules	RMSE (MW)	MAPE (%)
Best expert	744	1.29
Best convex combination	629	1.06
Best linear combination	629	1.06
EWA	624	1.07
FS	625	1.05
ML-Poly	626	1.05
Ridge	638	1.06

Table 4.2.: Performance of oracles and aggregation rules using the set of experts $E_0 \cup E_1$: GAM, CLR, KWF as well as the 60 bootstrapped experts.

Oracles and aggregation rules	RMSE (MW)	MAPE (%)
Best convex combination	601	1.01
Best linear combination	571	0.99
EWA	614	1.01
FS	619	1.03
ML-Poly	612	1.02
Ridge	629	1.05

Table 4.3.: Performance of oracles and aggregation rules using the set of experts $E_0 \cup E_2$: GAM, CLR, KWF as well as the 24 specialized experts.

Oracles and aggregation rules	RMSE (MW)	MAPE (%)
Best convex combination	604	1.02
Best linear combination	582	0.99
EWA	609	1.01
FS	610	1.02
ML-Poly	602	1.00
Ridge	613	1.01

Specialization

We start this section with the intuition that we need variety in our set of experts. We try to reuse the idea of bootstrapping to create new experts by modifying the training set. However, instead of sampling days uniformly in the training set E_0, we aim at assigning weights to training days with the goal to maximize the variety among themselves. To do so, we choose weights according to the values of the corresponding covariates (temperature, cloud cover, wind velocity, type of day, ...). *Specialized experts* are created this way to some specific scenarios like heatwave, cold spell, sunny days or cloudy days. Hopefully if we choose different enough scenarios, these experts may catch different effects in the consumption that we might combine by aggregating them.

We now describe how to design such new experts. We suppose that we have at our disposal a forecasting model such that, during the training of the model, we can assign different weights to the elements of the training data. This is the case for GAM, CLR, and KWF for example. We assume that we also have access to an exogenous variable $Z \in [0, 1]$ like the temperature or the cloudiness which was normalized in $[0, 1]$. Given this model and this exogenous variable Z, we build two specialized experts: the first one by assigning to the day d the weight $(1 - Z_d)^2$, the second one with the choice Z_d^2. We thus get one expert focusing on high values of Z, and another one focusing on low values. The form of these weights was set empirically but we might want to replace it by many other forms. For instance, we had first looked at weights in $\{0, 1\}$ so as to select days according to a threshold on Z. However this led to unstable experts and poor performance. We chose four covariates all based on temperature scenarios: the average, maximum, and minimum temperature of the day, and the variation of temperature with the previous day. We thus got 8 ($= 4$ scenarios $\times$ 2 experts: high and low) specialized experts by using each of the three models: GAM, CLR, and KWF. We call E_2 this set of 24 ($= 8$ experts $\times$ 3 methods) experts. The performance obtained by mixing the experts in $E_0 \cup E_2$ is reported in Table 4.3. We observe a better performance for all aggregation rules with respect to bagging although we consider fewer additional experts.

Note that we showed the interest of specialized experts when they are combined with initial experts. The individual performance of specialized experts is often poor. They do not necessarily perform better than initial experts even when they are evaluated only on the data they should be specialized to.

Temporal double-scale model

Now we study another way of constructing new experts by considering a temporal two-scale model. We follow the methodology detailed in Nedellec et al. [117] of the team TOLOLO for the "Kaggle Global Energy Forecasting Competition 2012: Load Forecasting".

To forecast the short-term load with the canonical generalized additive model (GAM), the electricity consumption is usually explained by a single model including all the covariates (meteorological, and calendar ones) together with the recent consumption. The consumption Y_t is here decomposed into two parts: a medium-term part Y_t^{mt} including meteorological and calendar effects and a short-term part Y_t^{st} containing what could not be captured in large temporal scales, $Y_t = Y_t^{mt} + Y_t^{st}$. The short-term part Y_t^{st} basically consists of capturing local effects like extreme weather, network reconfigurations and so on. The modeling approach is thus divided into two estimation steps. First, we fit a mid-term generalized additive model including the meteorological and calendar covariates

only. Second, we perform a residual analysis and we correct online the forecasts by using the observed consumptions of the last 30 days. This short-term readjusting is done by fitting another generalized additive model on the residuals.

The set containing this new expert is called E_3 and the performance obtained by combining this new expert with the three experts in E_0 is reported in Table 4.4. We observe RMSEs around 600 MW for all aggregation rules, which is a significant improvement considering that we add only one expert. The extension to other methods, like CLR and KWF, of this new way of creating experts is left for future work.

Boosting

In this section we investigate a final method to create new experts. We take now inspiration from boosting methods, like the AdaBoost algorithm of Freund and Schapire [70], that aims at correcting the mistakes of weak learners (or experts). The experts constructed in this section will be referred to as *boosted experts*.

Suppose that we have an expert that at an instance t of the training data estimates the consumption y_t by x_t. We want to build another expert predicting x'_t. Then reminding that our final aim is to aggregate well these predictions, it is irrelevant whether the second expert does not predict well y_t as soon as it counterbalances the error made by the original expert x_t. Improving the performance of the best convex combination should indeed only improve the prediction of the mixture. We can thus try to build the second expert so that the constant mixture $\gamma x_t + (1-\gamma)x'_t$ performs well for some $\gamma \in [0,1]$. This can be done by training the second experts not directly on the observed consumption y_t but on the modified one $y'_t = (y_t - \gamma x_t)/(1-\gamma)$. We can create several new experts by considering different values for $\gamma \in [0,1]$. Small values might lead to experts too similar from the original one, while larger values may create unstable experts.

We create 45 ($= 5 \times 3 \times 3$) new experts by using $\gamma \in \{0.5, 0.6, 0.7, 0.8, 0.9\}$, each of the three initial experts in E_0 are used as the original expert x_t and each of the three models (GAM, CLR, and KWF) are used to create the modified experts x'_t. We denote by E_4 the set of 45 experts thus constructed.

We report in Table 4.5 the performance obtained by mixing experts in $E_0 \cup E_4$. We did not consider $\gamma < 0.5$ because the created experts were too similar to the original ones. Considering all $\gamma \in \{0.1, \ldots, 0.9\}$ does not affect the results (neither improve nor worsen them). The step size 0.1 of the grid was chosen arbitrarily and the investigation of different values is left for future research. The best convex combination oracle achieves a RMSE of 528 MW and the best linear combination oracle suffers a RMSE of 543 MW. The performance of EWA and FS is not much improved compared to previous experiments. They both incur RMSEs of 609 MW. But ML-Poly and Ridge suffer *rmse*s under 580 MW, which is a significant improvement.

Combining the full set of experts

Table 4.6 reports the performance obtained by mixing all the experts created in the previous sections. We have now 133 experts at our disposal: 3 experts from in the starting set E_0, 60 bootstrapped experts in E_1, 24 specialized experts in E_2, 45 boosted experts in E_4 and 1 temporal

Table 4.4.: Performance of oracles and aggregation rules using the set of experts $E_0 \cup E_3$: only four experts.

Oracles and aggregation rules	RMSE (MW)	MAPE (%)
Best convex combination	596	1.00
Best linear combination	595	1.00
EWA	601	1.01
FS	599	1.00
ML-Poly	605	1.01
Ridge	605	1.00

Table 4.5.: Performance of oracles and aggregation rules using the set of experts $E_0 \cup E_4$: GAM, CLR, KWF as well as the 45 boosted experts.

Oracles and aggregation rules	RMSE (MW)	MAPE (%)
Best convex combination	543	0.93
Best linear combination	528	0.92
EWA	609	0.99
FS	609	0.99
ML-Poly	588	1.00
Ridge	578	0.98

Table 4.6.: Performance of oracles and aggregation rules using the full set of experts $E_0 \cup E_1 \cup E_2 \cup E_4 \cup E_3$: all the 133 constructed experts.

Oracles and aggregation rules	RMSE (MW)	MAPE (%)
Best convex combination	521	0.95
Best linear combination	479	0.84
EWA	578	0.95
FS	581	0.95
ML-Poly	565	0.95
Ridge	557	0.95

two-scale model in E_3. The best linear combination and the best convex combination perform better. But at the same time it is harder to compete with them. Thus while the performance of aggregation rules is improved, the gap between oracles and aggregation rules is increased as well.

Ridge suffers in Table 4.6 a RMSE of 557 MW while it got 638 MW when mixing only the three experts in E_0 (see Table 4.1). The several refinement of the set of experts thus reduced its RMSE by approximatively 13%. Similarly, the errors of EWA and FS were improved by about 7% while ML-Poly got a 10% reduction.

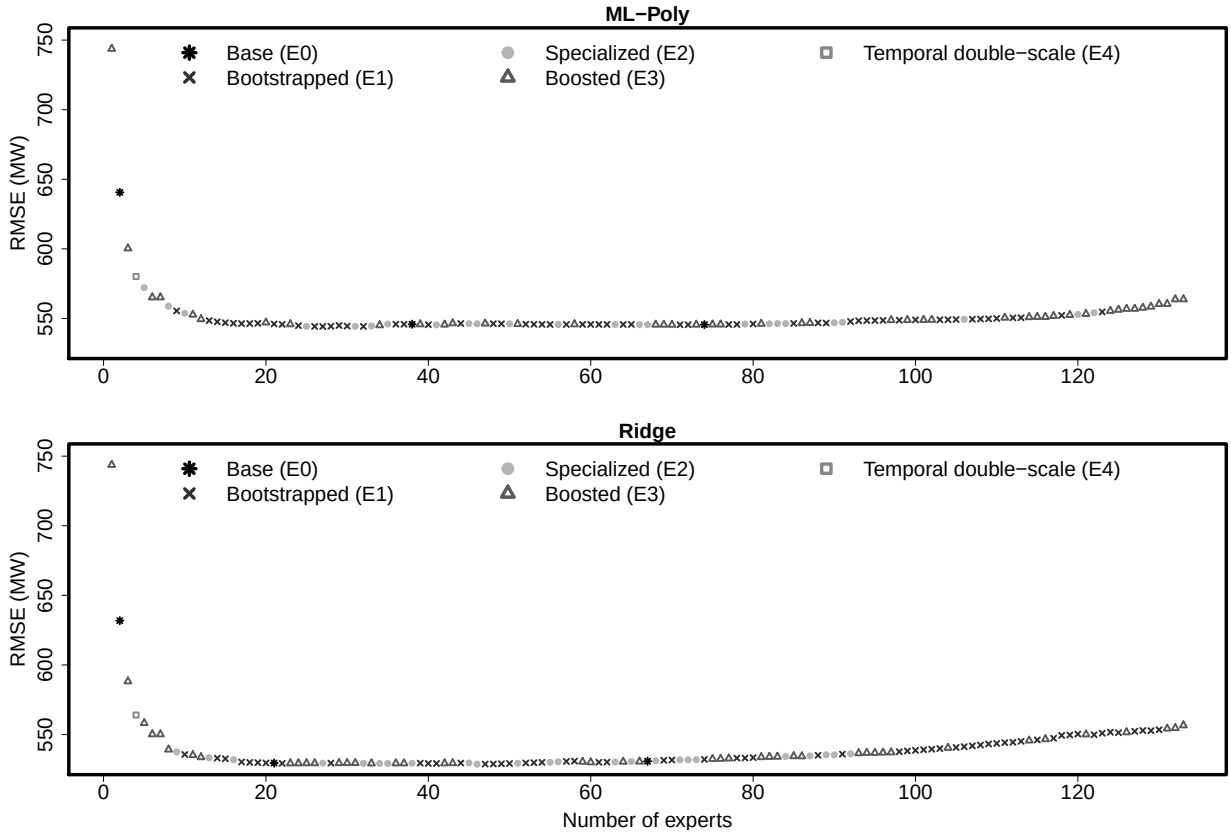


Figure 4.5.: Evolution of the performance according to the number of aggregated experts with ML-Poly [top] and Ridge [bottom].

Figure 4.5 provides the RMSEs according to the number of experts aggregated with ML-Poly and Ridge. The experts included in the mixture were chosen by induction on the number of experts by following a forward approach. The induction was initialized with the expert which performed the best (744 MW). Suppose we had a set of K experts, the expert $K+1$ was the one among the remaining experts that got the best results when it was mixed with the K experts using the considered rule. The procedure was stopped when all the 133 experts were used in the aggregation. The symbols in the figures represent the category (bootstrapped, specialized, boosting, etc.) of the last added expert.

Figure 4.5 shows the usual trade-off between having enough experts and over-fitting. If we could select a good subset of experts to include in the mixture we could reduce the RMSE under the 530 MW bar by using Ridge (and approximatively under 545 MW by using ML-Poly). A suitable number of experts seems to lie between 15 to 90 experts. In future work, a pruning step, that would

remove the less important experts before combining the forecasts of the remaining ones, might thus be a good option. Eban et al. [65] investigated in the framework of prediction of individual sequences a setting with many experts and few prediction instances. They remarked that trimming the worst experts often improves performance and suggested a procedure to do so online.

Note that the weights formed by ML-Poly and Ridge were different enough in Figure 4.2. The aggregation rules might thus capture different information and we may thus try to combine them in a second layer. The simple uniform average of the forecasts of these two rules incurs a RMSE of 541 MW, while using one of the fancier sequential aggregation rules for the second layer gets us around 548 MW.

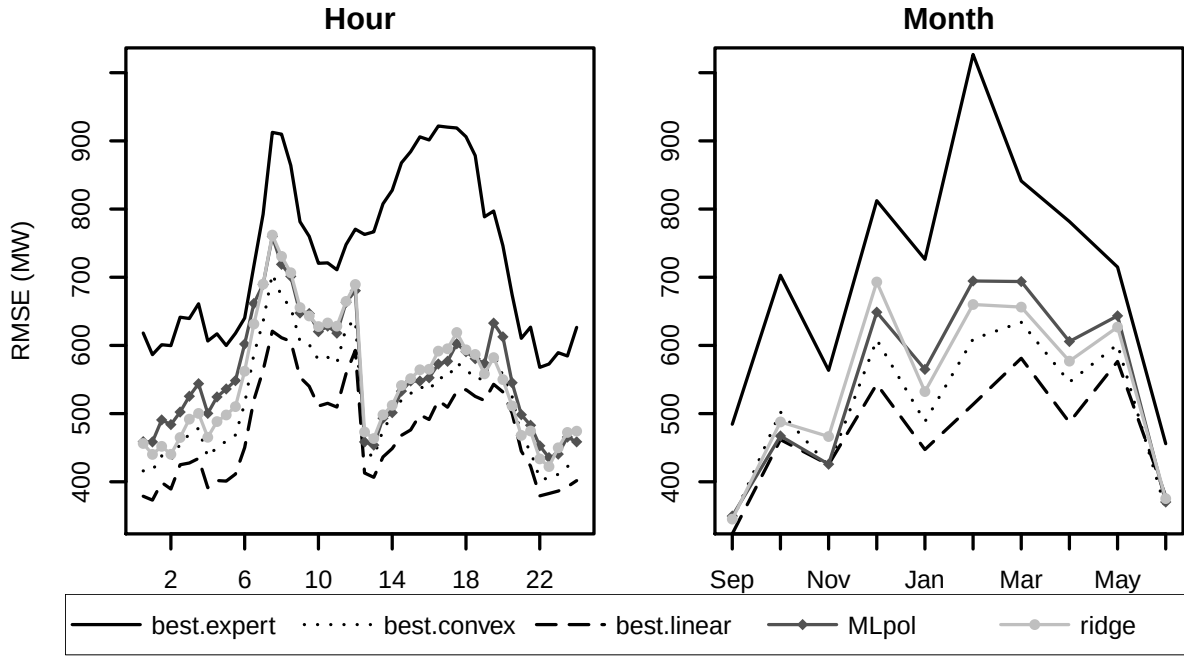


Figure 4.6.: Hourly and monthly RMSEs of the three benchmark oracles and of ML-Poly and Ridge described in Table 4.6.

Figure 4.6 plots the hourly and monthly RMSEs of the two best aggregation rules and the RMSEs of the benchmark oracles described in Table 4.6. It shows that the aggregation rules always outperform in average the best single expert at all 48 half-hours of the day and at all 10 months of the testing set. In addition, we note a significant improvement of the performance at 12:30. This can be explained by the update of the weights, which occurs at noon. The best expert oracle, which is built with a version of GAM, does not favor any hour of the day. The figure with monthly averaged RMSEs shows that aggregation rules do not only focus in improving forecasts when the task is easy. The best expert oracle is indeed outperformed every month, including November or February, which are month that are notoriously difficult to predict. Second, it illustrates that aggregation rules have a short learning period. They indeed encounter almost no regret during September and October with respect to all oracles although they just started to learn on September 1.

4.4. Conclusion

We presented in this chapter an extensive application of aggregation rules from the literature of individual sequences to short-term electrical consumption forecasting. We focused on building an efficient set of experts from three initial ones, where the efficiency is viewed in terms of what these new experts bring to the combined forecasts. In other terms, we assumed that we had an efficient aggregation rule and focused more on reducing the approximation error, that is, the first term in (4.1). We noticed that despite the vast literature on combining forecasts (including empirical studies) rare papers dealt with this important topic. We proposed different strategies to generate experts from the three initial approaches: KWF, GAM, and CLR. We then quantified the gains in terms of forecast accuracy of the combined forecasts on the test set (about 10 month of half-hourly data). A summary of our results is presented in Figure 4.7 for the two best aggregation rules: ML-Poly and Ridge. Combining all the experts that we generated with 4 different strategies, we achieved a 25% gain over the best expert (around 200 MW in RMSE), which is a significant gain considering that the three original experts had already been refined and worked extremely well (they are not week learners as in classical boosting). This gain can be decomposed into two parts: roughly half of it comes from combining three heterogeneous initial experts, the other half is due to the construction of new experts. Among the four proposed strategies, our boosting trick and what we call specialized experts bring the most important improvements. We believe that these strategies could be applied to other forecasting problems and there is still some work to derive theoretical guarantees for these tricks. We also observe that aggregating rules are quite robust to adding new experts, and it is clear in Figure 4.5 that combining forecasts does not suffer much from over fitting. Nevertheless, these results suggest that there is a way for improving the aggregation rules accuracy by adding a pruning step that could select the best set of experts in some online fashion.

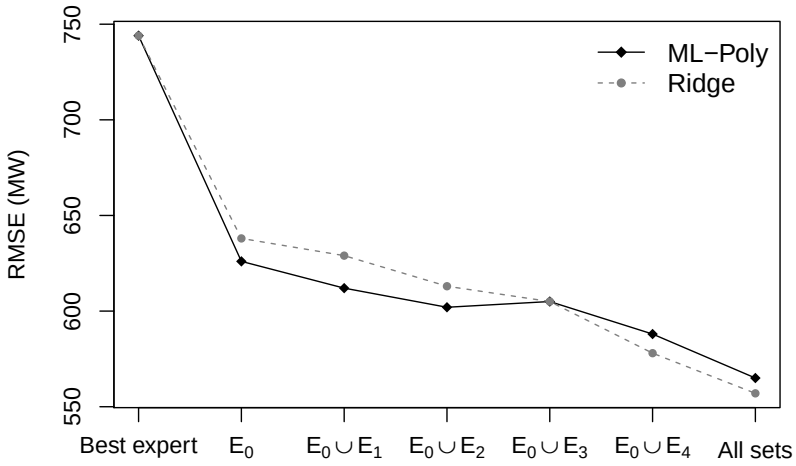


Figure 4.7.: RMSEs suffered by combining experts in E_0, $E_0 \cup E_1$, ..., $E_0 \cup E_4$ by using ML-Poly and Ridge. The performance of the best expert in E_0, and the final performance obtained by mixing all the experts in $E_0 \cup \cdots \cup E_4$ (referred to as 'All sets') are also reported.

II

Online nonparametric robust aggregation

Introduction

In this part we investigate nonparametric aggregation in the setting of sequential prediction of arbitrary sequences. At each time step $t = 1, \dots, T$, after the player observes the vector of expert advice $\boldsymbol{x}_t = (x_{1,t}, \dots, x_{K,t})$, he aims at forming forecast $\widehat{y}_t$ such that the cumulative regret

$$\underbrace{R_T}_{\text{regret}} \stackrel{\text{def}}{=} \underbrace{\sum_{t=1}^{T} \ell\big(\widehat{y}_t, y_t\big)}_{\text{performance of the player}} - \underbrace{\inf_{f \in \mathcal{F}} \sum_{t=1}^{T} \ell\big(f(\boldsymbol{x}_t), y_t\big)}_{\text{approximation error}} \tag{R2}$$

is as small as possible, where ℓ is a nonnegative loss function and $\mathcal{F}$ is some class of functions. In other words, the class of reference strategies the player is compared with consists of all fixed functions in $\mathcal{F}$.

In Chapter 5 the loss function ℓ is (mainly) the square loss. We design the first efficient strategy that achieves the (almost) optimal rate of convergence for R_T in the case of Hölder classes. The strategy borrows ideas from the chaining technique and our regret bound is expressed in terms of the metric entropy in the sup norm.

In Chapter 6 we focus on Lipschitz comparison classes with a convex loss function ℓ. We construct a fully data-driven calibrated strategy that uses ideas from CART regression trees while ensuring robust finite time guarantees. Although the regret bounds are stated uniformly over all possible data, the data-driven calibration of the parameters adapts well to the inherent difficulty of the problem and yields a significant gain in performance. In the end, we justify our initial goal to design strategies minimizing the cumulative regret (R2). We show that when the data is generated by some underlying ergodic process, ensuring small cumulative regret yields asymptotic optimality.

5

A chaining algorithm for online nonparametric regression

We consider the problem of online nonparametric regression with arbitrary deterministic sequences. Using ideas from the chaining technique, we design an algorithm that achieves a Dudley-type regret bound similar to the one obtained in a non-constructive fashion by Rakhlin and Sridharan [124]. Our regret bound is expressed in terms of the metric entropy in the sup norm, which yields optimal guarantees when the metric and sequential entropies are of the same order of magnitude. In particular our algorithm is the first one that achieves optimal rates for online regression over Hölder balls. In addition we show for this example how to adapt our chaining algorithm to get a reasonable computational efficiency with similar regret guarantees (up to a log factor).

Contents

Nota: This chapter is a joint work with Sébastien Gerchinovitz. It is based on the submitted paper [4].

5.1. Introduction

We consider the setting of online nonparametric regression for arbitrary deterministic sequences, which unfolds as follows. First, the environment chooses a sequence of observations $(y_t)_{t\geqslant 1}$ in $\mathbb{R}$ and a sequence of input vectors $(x_t)_{t\geqslant 1}$ in $\mathcal{X}$, both initially hidden from the forecaster. At each time instant $t \in \mathbb{N}^* = \{1, 2, \ldots\}$, the environment reveals the data $x_t \in \mathcal{X}$; the forecaster then gives a prediction $\widehat{y}_t \in \mathbb{R}$; the environment in turn reveals the observation $y_t \in \mathbb{R}$; and finally, the forecaster incurs the square loss $(y_t - \widehat{y}_t)^2$.

The term online *nonparametric* regression means that we are interested in forecasters whose regret

$$\mathrm{Reg}_T(\mathcal{F}) \stackrel{\text{def}}{=} \sum_{t=1}^{T} \big(y_t - \widehat{y}_t\big)^2 - \inf_{f\in\mathcal{F}} \sum_{t=1}^{T} \big(y_t - f(x_t)\big)^2$$

over standard nonparametric function classes $\mathcal{F} \subseteq \mathbb{R}^{\mathcal{X}}$ is as small as possible. In this chapter we design and study an algorithm that achieves a regret bound of the form

$$\mathrm{Reg}_T(\mathcal{F}) \leqslant c_1 B^2 \big(1 + \log \mathcal{N}_\infty(\mathcal{F}, \gamma)\big) + c_2 B\sqrt{T} \int_0^\gamma \sqrt{\log \mathcal{N}_\infty(\mathcal{F}, \varepsilon)}\, \mathrm{d}\varepsilon \;, \tag{5.1}$$

for all $\gamma \in \big(\frac{B}{T}, B\big)$, where B is an upper bound on $\max_{1\leqslant t\leqslant T} |y_t|$ and where $\log \mathcal{N}_\infty(\mathcal{F}, \varepsilon)$ denotes the metric entropy of the function set $\mathcal{F}$ in the sup norm at scale ε (cf. Section 5.1.4).

The integral on the right-hand side of (5.1) is very close to what is known in probability theory as *Dudley's entropy integral*, a useful tool to upper bound the expectation of a centered stochastic process with subgaussian increments (see, e.g., Talagrand [134], Boucheron et al. [31]). In statistical learning (with i.i.d. data), Dudley's entropy integral is key to derive risk bounds on empirical risk minimizers; see, e.g., Massart [113], Rakhlin et al. [125].

Very recently Rakhlin and Sridharan [124] showed that the same type of entropy integral appears naturally in regret bounds for online nonparametric regression. The most part of their analysis is non-constructive in the sense that their regret bounds are obtained without explicitly constructing an algorithm. One of our main contributions is to provide an explicit algorithm that achieves the regret bound (5.1). We note however that our regret bounds are in terms of a weaker notion of entropy, namely, metric entropy instead of the smaller (and optimal) sequential entropy. Fortunately, both notions are of the same order of magnitude for a reasonable number of examples, such as the ones outlined just below. We leave the question of modifying our algorithm to get sequential entropy regret bounds for future work.

The regret bound (5.1)—that we call *Dudley-type regret bound* thereafter—can be used to obtain optimal regret bounds for several classical nonparametric function classes. Indeed, when $\mathcal{F}$ has a metric entropy $\log \mathcal{N}_\infty(\mathcal{F}, \varepsilon) \leqslant C_p \varepsilon^{-p}$ with* $p \in (0, 2)$, the bound (5.1) entails

$$\begin{aligned}\mathrm{Reg}_T(\mathcal{F}) &\leqslant c_1 B^2 + c_1 B^2 C_p \gamma^{-p} + c_2 B\sqrt{C_p T} \int_0^\gamma \varepsilon^{-p/2}\, \mathrm{d}\varepsilon \\ &= c_1 B^2 + c_1 B^2 C_p \gamma^{-p} + \frac{2\, c_2 B}{2-p} \sqrt{C_p T}\, \gamma^{1-p/2} = \mathcal{O}\big(T^{p/(p+2)}\big)\end{aligned} \tag{5.2}$$

*When $p > 2$, we can also derive Dudley-type regret bounds that lead to a regret of $\mathcal{O}\big(T^{-1/p}\big)$ in the same spirit as in Rakhlin and Sridharan [124]. We omitted this case to ease the presentation.

for the choice of $\gamma = \Theta\big(T^{-1/(p+2)}\big)$. An example is given by Hölder classes $\mathcal{F}$ with regularity $\beta > 1/2$ (cf. Tsybakov [137, Def 1.2]). We know from Kolmogorov and Tikhomirov [102] or Lorentz [108, Theorem 2] that they satisfy $\log \mathcal{N}_\infty(\mathcal{F}, \varepsilon) = \mathcal{O}\big(\varepsilon^{-1/\beta}\big)$. Therefore, (5.2) entails a regret bound $\mathrm{Reg}_T(\mathcal{F}) = \mathcal{O}\big(T^{1/(2\beta+1)}\big)$, which is in a way optimal since it corresponds to the optimal (minimax) quadratic risk $T^{-2\beta/(2\beta+1)}$ in statistical estimation with i.i.d. data.

5.1.1. Why a simple Exponentially Weighted Average forecaster is not sufficient

A natural approach (see Vovk [144]) to compete against a nonparametric class $\mathcal{F}$ relies on running an Exponentially Weighted Average forecaster (EWA, see Cesa-Bianchi and Lugosi [43, p.14]) on an ε-net $\mathcal{F}^{(\varepsilon)}$ of $\mathcal{F}$ of finite size $\mathcal{N}_\infty(\mathcal{F}, \varepsilon)$. This yields a regret bound of order $\varepsilon T + \log \mathcal{N}_\infty(\mathcal{F}, \varepsilon)$. The first term εT is due to the approximation of $\mathcal{F}$ by $\mathcal{F}^{(\varepsilon)}$, while the second term is the regret suffered by EWA on the finite class of experts $\mathcal{F}^{(\varepsilon)}$. As noted by Rakhlin and Sridharan [124, Remark 11], the above regret bound is suboptimal for large nonparametric classes $\mathcal{F}$. Indeed, for a metric entropy of order ε^{-p} with $p \in (0, 2)$, optimizing the above regret bound in ε entails a regret of order $\mathcal{O}(T^{p/(p+1)})$ when (5.1) yields the better rate $\mathcal{O}(T^{p/(p+2)})$.

5.1.2. The chaining technique: a brief reminder

The idea of chaining was introduced by Dudley [64]. It provides a general method to bound the supremum of stochastic processes. For the convenience of the reader, we recall the main ideas underlying this technique; see, e.g., Boucheron et al. [31] for further details. We consider a centered stochastic process $(X_f)_{f\in\mathcal{F}}$ indexed by some finite metric space, say, $(\mathcal{F}, \|\cdot\|_\infty)$, with subgaussian increments, which means that $\log \mathbb{E} e^{\lambda(X_f - X_g)} \leqslant \frac{1}{2} v\lambda^2 \|f-g\|_\infty^2$ for all $\lambda > 0$ and all $\forall f, g \in \mathcal{F}$. The goal is to bound the quantity

$$\mathbb{E}\big[\sup_{f\in\mathcal{F}} X_f\big] = \mathbb{E}\big[\sup_{f\in\mathcal{F}} (X_f - X_{f_0})\big]$$

for any $f_0 \in \mathcal{F}$.

Lemma 5.1. — Boucheron et al. [31]. *Let $Z_1, \dots, Z_K$ be subgaussian random variables with parameter $v > 0$ (i.e., $\log \mathbb{E}\exp(\lambda Z_i) \leqslant \lambda^2 v/2$ for all $\lambda \in \mathbb{R}$), then $\mathbb{E}\max_{i=1,\dots,K} Z_i \leqslant \sqrt{2v \log K}$.*

Lemma 5.1 entails $\mathbb{E}\big[\sup_{f\in\mathcal{F}}(X_f - X_{f_0})\big] \leqslant B\sqrt{2v \log(\operatorname{card}\mathcal{F})}$, where $B = \sup_{f\in\mathcal{F}} \|f - f_0\|_\infty$. However, this bound is too crude since X_f and X_g are very correlated when f and g are very close. The chaining technique takes this remark into account by approximating the maximal value $\sup_f X_f$ by maxima over successive refining discretizations $\mathcal{F}^{(0)}, \dots, \mathcal{F}^{(K)}$ of $\mathcal{F}$. More formally, for any $f \in \mathcal{F}$, we consider a sequence of approximations $\pi_0(f) = f_0 \in \mathcal{F}^{(0)}, \pi_1(f) \in \mathcal{F}^{(1)}, \dots, \pi_K(f) = f \in \mathcal{F}^{(K)}$, where $\|f - \pi_k(f)\|_\infty \leqslant B/2^k$ and $\operatorname{card}\mathcal{F}^{(k)} = \mathcal{N}_\infty(\mathcal{F}, B/2^k)$, so that:

$$\begin{aligned}\mathbb{E}\Big[\sup_{f\in\mathcal{F}}(X_f - X_{f_0})\Big] &= \mathbb{E}\left[\sup_{f\in\mathcal{F}} \sum_{k=0}^{K-1} \Big(X_{\pi_{k+1}(f)} - X_{\pi_k(f)}\Big)\right] \\ &\leqslant \sum_{k=0}^{K-1} \mathbb{E}\left[\sup_{f\in\mathcal{F}} \Big(X_{\pi_{k+1}(f)} - X_{\pi_k(f)}\Big)\right],\end{aligned}$$

We apply Lemma 5.1 for each $k \in \{0, \dots, K-1\}$: since $\|\pi_{k+1}(f) - \pi_k(f)\|_\infty \leqslant 3B/2^{k+1}$ (by the triangle inequality) and $\operatorname{card}\{\pi_{k+1}(f) - \pi_k(f), f \in \mathcal{F}\} \leqslant \mathcal{N}_\infty(\mathcal{F}, B/2^{k+1})^2$, we get the well-known Dudley entropy bound (note that $\varepsilon \mapsto \mathcal{N}_\infty(\mathcal{F}, \varepsilon)$ is nonincreasing):

$$\begin{aligned}\mathbb{E}\Big[\sup_{f\in\mathcal{F}}(X_f - X_{f_0})\Big] &\leqslant 6\sum_{k=0}^{K-1} B2^{-k-1}\sqrt{v\log\mathcal{N}_\infty(\mathcal{F}, B/2^{k+1})} \\ &\leqslant 12\sqrt{v}\int_0^{B/2}\sqrt{\log\mathcal{N}_\infty(\mathcal{F},\varepsilon)}\,\mathrm{d}\varepsilon\ .\end{aligned}$$

5.1.3. Turning the chaining technique into an online algorithm

We explained in Section 1.1 that using an Exponentially Weighted Average forecaster is not sufficient to derive a Dudley-type regret bound (5.1). It turns out that the chaining technique can be used for that purpose. Rakhlin and Sridharan [124] already used it in their analysis to obtain a non-constructive Dudley-type regret bound. Next we briefly explain how to adapt the chaining principle in order to build an algorithm. We approximate any function $f \in \mathcal{F}$ by a sequence of refining approximations $\pi_0(f) \in \mathcal{F}^{(0)}, \pi_1(f) \in \mathcal{F}^{(1)}, \dots$, such that for all $k \geqslant 0$, $\sup_f \|\pi_k(f) - f\|_\infty \leqslant \gamma/2^k$ and $\operatorname{card}\mathcal{F}^{(k)} = \mathcal{N}_\infty(\mathcal{F}, \gamma/2^k)$, so that:

$$\inf_{f\in\mathcal{F}}\sum_{t=1}^T \big(y_t - f(x_t)\big)^2 = \inf_{f\in\mathcal{F}}\sum_{t=1}^T\Bigg(y_t - \pi_0(f)(x_t) - \sum_{k=0}^{\infty}\underbrace{\big[\pi_{k+1}(f) - \pi_k(f)\big](x_t)}_{\|\cdot\|_\infty\leqslant 3\gamma/2^{k+1}}\Bigg)^2 .$$

We use the above decomposition in Algorithm 9 (Section 5.2.2) by performing two simultaneous aggregation tasks at two different scales:

- high-scale aggregation: we run an Exponentially Weighted Average forecaster to be competitive against every function $\pi_0(f)$ in the coarsest set $\mathcal{F}^{(0)}$;
- low-scale aggregation: we run in parallel many instances of (an extension of) the Exponentiated Gradient (EG) algorithm so as to be competitive against the increments $\pi_{k+1}(f) - \pi_k(f)$. The advantage of using EG is that even if the number $N^{(k)}$ of increments $\pi_{k+1}(f) - \pi_k(f)$ is large for small scales ε, the size of the gradients is very small, hence a manageable regret.

At the core of the algorithm lies the Multi-variable Exponentiated Gradient algorithm (Algorithm 8) that makes it possible to perform low-scale aggregation at all scales $\varepsilon < \gamma$ simultaneously.

Main contributions and outline of the chapter Our contributions are threefold: we first design the Multi-variable Exponentiated Gradient algorithm (Section 5.2.1). We then present our main algorithm and derive a Dudley-type regret bound as in (5.1) (Section 5.2.2). Finally, in Section 5.3 we address computational issues in the case of Hölder classes. Some proofs are postponed to the appendix.

5.1.4. Some useful definitions

Let $\mathcal{F} \subseteq \mathbb{R}^{\mathcal{X}}$ be a set of bounded functions endowed with the sup norm $\|f\|_\infty \stackrel{\text{def}}{=} \sup_{x\in\mathcal{X}}|f(x)|$. For all $\varepsilon > 0$, we call *proper ε-net* any subset $\mathcal{G} \subseteq \mathcal{F}$ such that $\forall f \in \mathcal{F},\ \exists g \in \mathcal{G} :\ \|f - g\|_\infty \leqslant \varepsilon$. (If $\mathcal{G} \not\subseteq \mathcal{F}$, we call it *non-proper*.) The cardinality of the smallest proper ε-net is denoted by $\mathcal{N}_\infty(\mathcal{F}, \varepsilon)$,

and the logarithm $\log \mathcal{N}_\infty(\mathcal{F}, \varepsilon)$ is called the *metric entropy of* $\mathcal{F}$ *at scale* ε. When this quantity is finite for all $\varepsilon > 0$, we say that $\big(\mathcal{F}, \|\cdot\|_\infty\big)$ is *totally bounded*.

5.2. The Chaining Exponentially Weighted Average Forecaster

In this section we design an online algorithm—the *Chaining Exponentially Weighted Average forecaster*—that achieves the Dudley-type regret bound (5.1). In Section 5.2.1 below, we first define a subroutine that will prove crucial in our analysis, and whose applicability may extend beyond this chapter.

5.2.1. Preliminary: the Multi-variable Exponentiated Gradient Algorithm

Let $\Delta_N \stackrel{\text{def}}{=} \Big\{ \boldsymbol{u} \in \mathbb{R}_+^N : \sum_{i=1}^N u_i = 1 \Big\} \subseteq \mathbb{R}^N$ denotes the simplex in $\mathbb{R}^N$. In this subsection we define and study a new extension of the Exponentiated Gradient algorithm [97, 41]. This extension is meant to minimize a sequence of multi-variable loss functions $\big(\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(K)}\big) \mapsto \ell_t\big(\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(K)}\big)$ simultaneously over all the variables $(\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(K)}) \in \Delta_{N_1} \times \ldots \times \Delta_{N_K}$.

Our algorithm is described as Algorithm 8 below. We call it *Multi-variable Exponentiated Gradient*. When $K = 1$, it boils down to the classical Exponentiated Gradient algorithm over the simplex Δ_{N_1}. But when $K \geqslant 2$, it performs K simultaneous optimization updates (one for each direction $\boldsymbol{u}^{(k)}$) that lead to a global optimum by joint convexity of the loss functions ℓ_t.

Algorithm 8: Multi-variable Exponentiated Gradient

---

input : optimization domain $\Delta_{N_1} \times \ldots \times \Delta_{N_K}$ and tuning parameters $\eta^{(1)}, \dots, \eta^{(K)} > 0$.

initialization: set $\widehat{\boldsymbol{u}}_1^{(k)} \stackrel{\text{def}}{=} \big(\frac{1}{N_k}, \dots, \frac{1}{N_k}\big) \in \Delta_{N_k}$ for all $k = 1, \dots, K$.

for *each round* $t = 1, 2, \dots$ **do**

- Output $\big(\widehat{\boldsymbol{u}}_t^{(1)}, \dots, \widehat{\boldsymbol{u}}_t^{(K)}\big) \in \Delta_{N_1} \times \ldots \times \Delta_{N_K}$ and observe the differentiable and jointly convex loss function $\ell_t : \Delta_{N_1} \times \ldots \times \Delta_{N_K} \to \mathbb{R}$.
- Compute the new weight vectors $\big(\widehat{\boldsymbol{u}}_{t+1}^{(1)}, \dots, \widehat{\boldsymbol{u}}_{t+1}^{(K)}\big) \in \Delta_{N_1} \times \ldots \times \Delta_{N_K}$ as follows:

$$\widehat{\boldsymbol{u}}_{t+1,i}^{(k)} \stackrel{\text{def}}{=} \frac{\exp\left(-\eta^{(k)} \sum_{s=1}^{t} \partial_{\widehat{u}_{s,i}^{(k)}} \ell_s\Big(\widehat{\boldsymbol{u}}_s^{(1)}, \dots, \widehat{\boldsymbol{u}}_s^{(K)}\Big) \right)}{Z_{t+1}^{(k)}} \,, \quad i \in \{1, \dots, N_k\},$$

where $\partial_{\widehat{u}_{s,i}^{(k)}} \ell_s$ is the partial derivative of ℓ_s with respect to the i-th component of $\widehat{\boldsymbol{u}}_s^{(k)}$, and where the normalizing factor $Z_{t+1}^{(k)}$ is defined by

$$Z_{t+1}^{(k)} \stackrel{\text{def}}{=} \sum_{i=1}^{N_k} \exp\left(-\eta^{(k)} \sum_{s=1}^{t} \partial_{\widehat{u}_{s,i}^{(k)}} \ell_s\Big(\widehat{\boldsymbol{u}}_s^{(1)}, \dots, \widehat{\boldsymbol{u}}_s^{(K)}\Big) \right) .$$

end

The Multi-variable Exponentiated Gradient algorithm satisfies the regret bound of Theorem 5.2 below. We first need some notations. We define the *partial gradients*

$$\nabla_{\boldsymbol{u}^{(k)}} \ell_t = \left(\partial_{u_1^{(k)}} \ell_t, \dots, \partial_{u_{N_k}^{(k)}} \ell_t \right) , \qquad 1 \leqslant k \leqslant K ,$$

where $\partial_{u_i^{(k)}} \ell_t$ denotes the partial derivative of ℓ_t with respect to the scalar variable $u_i^{(k)}$. Note that $\nabla_{\boldsymbol{u}^{(k)}} \ell_t$ is a function that maps $\Delta_{N_1} \times \ldots \times \Delta_{N_K}$ to $\mathbb{R}^{N_k}$. Next we also use the notation

$$\|\varphi\|_\infty \stackrel{\text{def}}{=} \sup_{\boldsymbol{u}^{(1)},\dots,\boldsymbol{u}^{(K)}} \max_{1 \leqslant i \leqslant N_k} \left| \varphi_i\left(\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(K)}\right) \right|$$

for the sup norm of any vector-valued function $\varphi : \Delta_{N_1} \times \ldots \times \Delta_{N_K} \to \mathbb{R}^{N_k}$, $1 \leqslant k \leqslant K$.

Theorem 5.2. *Assume that the loss functions $\ell_t : \Delta_{N_1} \times \ldots \times \Delta_{N_K} \to \mathbb{R}$, $t \geqslant 1$, are differentiable and jointly convex. Assume also the following upper bound on their partial gradients: for all $k \in \{1, \dots, K\}$,*

$$\max_{1 \leqslant t \leqslant T} \|\nabla_{\boldsymbol{u}^{(k)}} \ell_t\|_\infty \leqslant G^{(k)} . \tag{5.3}$$

Then, the Multi-variable Exponentiated Gradient algorithm (Algorithm 8) tuned with the parameters $\eta^{(k)} = \sqrt{2\log(N_k)/T}\,/G^{(k)}$ has a regret bounded as follows:

$$\sum_{t=1}^T \ell_t\left(\widehat{\boldsymbol{u}}_t^{(1)}, \dots, \widehat{\boldsymbol{u}}_t^{(K)}\right) - \min_{\boldsymbol{u}^{(1)},\dots,\boldsymbol{u}^{(K)}} \sum_{t=1}^T \ell_t\left(\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(K)}\right) \leqslant \sqrt{2T} \sum_{k=1}^K G^{(k)} \sqrt{\log N_k} ,$$

where the minimum is taken over all $\left(\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(K)}\right) \in \Delta_{N_1} \times \ldots \times \Delta_{N_K}$.

The proof of Theorem 5.2 is postponed to Appendix 5.C.1.

5.2.2. The Chaining Exponentially Weighted Average Forecaster

In this section we introduce our main algorithm: the *Chaining Exponentially Weighted Average forecaster*. A precise definition will be given in Algorithm 9 below. For the sake of clarity, we first describe the main ideas underlying this algorithm.

Recall that we aim at proving a regret bound of the form (5.1), whose right-hand side consists of two main terms:

$$B^2 \log \mathcal{N}_\infty(\mathcal{F}, \gamma) \qquad \text{and} \qquad B\sqrt{T} \int_0^\gamma \sqrt{\log \mathcal{N}_\infty(\mathcal{F}, \varepsilon)}\, \mathrm{d}\varepsilon .$$

Our algorithm performs aggregation at two different levels: one level (at all scales $\varepsilon \in (0, \gamma]$) to get the entropy integral above, and another level (at scale γ) to get the other term $B^2 \log \mathcal{N}_\infty(\mathcal{F}, \gamma)$. More precisely:

- for all $k \in \mathbb{N}$, let $\mathcal{F}^{(k)}$ be a proper $\gamma/2^k$-net of $(\mathcal{F}, \|\cdot\|_\infty)$ of minimal cardinality[†] $\mathcal{N}_\infty\left(\mathcal{F}, \gamma/2^k\right)$;

[†]We assume that $(\mathcal{F}, \|\cdot\|_\infty)$ is totally bounded.

- for all $k \geqslant 1$, set $\mathcal{G}^{(k)} \stackrel{\text{def}}{=} \{\pi_k(f) - \pi_{k-1}(f) : \ f \in \mathcal{F}\}$, where

$$\forall f \in \mathcal{F}, \quad \pi_k(f) \in \operatorname*{arg\,min}_{h \in \mathcal{F}^{(k)}} \|f - h\|_\infty \ .$$

We denote:

- the elements of $\mathcal{F}^{(0)}$ by $f_1^{(0)}, \dots, f_{N_0}^{(0)}$ with $N_0 = \mathcal{N}_\infty\big(\mathcal{F}, \gamma\big)$;
- the elements of $\mathcal{G}^{(k)}$ by $g_1^{(k)}, \dots, g_{N_k}^{(k)}$; note that $N_k \leqslant \mathcal{N}_\infty\big(\mathcal{F}, \gamma/2^k\big)\mathcal{N}_\infty\big(\mathcal{F}, \gamma/2^{k-1}\big)$.

With the above definitions, our algorithm can be described as follows:

1. Low-scale aggregation: for every $j \in \{1, \dots, N_0\}$, we use a Multi-variable Exponentiated Gradient forecaster to mimic the best predictor in the neighborhood of $f_j^{(0)}$: we set, at each round $t \geqslant 1$,

$$\widehat{f}_{t,j} \stackrel{\text{def}}{=} f_j^{(0)} + \sum_{k=1}^{K} \sum_{i=1}^{N_k} \widehat{u}_{t,i}^{(j,k)} g_i^{(k)} \ , \tag{5.4}$$

   where $K \stackrel{\text{def}}{=} \big\lceil \log_2(\gamma T/B) \big\rceil$, so that the lowest scale is $\gamma/2^K \approx B/T$. The above weight vectors $\widehat{u}_t^{(j,k)} \in \Delta_{N_k}$ are defined in Equation (5.6) of Algorithm 9. They correspond exactly to the weight vectors output by the Multi-variable Exponentiated Gradient forecaster (Algorithm 8) applied to the loss functions $\ell_t^{(j)} : \Delta_{N_1} \times \ldots \times \Delta_{N_K} \to \mathbb{R}$ defined for all $t \geqslant 1$ (j is fixed) by

$$\ell_t^{(j)}\Big(\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(K)}\Big) = \left(y_t - f_j^{(0)}(x_t) - \sum_{k=1}^{K} \sum_{i=1}^{N_k} u_i^{(k)} g_i^{(k)}(x_t)\right)^2 \ . \tag{5.5}$$

2. High-scale aggregation: we use a standard Exponentially Weighted Average forecaster to aggregate all the $\widehat{f}_{t,j}$, $j = 1, \dots, N_0$, as follows:

$$\widehat{f}_t = \sum_{j=1}^{N_0} \widehat{w}_{t,j} \widehat{f}_{t,j} \ ,$$

   where the weights $\widehat{w}_{t,j}$ are defined in Equation (5.7) of Algorithm 9. At time t, our algorithm predicts y_t with $\widehat{y}_t \stackrel{\text{def}}{=} \widehat{f}_t(x_t)$.

Next we show that the Chaining Exponentially Weighted Average forecaster satisfies a Dudley-type regret bound as in (5.1).

Theorem 5.3. *Let $B > 0$, $T \geqslant 1$, and $\gamma \in \big(\frac{B}{T}, B\big)$.*

- *Assume that $\max_{1 \leqslant t \leqslant T} |y_t| \leqslant B$ and that $\sup_{f \in \mathcal{F}} \|f\|_\infty \leqslant B$.*
- *Assume that $(\mathcal{F}, \|\cdot\|_\infty)$ is totally bounded and define $\mathcal{F}^{(0)} = \big\{f_1^{(0)}, \dots, f_{N_0}^{(0)}\big\}$ and $\mathcal{G}^{(k)} = \big\{g_1^{(k)}, \dots, g_{N_k}^{(k)}\big\}$, $k = 1, \dots, K$, as above.*

Then, the Chaining Exponentially Weighted Average forecaster (Algorithm 9) tuned with the parameters $\eta^{(0)} = 1/(50B^2)$ and $\eta^{(k)} = \sqrt{2\log(N_k)/T}\, 2^k/(30B\gamma)$ for all $k = 1, \dots, K$ satisfies:

$$\mathrm{Reg}_T(\mathcal{F}) \leqslant B^2\big(5 + 50 \log \mathcal{N}_\infty(\mathcal{F}, \gamma)\big) + 120B\sqrt{T} \int_0^{\gamma/2} \sqrt{\log \mathcal{N}_\infty(\mathcal{F}, \varepsilon)}\, \mathrm{d}\varepsilon \ .$$

Algorithm 9: Chaining Exponentially Weighted Average forecaster

input : maximal range $B > 0$, tuning parameters $\eta^{(0)}, \eta^{(1)}, \dots, \eta^{(K)} > 0$,
high-scale functions $f_j^{(0)} : \mathcal{X} \to \mathbb{R}$ for $1 \leqslant j \leqslant N_0$,
low-scale functions $g_i^{(k)} : \mathcal{X} \to \mathbb{R}$ for $k \in \{1, \dots, K\}$ and $i \in \{1, \dots, N_k\}$.

initialization: set $\widehat{\boldsymbol{w}}_1 = \left(\frac{1}{N_0}, \dots, \frac{1}{N_0}\right) \in \Delta_{N_0}$ and
$\widehat{\boldsymbol{u}}_1^{(j,k)} \stackrel{\text{def}}{=} \left(\frac{1}{N_k}, \dots, \frac{1}{N_k}\right) \in \Delta_{N_k}$ for all $j \in \{1, \dots, N_0\}$ and $k \in \{1, \dots, K\}$.

for *each round* $t = 1, 2, \dots$ **do**

- Define the aggregated functions $\widehat{f}_{t,j} : \mathcal{X} \to \mathbb{R}$ for all $j \in \{1, \dots, N_0\}$ by

$$\widehat{f}_{t,j} \stackrel{\text{def}}{=} f_j^{(0)} + \sum_{k=1}^{K} \sum_{i=1}^{N_k} \widehat{u}_{t,i}^{(j,k)} g_i^{(k)} \ .$$

- Observe $x_t \in \mathcal{X}$, predict $\widehat{y}_t = \sum_{j=1}^{N_0} \widehat{w}_{t,j} \widehat{f}_{t,j}(x_t)$, and observe $y_t \in [-B, B]$.

- Low-scale update: compute the new weight vectors $\widehat{\boldsymbol{u}}_{t+1}^{(j,k)} = \left(\widehat{u}_{t+1,i}^{(j,k)}\right)_{1 \leqslant i \leqslant N_k} \in \Delta_{N_k}$ for all $j \in \{1, \dots, N_0\}$ and $k \in \{1, \dots, K\}$ as follows:

$$\widehat{u}_{t+1,i}^{(j,k)} \stackrel{\text{def}}{=} \frac{\exp\left(-\eta^{(k)} \sum_{s=1}^{t} -2\Big(y_s - \widehat{f}_{s,j}(x_s)\Big) g_i^{(k)}(x_s)\right)}{\sum_{i'=1}^{N_k} \exp\left(-\eta^{(k)} \sum_{s=1}^{t} -2\Big(y_s - \widehat{f}_{s,j}(x_s)\Big) g_{i'}^{(k)}(x_s)\right)} \ , \quad i \in \{1, \dots, N_k\} \ . \tag{5.6}$$

- High-scale update: compute the new weight vector $\widehat{\boldsymbol{w}}_{t+1} = \left(\widehat{w}_{t+1,j}\right)_{1 \leqslant j \leqslant N_0} \in \Delta_{N_0}$ as follows:

$$\widehat{w}_{t+1,j} \stackrel{\text{def}}{=} \frac{\exp\left(-\eta^{(0)} \sum_{s=1}^{t} \Big(y_s - \widehat{f}_{s,j}(x_s)\Big)^2\right)}{\sum_{j'=1}^{N_0} \exp\left(-\eta^{(0)} \sum_{s=1}^{t} \Big(y_s - \widehat{f}_{s,j'}(x_s)\Big)^2\right)} \ , \quad j \in \{1, \dots, N_0\} \ . \tag{5.7}$$

end

Remark 5.1. In Theorem 5.3 above, we assumed that the observations y_t and the predictions $f(x_t)$ are all bounded by B, and that B is known in advance by the forecaster. We can actually remove this requirement by using adaptive techniques of Gerchinovitz and Yu [79], namely, adaptive clipping of the intermediate predictions $\widehat{f}_{t,j}(x_t)$ and adaptive Lipschitzification of the square loss functions $\ell_t^{(j)}$. This modification enables us to derive the same regret bound (up to multiplicative constant factors) with $B = \max_t |y_t|$, but without knowing B in advance, and without requiring that $\sup_{f \in \mathcal{F}} \|f\|_\infty$ is also upper bounded by B. Of course these adaptation techniques also make it possible to tune all parameters without knowing T in advance.

Remark 5.2. Even in the case when B is known by the forecaster, the clipping and Lipschitzification techniques of Gerchinovitz and Yu [79] can be useful to get smaller constants in the regret bound. We could indeed replace the constants 50 and 120 with 8 and 48 respectively. (Moreover, the regret bound would also hold true for $\gamma > B$.) We chose however not to use these refinements in order to simplify the analysis.

Remark 5.3. We assumed that the performance of a forecast $\widehat{y}_t$ at round $t \geqslant 1$ is measured through the square loss $\ell_t(\widehat{y}_t) = (\widehat{y}_t - y_t)^2$, which is $1/(50B^2)$-exp-concave on $[-4B, 4B]$. The analysis can easily be extended to all η-exp-concave (and thus convex) loss functions ℓ_t on $[-4B, 4B]$ that also satisfy a self-bounding property of the form $\left|d\ell_t/d\widehat{y}_t\right| \leqslant C\ell_t^r$ (an example is given by $\ell_t(\widehat{y}_t) = \left|\widehat{y}_t - y_t\right|^r$ with $r \geqslant 2$). The regret bound of Theorem 5.3 remains unchanged up to a multiplicative factor depending on B, C, and r. If the loss functions ℓ_t are only convex (e.g., the absolute loss $\ell_t(\widehat{y}_t) = |\widehat{y}_t - y_t|$ or the pinball loss to perform quantile regression), the high-scale aggregation step is more costly: the term of order $\log \mathcal{N}_\infty(\mathcal{F}, \gamma)$ is replaced with a term of order $\sqrt{T \log \mathcal{N}_\infty(\mathcal{F}, \gamma)}$.

Proof (of Theorem 5.3) We split our proof into two parts—one for each aggregation level.

Part 1: low-scale aggregation.

In this part, we fix $j \in \{1, \dots, N_0\}$. As explained right before (5.5), the tuple of weight vectors $\left(\widehat{\boldsymbol{u}}_t^{(j,1)}, \dots, \widehat{\boldsymbol{u}}_t^{(j,K)}\right) \in \Delta_{N_1} \times \dots \Delta_{N_K}$ computed at all rounds $t \geqslant 1$ corresponds exactly to the output of the Multi-variable Exponentiated Gradient forecaster when applied to the loss functions $\ell_t^{(j)}$, $t \geqslant 1$, defined in (5.5). We can therefore apply Theorem 5.2 after checking its assumptions:

- the loss functions $\ell_t^{(j)}$ are indeed differentiable and jointly convex;
- the norms $\left\|\nabla_{\widehat{\boldsymbol{u}}_t^{(j,k)}} \ell_t^{(j)}\right\|_\infty$ of the partial gradients are bounded by $30B\gamma/2^k$ for all $1 \leqslant k \leqslant K$. Indeed, the i-th coordinate of $\nabla_{\widehat{\boldsymbol{u}}_t^{(j,k)}} \ell_t^{(j)}$ is equal to

$$\partial_{\widehat{u}_{t,i}^{(j,k)}} \ell_t^{(j)} = -2\Big(y_t - \widehat{f}_{t,j}(x_t)\Big) g_i^{(k)}(x_t), \tag{5.8}$$

  which can be upper bounded (in absolute value) by $2 \times 5B \times 3\gamma/2^k$. To see why this is true, first note that $\left|g_i^{(k)}(x_t)\right| \leqslant \left\|g_i^{(k)}\right\|_\infty = \left\|\pi_k(f) - \pi_{k-1}(f)\right\|_\infty$ for some $f \in \mathcal{F}$ (by definition of $\mathcal{G}^{(k)}$), so that, by the triangle inequality and by definition of $\pi_k(f)$ and $\mathcal{F}^{(k)}$:

$$\left|g_i^{(k)}(x_t)\right| \leqslant \left\|\pi_k(f) - f\right\|_\infty + \left\|\pi_{k-1}(f) - f\right\|_\infty \leqslant \frac{\gamma}{2^k} + \frac{\gamma}{2^{k-1}} = \frac{3\gamma}{2^k}\ . \tag{5.9}$$

  Second, note that $\left|y_t - \widehat{f}_{t,j}(x_t)\right| \leqslant \left|y_t\right| + \left|\widehat{f}_{t,j}(x_t)\right| \leqslant 5B$. Indeed, we have $|y_t| \leqslant B$ by assumption and, by definition of $\widehat{f}_{t,j}$ in (5.4), we also have

$$\left|\widehat{f}_{t,j}(x_t)\right| \leqslant \left\|f_j^{(0)}\right\|_\infty + \sum_{k=1}^{K} \sum_{i=1}^{N_k} \widehat{u}_{t,i}^{(j,k)} \left|g_i^{(k)}(x_t)\right| \leqslant B + \sum_{k=1}^{K} \frac{3\gamma}{2^k} \leqslant B + 3\gamma \leqslant 4B\ , \tag{5.10}$$

  where we used the inequalities $\left\|f_j^{(0)}\right\|_\infty \leqslant \sup_{f \in \mathcal{F}} \|f\|_\infty \leqslant B$ (by assumption), and where we combined (5.9) with the fact that $\sum_{i=1}^{N_k} \widehat{u}_{t,i}^{(j,k)} = 1$. The last inequality above is obtained from the assumption $\gamma \leqslant B$. Substituting the above various upper bounds in (5.8) entails that

$\big\|\nabla_{\widehat{\boldsymbol{u}}_t^{(j,k)}}\ell_t^{(j)}\big\|_\infty \leqslant 30B\gamma/2^k$ for all $1 \leqslant k \leqslant K$, as claimed earlier.

We are now in a position to apply Theorem 5.2. It yields:

$$\sum_{t=1}^T \Big(y_t - \widehat{f}_{t,j}(x_t)\Big)^2 \leqslant \inf_{g_1,\dots,g_K} \sum_{t=1}^T \Big(y_t - \Big(f_j^{(0)} + g_1 + \ldots + g_K\Big)(x_t)\Big)^2 + \sqrt{2T}\sum_{k=1}^K 30B\gamma/2^k\sqrt{\log N_k}\;, \tag{5.11}$$

where the infimum is over all functions $g_1 \in \mathcal{G}^{(1)}, \dots, g_K \in \mathcal{G}^{(K)}$ (we used the regret bound of Theorem 5.2 with Dirac weight vectors $\boldsymbol{u}^{(k)} = \delta_{i_k}$, $i_k = 1, \dots, N_k$).

Now, using the fact that $N_k \leqslant \mathcal{N}_\infty\big(\mathcal{F}, \gamma/2^k\big)\mathcal{N}_\infty\big(\mathcal{F}, \gamma/2^{k-1}\big) \leqslant \big(\mathcal{N}_\infty\big(\mathcal{F}, \gamma/2^k\big)\big)^2$, we get

$$\begin{aligned}\sum_{k=1}^K \frac{\gamma}{2^k}\sqrt{\log N_k} &\leqslant 2\sqrt{2}\sum_{k=1}^K \Big(\frac{\gamma}{2^k} - \frac{\gamma}{2^{k+1}}\Big)\sqrt{\log \mathcal{N}_\infty\big(\mathcal{F}, \gamma/2^k\big)} \\ &\leqslant 2\sqrt{2}\sum_{k=1}^K \int_{\gamma/2^{k+1}}^{\gamma/2^k} \sqrt{\log \mathcal{N}_\infty\big(\mathcal{F}, \varepsilon\big)}\,\mathrm{d}\varepsilon \leqslant 2\sqrt{2}\int_0^{\gamma/2} \sqrt{\log \mathcal{N}_\infty\big(\mathcal{F}, \varepsilon\big)}\,\mathrm{d}\varepsilon\;,\end{aligned}$$

where the inequality before last follows by monotonicity of $\varepsilon \mapsto \mathcal{N}_\infty(\mathcal{F}, \varepsilon)$ on every interval $\big[\gamma/2^{k+1}, \gamma/2^k\big]$. Finally, substituting the above integral in (5.11) yields

$$\sum_{t=1}^T \Big(y_t - \widehat{f}_{t,j}(x_t)\Big)^2 \leqslant \inf_{g_1,\dots,g_K} \sum_{t=1}^T \Big(y_t - \Big(f_j^{(0)} + g_1 + \ldots + g_K\Big)(x_t)\Big)^2 + 120B\sqrt{T}\int_0^{\gamma/2} \sqrt{\log \mathcal{N}_\infty\big(\mathcal{F}, \varepsilon\big)}\,\mathrm{d}\varepsilon\;. \tag{5.12}$$

Part 2: high-scale aggregation.

The prediction $\widehat{y}_t = \widehat{f}_t(x_t) = \sum_{j=1}^{N_0} \widehat{w}_{t,j}\widehat{f}_{t,j}(x_t)$ at time t is a convex combination of the intermediate predictions $\widehat{f}_{t,j}(x_t)$, where the weights $\widehat{w}_{t,j}$ correspond exactly to those of the standard Exponentially Weighted Average forecaster tuned with $\eta^{(0)} = 1/(50B^2) = 1/\big(2(5B)^2\big)$. Since the intermediate predictions $\widehat{f}_{t,j}(x_t)$ lie in $[-4B, 4B]$ (by (5.10) above), and since the square loss $z \mapsto (y_t - z)^2$ is $\eta^{(0)}$-exp-concave on $[-4B, 4B]$ for any $y_t \in [-B, B]$, we get from Proposition 3.1 and Page 46 of Cesa-Bianchi and Lugosi [43] that

$$\begin{aligned}\sum_{t=1}^T (y_t - \widehat{y}_t)^2 &\leqslant \min_{1\leqslant j\leqslant N_0} \sum_{t=1}^T \Big(y_t - \widehat{f}_{t,j}(x_t)\Big)^2 + \frac{\log N_0}{\eta^{(0)}} \\ &\leqslant \inf_{f_0,g_1,\dots,g_K} \sum_{t=1}^T \big(y_t - (f_0 + g_1 + \ldots + g_K)(x_t)\big)^2 \\ &\quad + 120B\sqrt{T}\int_0^{\gamma/2} \sqrt{\log \mathcal{N}_\infty\big(\mathcal{F}, \varepsilon\big)}\,\mathrm{d}\varepsilon + 50B^2\log \mathcal{N}_\infty\big(\mathcal{F}, \gamma\big)\;,\end{aligned} \tag{5.13}$$

where the infimum is over all functions $f_0 \in \mathcal{F}^{(0)}, g_1 \in \mathcal{G}^{(1)}, \dots, g_K \in \mathcal{G}^{(K)}$. The last inequality above was a consequence of (5.12). Next we apply the chaining idea: by definition of the function

sets $\mathcal{F}^{(0)} \supseteq \{\pi_0(f) : f \in \mathcal{F}\}$ and $\mathcal{G}^{(k)} = \{\pi_k(f) - \pi_{k-1}(f) : f \in \mathcal{F}\}$, we have

$$\begin{aligned}
&\inf_{f_0,g_1,\dots,g_K} \sum_{t=1}^{T} \big(y_t - \big(f_0 + g_1 + \ldots + g_K\big)(x_t)\big)^2 \\
&\quad\leqslant \inf_{f\in\mathcal{F}} \sum_{t=1}^{T} \Big(y_t - \big(\pi_0(f) + \big[\pi_1(f) - \pi_0(f)\big] + \ldots + \big[\pi_K(f) - \pi_{K-1}(f)\big]\big)(x_t)\Big)^2 \\
&\quad= \inf_{f\in\mathcal{F}} \sum_{t=1}^{T} \big(y_t - \pi_K(f)(x_t)\big)^2 \\
&\quad\leqslant \inf_{f\in\mathcal{F}} \sum_{t=1}^{T} \Big[\big(y_t - f(x_t)\big)^2 + 2\cdot 2B\,\|\pi_K(f) - f\|_\infty + \|\pi_K(f) - f\|_\infty^2\Big] \qquad (5.14) \\
&\quad\leqslant \inf_{f\in\mathcal{F}} \sum_{t=1}^{T} \big(y_t - f(x_t)\big)^2 + 4B^2 + \frac{B^2}{T}\,, \qquad (5.15)
\end{aligned}$$

where (5.14) is obtained by expanding the square $\big(y_t - \pi_K(f)(x_t)\big)^2 = \big(y_t - f(x_t) + f(x_t) - \pi_K(f)(x_t)\big)^2$, and where (5.15) follows from the fact that $\|\pi_K(f) - f\|_\infty \leqslant \gamma/2^K \leqslant B/T$ by definition of $\pi_K(f)$ and $K = \lceil \log_2(\gamma T/B)\rceil$. Combining (5.13) and (5.15) concludes the proof. ∎

5.3. An efficient chaining algorithm for Hölder classes

The Chaining Exponentially Weighted Average forecaster of the previous section is quite natural since it explicitly exploits the ε-nets that appear in the Dudley-type regret bound (5.1). However its time and space computational complexities are prohibitively large (exponential in T) since it is necessary to update exponentially many weights at every round t. It actually turns out that, fortunately, most standard function classes have a sufficiently nice structure. This enables us to adapt the previous chaining technique on (quasi-optimal) ε-nets that are much easier to exploit from an algorithmic viewpoint. We describe below the particular case of Lipschitz classes; the more general case of Hölder classes is postponed to the appendix.

In all the sequel, $\mathcal{F}$ denotes the set of functions from $[0,1]$ to $[-B,B]$ that are 1-Lipschitz. Recall from the introduction that $\log \mathcal{N}_\infty(\mathcal{F},\varepsilon) = \Theta(\varepsilon^{-1})$, so that, by Theorem 5.3 and (5.2), the Chaining Exponentially Weighted Average forecaster guarantees a regret of $\mathcal{O}\big(T^{1/3}\big)$. We explain below how to modify this algorithm with ε-nets of $\big(\mathcal{F}, \|\cdot\|_\infty\big)$ that are easier to manage from a computational viewpoint. This leads to a quasi-optimal regret of $\mathcal{O}\big(T^{1/3}\log T\big)$; see Theorem 5.4.

5.3.1. Constructing computationally-manageable ε-nets via a dyadic discretization

Let $\gamma \in \big(\frac{B}{T}, B\big)$ be a fixed real number that will play the same role as in Theorem 5.3. Using the fact that all functions in $\mathcal{F}$ are 1-Lipschitz, we can approximate $\mathcal{F}$ with piecewise-constant functions as follows. We partition the x-axis $[0,1]$ into $1/\gamma$ subintervals $I_a \stackrel{\text{def}}{=} \big[(a-1)\gamma, a\gamma\big)$, $a = 1,\dots,1/\gamma$ (the last interval is closed at $x=1$). We also use a discretization of length γ on the y-axis $[-B,B]$, by considering values of the form $c^{(0)} = -B + j\gamma$, $j = 0,\dots,2B/\gamma$. (For the sake of simplicity, we assume that both $1/\gamma$ and $2B/\gamma$ are integers.) We then define the set $\mathcal{F}^{(0)}$ of piecewise-constant

functions $f^{(0)} : [0,1] \to [-B, B]$ of the form

$$f^{(0)}(x) = \sum_{a=1}^{1/\gamma} c_a^{(0)} \mathbb{1}_{x \in I_a} \ , \qquad c_1^{(0)}, \dots, c_{1/\gamma}^{(0)} \in \mathcal{C}^{(0)} \stackrel{\text{def}}{=} \left\{ -B + j\gamma : j = 0, \dots, \frac{2B}{\gamma} \right\} . \tag{5.16}$$

Using the fact that all functions in $\mathcal{F}$ are 1-Lipschitz, it is quite straightforward to see that $\mathcal{F}^{(0)}$ is a γ-net[‡] of $\big(\mathcal{F}, \|\cdot\|_\infty\big)$. (To see why this is true, we can choose $c_a^{(0)} \in \arg\min_{c \in C^{(0)}} \big|f(x_a) - c\big|$, where x_a is the center of the subinterval I_a. See Lemma 5.9 in the appendix for further details.)

Refinement via a dyadic discretization Next we construct $\gamma/2^m$-nets that are refinements of the γ-net $\mathcal{F}^{(0)}$. We need to define a dyadic discretization for each subinterval I_a as follows: for any level $m \geqslant 1$, we partition I_a into 2^m subintervals $I_a^{(m,n)}$, $n = 1, \dots, 2^m$, of equal size $\gamma/2^m$. Note that the subintervals $I_a^{(m,n)}$, $a = 1, \dots, 1/\gamma$ and $n = 1, \dots, 2^m$, form a partition of $[0,1]$. We call it *the level-m partition*. We enrich the set $\mathcal{F}^{(0)}$ by looking at all the functions of the form $f^{(0)} + \sum_{m=1}^M g^{(m)}$, where $f^{(0)} \in \mathcal{F}^{(0)}$ and where every function $g^{(m)}$ is piecewise-constant on the level-m partition, with values $c_a^{(m,n)} \in \big[-\gamma/2^{m-1}, \gamma/2^{m-1}\big]$ that are small when m is large. In other words, we define *the level-M approximation set* $\mathcal{F}^{(M)}$ as the set of all functions $f_{\boldsymbol{c}} : [0,1] \to \mathbb{R}$ of the form

$$f_{\boldsymbol{c}}(x) = \underbrace{\sum_{a=1}^{1/\gamma} c_a^{(0)} \mathbb{1}_{x \in I_a}}_{f^{(0)}(x)} + \sum_{m=1}^{M} \underbrace{\sum_{a=1}^{1/\gamma} \sum_{n=1}^{2^m} c_a^{(m,n)} \mathbb{1}_{x \in I_a^{(m,n)}}}_{g^{(m)}(x)} \ , \tag{5.17}$$

where $c_a^{(0)} \in \mathcal{C}^{(0)}$ and $c_a^{(m,n)} \in \big[-\gamma/2^{m-1}, \gamma/2^{m-1}\big]$. An example of function $f_{\boldsymbol{c}} = f^{(0)} + \sum_{m=1}^M g^{(m)}$ is plotted on Figure 5.1 in the case when $M = 2$ (the plot is restricted to the interval I_a).

Since all functions in $\mathcal{F}$ are 1-Lipschitz, the set $\mathcal{F}^{(M)}$ of all functions $f_{\boldsymbol{c}}$ is a $\gamma/2^{M+1}$-net of $(\mathcal{F}, \|\cdot\|_\infty)$; see Lemma 5.9 in the appendix for a proof. Note that $\mathcal{F}^{(M)}$ is infinite (the $c_a^{(m,n)}$ are continuously valued); fortunately this is not a problem since the $c_a^{(m,n)}$ can be rewritten as convex combinations $c_a^{(m,n)} = u_1^{(m,n)}(-\gamma/2^{m-1}) + u_2^{(m,n)}(\gamma/2^{m-1})$ of only two values; cf. (5.18) below.

5.3.2. A chaining algorithm using this dyadic discretization

Next we design an algorithm which, as in Section 5.2.2, is able to be competitive against any function $f_{\boldsymbol{c}} = f^{(0)} + \sum_{m=1}^M g^{(m)}$. However, instead of maintaining exponentially many weights as in Algorithm 9, we use the dyadic discretization in a crucial way. More precisely:

We run $1/\gamma$ instances of the same algorithm $\mathcal{A}$ in parallel; the a-th instance $\mathcal{A}_a$, $a = 1, \dots, 1/\gamma$, corresponds to the subinterval I_a and it is updated only at rounds t such that $x_t \in I_a$.

Next we focus on subalgorithm $\mathcal{A}_a$. As in Algorithm 9, we use a combination of the EWA and the Multi-variable EG forecasters to perform high-scale and low-scale aggregation simultaneously:

Low-scale aggregation: we run $2B/\gamma + 1$ instances $\mathcal{B}_{a,j}$, $j = 0, \dots, 2B/\gamma$, of the Adaptive Multi-variable Exponentiated Gradient algorithm (Algorithm 10 in the appendix) simultaneously. Each instance $\mathcal{B}_{a,j}$ corresponds to a particular constant $c^{(0)} = -B + j\gamma \in \mathcal{C}^{(0)}$ and is run (similarly

[‡]This γ-net is not proper since $\mathcal{F}^{(0)} \not\subseteq \mathcal{F}$.

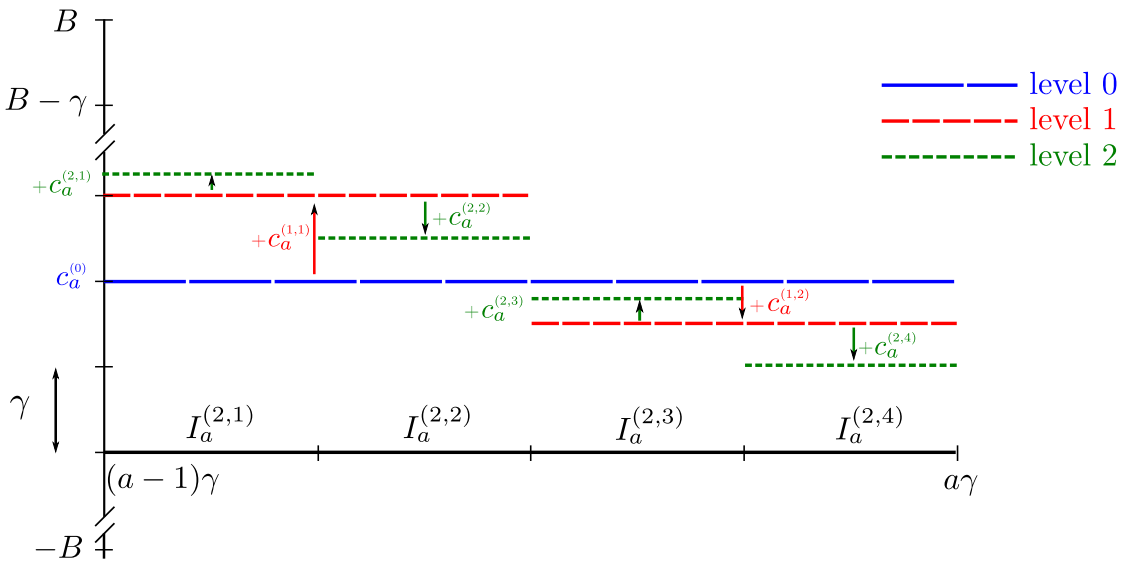


Figure 5.1.: An example of function $f^{(0)} + \sum_{m=1}^{M} g^{(m)}$ for $M = 2$, plotted on the subinterval I_a. This function corresponds to the dotted line (level 2).

to (5.5)) with the loss function ℓ_t defined for all weight vectors $\boldsymbol{u}^{(m,n)} = \left(u_1^{(m,n)}, u_2^{(m,n)}\right) \in \Delta_2$ by

$$\begin{aligned}&\ell_t\left(\boldsymbol{u}^{(m,n)},\, m = 1, \dots, M,\, n = 1, \dots, 2^m\right)\\ &\quad = \left(y_t - (-B + j\gamma) - \sum_{m=1}^{M} \sum_{n=1}^{2^m} \left(u_1^{(m,n)} \frac{-\gamma}{2^{m-1}} + u_2^{(m,n)} \frac{\gamma}{2^{m-1}}\right) \mathbb{1}_{x_t \in I_a^{(m,n)}}\right)^2 . \end{aligned} \tag{5.18}$$

The above convex combinations $u_1^{(m,n)}(-\gamma/2^{m-1}) + u_2^{(m,n)}(\gamma/2^{m-1})$ ensure that subalgorithm $\mathcal{B}_{a,j}$ is competitive against the best constants $c_a^{(m,n)} \in \left[-\gamma/2^{m-1}, \gamma/2^{m-1}\right]$ for all m and n.

The weight vectors output by subalgorithm $\mathcal{B}_{a,j}$ (when $x_t \in I_a$) are denoted by $\widehat{\boldsymbol{u}}_{t,a,j}^{(m,n)}$, and we set

$$\widehat{f}_{t,a,j}(x) \stackrel{\text{def}}{=} -B + j\gamma + \sum_{m=1}^{M} \sum_{n=1}^{2^m} \left(\widehat{u}_{t,a,j,1}^{(m,n)} \frac{-\gamma}{2^{m-1}} + \widehat{u}_{t,a,j,2}^{(m,n)} \frac{\gamma}{2^{m-1}}\right) \mathbb{1}_{x \in I_a^{(m,n)}}$$

for all $j = 0, \dots, 2B/\gamma$.

High-scale aggregation: we aggregate the $2B/\gamma + 1$ forecasters above with a standard Exponentially Weighted Average forecaster (tuned, e.g., with the parameter $\eta = 1/(2(4B)^2) = 1/(32B^2)$):

$$\widehat{f}_{t,a} = \sum_{j=0}^{2B/\gamma} \widehat{w}_{t,a,j} \widehat{f}_{t,a,j} \ . \tag{5.19}$$

Putting all things together: at every time $t \geqslant 1$, we make the prediction

$$\widehat{f}_t(x_t) \stackrel{\text{def}}{=} \sum_{a=1}^{1/\gamma} \widehat{f}_{t,a}(x_t) \mathbb{1}_{x_t \in I_a} \, .$$

We call this algorithm the *Dyadic Chaining Algorithm.*

Theorem 5.4. *Let $B > 0$, $T \geqslant 2$, and $\mathcal{F}$ be the set of all 1-Lipschitz functions from $[0,1]$ to $[-B,B]$. Assume that $\max_{1\leqslant t\leqslant T}|y_t| \leqslant B$. Then, the Dyadic Chaining Algorithm defined above and tuned with the parameters $\gamma = BT^{-1/3}$ and $M = \lceil \log_2(\gamma T/B) \rceil$ satisfies, for some absolute constant $c > 0$,*

$$\mathrm{Reg}_T(\mathcal{F}) \leqslant c \max\{B, B^2\} T^{1/3} \log T \ .$$

The proof is postponed to the appendix. Note that the Dyadic Chaining Algorithm is computationally tractable: at every round t, the point x_t only falls into one subinterval $I_a^{(m,n)}$ for each level $m = 1, \dots, M$, so that we only need to update $\mathcal{O}\big(2B/\gamma \times M\big) = \mathcal{O}\big(T^{1/3}\log T\big)$ weights at every round. For the same reason, the per-round space complexity is $\mathcal{O}(T \times 2B/\gamma \times M) = \mathcal{O}\big(T^{4/3}\log T\big)$.

Appendices for Chapter 5

5.A. Adaptive Multi-variable Exponentiated Gradient

In this subsection, we provide an adaptive version of Algorithm 8 when the time horizon T is not known in advance. We adopt the notations of Section 5.2.1. Basically, the fixed tuning parameters $\eta^{(1)}, \dots, \eta^{(k)}$ are replaced with time-varying learning rates $\eta_t^{(1)}, \dots, \eta_t^{(k)}$.

The Adaptive Multi-variable Exponentiated Gradient algorithm satisfies the regret bound of Theorem 5.5 below.

Theorem 5.5. *Assume that the loss functions $\ell_t : \Delta_{N_1} \times \ldots \times \Delta_{N_K} \to \mathbb{R}$, $t \geqslant 1$, are differentiable and jointly convex. Assume also the following upper bound on their partial gradients: for all $k \in \{1, \dots, K\}$,*

$$\max_{1\leqslant t\leqslant T} \big\|\nabla_{\boldsymbol{u}^{(k)}} \ell_t\big\|_\infty \leqslant G^{(k)} \ . \tag{5.20}$$

Then, the Multi-variable Exponentiated Gradient algorithm (Algorithm 10) has a regret bounded as follows:

$$\sum_{t=1}^T \ell_t\Big(\widehat{\boldsymbol{u}}_t^{(1)}, \dots, \widehat{\boldsymbol{u}}_t^{(K)}\Big) - \min_{\boldsymbol{u}^{(1)},\dots,\boldsymbol{u}^{(K)}} \sum_{t=1}^T \ell_t\Big(\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(K)}\Big) \leqslant 2\sum_{k=1}^K G^{(k)} \sqrt{T^{(k)} \log N_k} \ ,$$

where $T^{(k)} = \sum_{t=1}^T \mathbb{1}_{\|\nabla_{\boldsymbol{u}^{(k)}}\ell_t\|_\infty > 0}$ and where the minimum is taken over all $\big(\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(K)}\big) \in \Delta_{N_1} \times \ldots \times \Delta_{N_K}$.

Algorithm 10: Adaptive Multi-variable Exponentiated Gradient

input : optimization domain $\Delta_{N_1} \times \ldots \times \Delta_{N_K}$ (where $N_1, \ldots, N_K$ are positive integers).

initialization: set $\widehat{\boldsymbol{u}}_1^{(k)} \stackrel{\text{def}}{=} \left(\frac{1}{N_k}, \ldots, \frac{1}{N_k}\right) \in \Delta_{N_k}$ for all $k = 1, \ldots, K$.

for *each round* $t = 1, 2, \ldots$ **do**

- Output $\left(\widehat{\boldsymbol{u}}_t^{(1)}, \ldots, \widehat{\boldsymbol{u}}_t^{(K)}\right) \in \Delta_{N_1} \times \ldots \times \Delta_{N_K}$ and observe the differentiable and jointly convex loss function $\ell_t : \Delta_{N_1} \times \ldots \times \Delta_{N_K} \to \mathbb{R}$.
- Update the tuning parameters, $\eta_t^{(k)}$ for all $k = 1, \ldots, K$ as follows:

$$\eta_{t+1}^{(k)} = \frac{1}{G^{(k)}} \sqrt{\frac{\log N^{(k)}}{1 + \sum_{s=1}^{t} \mathbb{1}_{\left\|\nabla_{\boldsymbol{u}^{(k)}} \ell_s\right\|_\infty > 0}}}$$

- Compute the new weight vectors $\left(\widehat{\boldsymbol{u}}_{t+1}^{(1)}, \ldots, \widehat{\boldsymbol{u}}_{t+1}^{(K)}\right) \in \Delta_{N_1} \times \ldots \times \Delta_{N_K}$ as follows:

$$\widehat{\boldsymbol{u}}_{t+1,i}^{(k)} \stackrel{\text{def}}{=} \frac{\exp\left(-\eta_{t+1}^{(k)} \sum_{s=1}^{t} \partial_{\widehat{u}_{s,i}^{(k)}} \ell_s\left(\widehat{\boldsymbol{u}}_s^{(1)}, \ldots, \widehat{\boldsymbol{u}}_s^{(K)}\right)\right)}{Z_{t+1}^{(k)}}, \quad i \in \{1, \ldots, N_k\},$$

where $\partial_{\widehat{u}_{s,i}^{(k)}} \ell_s$ denotes the partial derivative of ℓ_s with respect to i-th component of the vector variable $\widehat{\boldsymbol{u}}_s^{(k)}$, and where the normalization factor $Z_{t+1}^{(k)}$ is defined by

$$Z_{t+1}^{(k)} \stackrel{\text{def}}{=} \sum_{i=1}^{N_k} \exp\left(-\eta_{t+1}^{(k)} \sum_{s=1}^{t} \partial_{\widehat{u}_{s,i}^{(k)}} \ell_s\left(\widehat{\boldsymbol{u}}_s^{(1)}, \ldots, \widehat{\boldsymbol{u}}_s^{(K)}\right)\right) .$$

end

Proof (of Theorem 5.5) The proof starts as the one of Theorem 5.2. From (5.31), we can see that

$$\begin{aligned}
&\sum_{t=1}^{T} \ell_t\left(\widehat{\boldsymbol{u}}_t^{(1)}, \ldots, \widehat{\boldsymbol{u}}_t^{(K)}\right) - \min_{\boldsymbol{u}^{(1)}, \ldots, \boldsymbol{u}^{(K)}} \sum_{t=1}^{T} \ell_t\left(\boldsymbol{u}^{(1)}, \ldots, \boldsymbol{u}^{(K)}\right) \\
&\qquad = \sum_{k=1}^{K} \left(\sum_{t=1}^{T} \boldsymbol{g}_t^{(k)} \cdot \widehat{\boldsymbol{u}}_t^{(k)} - \min_{1 \leqslant i \leqslant N_k} \sum_{t=1}^{T} g_{t,i}^{(k)} \right) \\
&\qquad = \sum_{k=1}^{K} \left(\sum_{t \in \mathcal{T}^{(k)}} \boldsymbol{g}_t^{(k)} \cdot \widehat{\boldsymbol{u}}_t^{(k)} - \min_{1 \leqslant i \leqslant N_k} \sum_{t \in \mathcal{T}^{(k)}} g_{t,i}^{(k)} \right) , \qquad (5.21)
\end{aligned}$$

where $\boldsymbol{g}_t^{(k)} \stackrel{\text{def}}{=} \nabla_{\widehat{\boldsymbol{u}}_t^{(k)}} \ell_t(\widehat{\boldsymbol{u}}_t^{(1)}, \ldots, \widehat{\boldsymbol{u}}_t^{(K)})$ and where $\mathcal{T}^{(k)} = \left\{t = 1, \ldots, T, \quad \|\cdot\| \nabla_{[} \boldsymbol{u}^{(k)} \ell_t]_\infty > 0\right\}$.

Note that the right-hand side of (5.31) is the sum of K regrets. Let $k \in \{1, \ldots, K\}$. By definition of the Adaptive Multi-variable Exponentiated Gradient algorithm, the sequence of weight vectors $\left(\widehat{\boldsymbol{u}}_t^{(k)}\right)_{t \geqslant 1}$ corresponds exactly to the weight vectors output by the Exponentially Weighted Average

forecaster with time-varying parameter (see Page 50 of Gerchinovitz [78]) applied to N_k experts associated with the loss vectors $\boldsymbol{g}_t^{(k)} \in \mathbb{R}^{N_k}$, $t \in \mathcal{T}^{(k)}$. We can therefore use the well-known corresponding regret bound available, e.g., in Proposition 2.1 of Gerchinovitz [78]. Noting that the loss vectors $\boldsymbol{g}_t^{(k)}$ lie in $\left[-G^{(k)}, G^{(k)}\right]^{N_k}$ by Assumption (5.20), and setting $T^{(k)} = \operatorname{card} \mathcal{T}^{(k)}$, we thus get that

$$\sum_{t \in \mathcal{T}^{(k)}} \boldsymbol{g}_t^{(k)} \cdot \widehat{\boldsymbol{u}}_t^{(k)} - \min_{1 \leqslant i \leqslant N_k} \sum_{t \in \mathcal{T}^{(k)}} g_{t,i}^{(k)} \leqslant 2G^{(k)} \sqrt{T^{(k)} \log N_k}\,.$$

Note that the additional term $G^{(k)}\sqrt{\log N_k}$ in the upper-bound of Gerchinovitz [78] is actually not needed, since we can assume that $\eta_{T+1}^{(k)} = \eta_T^{(k)}$ because $\eta_{T+1}^{(k)}$ is not used by the algorithm at rounds $t \leqslant T$. Substituting the last upper bound in the right-hand side of (5.21) concludes the proof. ∎

5.B. An efficient chaining algorithm for Hölder classes

In this appendix, we extend the analysis of Section 5.3 to Hölder function classes. In the sequel $\mathcal{F}$ denotes the set of functions on $[0,1]$ whose q first derivatives ($q \in \mathbb{N}$) exist and are all bounded in supremum norm by a constant B, and whose qth derivative is Hölder continuous of order $\alpha \in (0,1]$ with coefficient $\lambda > 0$. In other words, any function $f \in \mathcal{F}$ satisfies

$$\forall x, y \in [0,1], \quad \left|f^{(q)}(x) - f^{(q)}(y)\right| \leqslant \lambda |x-y|^{\alpha}\,, \tag{5.22}$$

and $\|f^{(k)}\|_\infty \leqslant B$ for all $k \in \{0, \dots, q\}$. We denote by $\beta = q + \alpha$ the coefficient of regularity of $\mathcal{F}$. Recall from the introduction that $\log \mathcal{N}_\infty(\mathcal{F}, \varepsilon) = \mathcal{O}(\varepsilon^{-1/\beta})$, so that, by Theorem 5.3 and (5.2), if $\beta > 1/2$, the Chaining Exponentially Weighted Average forecaster guarantees a regret of $\mathcal{O}\big(T^{1/(2\beta+1)}\big)$, which is optimal. We explain below how to modify this algorithm with non-proper ε-nets of $(\mathcal{F}, \|\cdot\|_\infty)$ that are easier to manage from a computational viewpoint. This leads to a quasi-optimal regret of $\mathcal{O}\big(T^{1/(2\beta+1)}(\log T)^{3/2}\big)$.

The analysis follows the one of Section 5.3 which dealt with the special case of 1-Lipschitz functions. The main difference consists in replacing piecewise-constant approximations with piecewise-polynomial approximations.

5.B.1. Constructing computationally-manageable ε-nets via exponentially nested discretization

Let $\gamma \in \left(\frac{B}{T}, B\right)$ be a fixed real number that will play the same role as in Theorem 5.3. Using the fact that all functions in $\mathcal{F}$ are Hölder, we can approximate $\mathcal{F}$ with piecewise-polynomial functions as follows.

Let $\delta_x > 0$ and $\delta_y > 0$ be two discretization widths that will be fixed later by the analysis. We partition the x-axis $[0,1]$ into $1/\delta_x$ subintervals $I_a \stackrel{\text{def}}{=} \big[(a-1)\delta_x, a\delta_x\big)$, $a = 1, \dots, 1/\delta_x$ (the last interval is closed at $x = 1$). We also use a discretization of length δ_y on the y-axis $[-B, B]$, by considering the set

$$\mathcal{Y}^{(0)} \stackrel{\text{def}}{=} \Big\{ -B + j\delta_y \;:\; j = 0, \dots, 2B/\delta_y \Big\}\,.$$

For the sake of simplicity, we assume that both $1/\delta_x$ and $2B/\delta_y$ are integers. Otherwise, it suffices to consider $\lceil 1/\delta_x \rceil$ and $\lceil 2B/\delta_y \rceil$, which only impacts the constants of the final Theorem 5.7. We

then define the sets of clipped polynomial functions for every $a \in \{1, \dots, 1/\delta_x\}$

$$\mathcal{P}_a^{(0)} \stackrel{\text{def}}{=} \left\{ x \mapsto \left[a_0 + \frac{a_1}{1!}(x - x_a)^2 + \cdots + \frac{a_q}{q!}(x - x_a)^q \right]_B \; : \quad a_0, \dots, a_q \in \mathcal{Y}^{(0)} \right\} .$$

Here, $[\cdot]_B$ is the clipping operator defined by $[x]_B \stackrel{\text{def}}{=} \min\big\{B, \max\{-B, x\}\big\}$ and x_a is the center of I_a. Now, we define the set $\mathcal{F}^{(0)}$ of piecewise-clipped polynomial functions $f^{(0)} : [0,1] \to [-B, B]$ of the form

$$f^{(0)}(x) = \sum_{a=1}^{1/\delta_x} P_a^{(0)}(x) \mathbb{1}_{x \in I_a} , \qquad \forall a \in \{1, \dots, 1/\delta_x\} \quad P_a^{(0)} \in \mathcal{P}_a^{(0)} . \tag{5.23}$$

Remark that the above definition is similar to (5.16), where the constants $c_a^{(0)}$ have been substituted with clipped polynomials. Using the fact that all functions in $\mathcal{F}$ are Hölder, we can see (cf. Lemma 5.6) that for $\delta_x = 2(q!\gamma/(2\lambda))^{1/\beta}$ and $\delta_y = \gamma/e$, the set $\mathcal{F}^{(0)}$ is a γ-net[§] of $\big(\mathcal{F}, \|\cdot\|_\infty\big)$.

Refinement via an exponentially nested discretization Next we construct $\gamma/2^m$-nets that are refinements of the γ-net $\mathcal{F}^{(0)}$. We need to define an exponentially nested discretization for each subinterval I_a as follows: for any level $m \geqslant 1$, we partition I_a into 4^m subintervals $I_a^{(m,n)}$, $n = 1, \dots, 4^m$, of equal size $\delta_x/4^m$. Note that the subintervals $I_a^{(m,n)}$, $a = 1, \dots, 1/\delta_x$ and $n = 1, \dots, 4^m$, form a partition of $[0,1]$. We call it *the level-m partition*.

Now, we design the sets of clipped polynomial functions $\mathcal{Q}_a^{(m,n)}$ that will refine the approximation of $\mathcal{F}$ on each interval $I_a^{(m,n)}$. To do so, for every $m \geqslant 1$ we set successive dyadic refining discretizations of the coefficients space $[-B, B]$:

$$\mathcal{Y}^{(m)} \stackrel{\text{def}}{=} \Big\{ -B + j\delta_y/2^m \; : \quad j = 0, \dots, 2^{m+1}B/\delta_y \Big\} , \tag{5.24}$$

and we define the corresponding sets of clipped polynomial functions for all $a \in \{1, \dots, 1/\delta_x\}$, all $m \in \{1, \dots, M\}$, and $n \in \{1, \dots, 4^m\}$

$$\mathcal{P}_a^{(m,n)} \stackrel{\text{def}}{=} \left\{ x \mapsto \left[a_0 + \frac{a_1}{1!}\left(x - x_a^{(m,n)}\right)^2 + \cdots + \frac{a_q}{q!}\left(x - x_a^{(m,n)}\right)^q \right]_B \; : \quad a_0, \dots, a_q \in \mathcal{Y}^{(m)} \right\} , \tag{5.25}$$

where $x_a^{(m,n)}$ is the center of the interval $I_a^{(m,n)}$. Then, we define the sets of differences between clipped polynomial functions of two consecutive levels

$$\mathcal{Q}_a^{(m,n)} = \left\{ \left[P^{(m)} - P^{(m-1)} \right]_{3\gamma/2^m} : P^{(m)} \in \mathcal{P}_a^{(m,n)} \text{ and } P^{(m-1)} \in \mathcal{P}_a^{(m-1,n_{m-1})} \right\}$$

where n_{m-1} denotes the unique integer n' such that $I_a^{(m,n)} \subset I_a^{(m-1,n')}$. (For $m = 1$, $\mathcal{P}_a^{(m-1,n_{m-1})}$ is replaced with $\mathcal{P}_a^{(0)}$ in the definition of $\mathcal{Q}_a^{(m,n)}$). The functions in $\mathcal{Q}_a^{(m,n)}$ will play the same role as the constants $c_a^{(m,n)}$ for the Lipschitz case to refine the approximation from the level-$(m-1)$ partition to the level-m partition. Note that each $Q_a^{(m,n)} \in \mathcal{Q}_a^{(m,n)}$ takes values in $[-3\gamma/2^m, 3\gamma/2^m]$.

Then, we enrich the set $\mathcal{F}^{(0)}$ by looking at all the functions of the form $f^{(0)} + \sum_{m=1}^{M} g^{(m)}$, where $f^{(0)} \in \mathcal{F}^{(0)}$ and where every function $g^{(m)}$ is the difference of a piecewise-clipped polynomial on

[§]This γ-net is not proper since $\mathcal{F}^{(0)} \not\subseteq \mathcal{F}$.

the level-m partition and a piecewise-clipped polynomial on the previous level $m-1$, with values $Q_a^{(m,n)} \in \mathcal{Q}_a^{(m,n)}$.

In other words, we define *the level-M approximation set* $\mathcal{F}^{(M)}$ as the set of all functions $f_{\boldsymbol{c}} : [0,1] \to \mathbb{R}$ of the form

$$f_{\boldsymbol{c}}(x) = \underbrace{\sum_{a=1}^{1/\delta_x} P_a^{(0)}(x) \mathbb{1}_{x \in I_a}}_{f^{(0)}(x)} + \underbrace{\sum_{m=1}^{M} \sum_{a=1}^{1/\delta_x} \sum_{n=1}^{4^m} Q_a^{(m,n)}(x) \mathbb{1}_{x \in I_a^{(m,n)}}}_{g^{(m)}(x)} , \tag{5.26}$$

where $P_a^{(0)} \in \mathcal{P}_a^{(0)}$ and $Q_a^{(m,n)} \in \mathcal{Q}_a^{(m,n)}$. Once again, see (5.26) as an extension of (5.17), where the constants $c_a^{(m,n)}$ have been replaced with $Q_a^{(m,n)}$.

Using again the fact that all functions in $\mathcal{F}$ are Hölder, we can show that the set $\mathcal{F}^{(M)}$ of all functions $f_{\boldsymbol{c}}$ is a $\gamma/2^M$-net of $(\mathcal{F}, \|\cdot\|_\infty)$; see Lemma 5.6 below (whose proof is postponed to Appendix 5.C.3) for further details.

Lemma 5.6. *Let $\mathcal{F}$ be the set of Hölder functions defined in (5.22). Assume that $\beta \stackrel{\text{def}}{=} q + \alpha \geqslant 1/2$. Let $\delta_x = 2(q!\gamma/(2\lambda))^{1/\beta}$ and $\delta_y = \gamma/e$. Then:*

- *the set $\mathcal{F}^{(0)}$ defined in (5.23) is a γ-net of $\big(\mathcal{F}, \|\cdot\|_\infty\big)$;*
- *for all $M \geqslant 1$, the set $\mathcal{F}^{(M)}$ defined in (5.26) is a $\gamma/2^M$-net of $\big(\mathcal{F}, \|\cdot\|_\infty\big)$.*

5.B.2. A chaining algorithm using this exponentially nested refining discretization

Next we design an algorithm which, as in Section 5.3, is able to be competitive against any function $f_{\boldsymbol{c}} = f^{(0)} + \sum_{m=1}^{M} g^{(m)}$ and is computationally tractable. More precisely:

We run $1/\delta_x$ instances of the same algorithm $\mathcal{A}$ in parallel; the a-th instance corresponds to the subinterval I_a and it is updated only at rounds t such that $x_t \in I_a$.

Next we focus on the a-th instance of the algorithm $\mathcal{A}$, whose local time is only incremented when a new x_t falls into I_a. As in Algorithm 9, we use a combination of the EWA and the Multi-variable EG forecasters to perform high-scale and low-scale aggregation simultaneously:

Low-scale aggregation: we run $\operatorname{card} \mathcal{P}_a^{(0)} \leqslant (2B/\delta_y + 1)^{(q+1)}$ instances $\mathcal{B}_{a,j}$, $j = 1, \dots, \operatorname{card} \mathcal{P}_a^{(0)}$ of the Adaptive Multi-variable Exponentiated Gradient algorithm (Algorithm 10 in the appendix) simultaneously. Each instance $\mathcal{B}_{a,j}$ corresponds to a particular polynomial $P_{a,j}^{(0)} \in \mathcal{P}_a^{(0)}$ and is run (similarly to (5.5)) with the loss function ℓ_t defined for all weight vectors $\boldsymbol{u}^{(m,n)} \in \Delta_{\operatorname{card} \mathcal{Q}_a^{(m,n)}}$ by

$$\begin{aligned} &\ell_t\left(\boldsymbol{u}^{(m,n)},\, m = 1, \dots, M,\, n = 1, \dots, 4^m\right) \\ &\quad = \left(y_t - P_{a,j}^{(0)}(x_t) - \sum_{m=1}^{M} \sum_{n=1}^{4^m} \sum_{k=1}^{\operatorname{card} \mathcal{Q}_a^{(m,n)}} u_k^{(m,n)} Q_{a,k}^{(m,n)}(x_t) \mathbb{1}_{x_t \in I_a^{(m,n)}} \right)^2 . \end{aligned} \tag{5.27}$$

Here, $Q_{a,1}^{(m,n)}, Q_{a,2}^{(m,n)}, \dots$ denote the elements of $\mathcal{Q}_a^{(m,n)}$ that have been ordered. The above convex combinations $\sum_k u_k^{(m,n)} Q_{a,k}^{(m,n)}$ ensure that subalgorithm $\mathcal{B}_{a,j}$ is competitive against the best elements in $\mathcal{Q}_a^{(m,n)}$ on subintervals $I_a^{(m,n)}$ for all m and n. The weight vectors formed by this subalgorithm $\mathcal{B}_{a,j}$ (when $x_t \in I_a$) are denoted by $\widehat{\boldsymbol{u}}_{t,a,j}^{(m,n)}$, and we set for all $j = 1, \dots, \operatorname{card} \mathcal{P}_a^{(0)}$

$$\widehat{f}_{t,a,j}(x) \stackrel{\text{def}}{=} P_{a,j}^{(0)}(x) + \sum_{m=1}^{M} \sum_{n=1}^{4^m} \sum_{k=1}^{\operatorname{card} \mathcal{Q}_a^{(m,n)}} \widehat{u}_{t,a,j,k}^{(m,n)} \, Q_{a,k}^{(m,n)}(x) \mathbb{1}_{x \in I_a^{(m,n)}} ,$$

where $P_{a,j}^{(0)}$ is the jth element of $\mathcal{P}_a^{(0)}$.

High-scale aggregation: we aggregate the forecasters above $\widehat{f}_{t,a,j}$ for $j \in \{1, \dots, \operatorname{card} \mathcal{P}_a^{(0)}\}$ with a standard Exponentially Weighted Average forecaster (tuned, e.g., with the parameter $\eta = 1/(2(5B)^2) = 1/(50B^2)$):

$$\widehat{f}_{t,a} = \sum_{j=1}^{\operatorname{card} \mathcal{P}_a^{(0)}} \widehat{w}_{t,a,j} \widehat{f}_{t,a,j} \, . \tag{5.28}$$

Putting all things together: at every time $t \geqslant 1$, we make the prediction

$$\widehat{f}_t(x_t) \stackrel{\text{def}}{=} \sum_{a=1}^{1/\delta_x} \widehat{f}_{t,a}(x_t) \mathbb{1}_{x_t \in I_a} \, .$$

We call this algorithm the *Nested Chaining Algorithm for Hölder functions*.

Theorem 5.7. *Let $B > 0$, $T \geqslant 2$, and $\mathcal{F}$ be the set of Hölder functions defined in (5.22). Assume that $\beta \stackrel{\text{def}}{=} q + \alpha \geqslant 1/2$ and that $\max_{1 \leqslant t \leqslant T} |y_t| \leqslant B$. Then, the Nested Chaining Algorithm for Hölder functions defined above and tuned with the parameters $\delta_x = 2(q!\gamma/(2\lambda))^{1/\beta}$, $\delta_y = \gamma/e$, $\gamma = BT^{-\beta/(2\beta+1)}$ and $M = \lceil \log_2(\gamma T/B) \rceil$ satisfies, for some constant $c > 0$ depending only on q and λ,*

$$\operatorname{Reg}_T(\mathcal{F}) \leqslant c \max\{B^{2-1/\beta}, B^2\} T^{\frac{1}{2\beta+1}} (\log T)^{3/2} \, .$$

The proof is postponed to Appendix 5.C.5. The logarithmic factor $(\log T)^{3/2}$ can be reduced to $\log T$, by partitioning I_a into $2^{m/\beta}$ subintervals $I_a^{(m,n)}$ instead of 4^m subintervals. However, the partition at level $m \geqslant 2$ is then not necessarily nested in the partitions of lower levels, which makes the proof slightly more difficult.

Note that the Nested Chaining Algorithm for Hölder functions is computationally tractable as shown by the following lemma, whose proof is deferred to Appendix 5.C.6.

Lemma 5.8. *Under the assumptions of Theorem 5.7, the complexity of the Nested Chaining Algorithm for Hölder functions defined above satisfies:*

- *Storage complexity:* $\mathcal{O}\left(T^{2q+4+\frac{\beta(q-1)+1}{2\beta+1}} \log T\right)$;
- *Time complexity:* $\mathcal{O}\left(T^{(q+1)\left(2+\frac{\beta}{2\beta+1}\right)} \log T\right)$.

5.C. Omitted proofs

In this appendix, we provide the proofs which were omitted in the main body of the chapter.

5.C.1. Proof of Theorem 5.2

As is the case for the classical Exponentiated Gradient algorithm, the proof relies on a linearization argument. Let $\left(\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(K)}\right) \in \Delta_{N_1} \times \dots \times \Delta_{N_K}$. By differentiability and joint convexity of ℓ_t for all $t = 1, \dots, T$, we have that

$$\begin{aligned}
\sum_{t=1}^{T} \ell_t\Big(\widehat{\boldsymbol{u}}_t^{(1)}, \dots, \widehat{\boldsymbol{u}}_t^{(K)}\Big) - \sum_{t=1}^{T} \ell_t\Big(\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(K)}\Big) & \\
\leqslant \sum_{t=1}^{T} \nabla \ell_t\Big(\widehat{\boldsymbol{u}}_t^{(1)}, \dots, \widehat{\boldsymbol{u}}_t^{(K)}\Big) \cdot \Big(\widehat{\boldsymbol{u}}_t^{(1)} - \boldsymbol{u}^{(1)}, \dots, \widehat{\boldsymbol{u}}_t^{(K)} - \boldsymbol{u}^{(K)}\Big) & \qquad (5.29) \\
= \sum_{t=1}^{T} \sum_{k=1}^{K} \nabla_{\widehat{\boldsymbol{u}}_t^{(k)}} \ell_t\Big(\widehat{\boldsymbol{u}}_t^{(1)}, \dots, \widehat{\boldsymbol{u}}_t^{(K)}\Big) \cdot \Big(\widehat{\boldsymbol{u}}_t^{(k)} - \boldsymbol{u}^{(k)}\Big) \ , & \qquad (5.30)
\end{aligned}$$

where $\nabla \ell_t$ in (5.29) denotes the usual (joint) gradient of ℓ_t (with $\sum_{k=1}^{K} N_k$ components), and where (5.30) follows from splitting the gradient into K partial gradients: $\nabla \ell_t = \Big(\nabla_{\widehat{\boldsymbol{u}}_t^{(1)}} \ell_t, \dots, \nabla_{\widehat{\boldsymbol{u}}_t^{(K)}} \ell_t\Big)$.

As a consequence, setting $\boldsymbol{g}_t^{(k)} \stackrel{\text{def}}{=} \nabla_{\widehat{\boldsymbol{u}}_t^{(k)}} \ell_t\Big(\widehat{\boldsymbol{u}}_t^{(1)}, \dots, \widehat{\boldsymbol{u}}_t^{(K)}\Big) \in \mathbb{R}^{N_k}$, and taking the maximum of the last inequality over all $\left(\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(K)}\right) \in \Delta_{N_1} \times \dots \times \Delta_{N_K}$, we can see that

$$\begin{aligned}
\sum_{t=1}^{T} \ell_t\Big(\widehat{\boldsymbol{u}}_t^{(1)}, \dots, \widehat{\boldsymbol{u}}_t^{(K)}\Big) - \min_{\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(K)}} \sum_{t=1}^{T} \ell_t\Big(\boldsymbol{u}^{(1)}, \dots, \boldsymbol{u}^{(K)}\Big) & \\
\leqslant \sum_{k=1}^{K} \max_{\boldsymbol{u}^{(k)} \in \Delta_{N_k}} \sum_{t=1}^{T} \boldsymbol{g}_t^{(k)} \cdot \Big(\widehat{\boldsymbol{u}}_t^{(k)} - \boldsymbol{u}^{(k)}\Big) & \\
= \sum_{k=1}^{K} \left(\sum_{t=1}^{T} \boldsymbol{g}_t^{(k)} \cdot \widehat{\boldsymbol{u}}_t^{(k)} - \min_{1 \leqslant i \leqslant N_k} \sum_{t=1}^{T} g_{t,i}^{(k)} \right) \ , & \qquad (5.31)
\end{aligned}$$

where the last inequality follows from the fact that the function $\boldsymbol{u}^{(k)} \mapsto \sum_{t=1}^{T} \boldsymbol{g}_t^{(k)} \cdot \boldsymbol{u}^{(k)}$ is linear over the polytope Δ_{N_k}, so that its minimum is achieved on at least one of the N_k vertices of Δ_{N_k}.

Note that the right-hand side of (5.31) is the sum of K regrets. Let $k \in \{1, \dots, K\}$. By definition of the Multi-variable Exponentiated Gradient algorithm, the sequence of weight vectors $\left(\widehat{\boldsymbol{u}}_t^{(k)}\right)_{t \geqslant 1}$ corresponds exactly to the weight vectors output by the Exponentially Weighted Average forecaster (see Page 14 of Cesa-Bianchi and Lugosi [43]) applied to N_k experts associated with the loss vectors $\boldsymbol{g}_t^{(k)} \in \mathbb{R}^{N_k}$, $t \geqslant 1$. We can therefore use the well-known corresponding regret bound available, e.g., in Theorem 2.2 of Cesa-Bianchi and Lugosi [43] or in Theorem 2.1 of Gerchinovitz [78]. Noting that the loss vectors $\boldsymbol{g}_t^{(k)}$ lie in $\left[-G^{(k)}, G^{(k)}\right]^{N_k}$ by Assumption (5.3), we thus get that

$$\sum_{t=1}^{T} \boldsymbol{g}_t^{(k)} \cdot \widehat{\boldsymbol{u}}_t^{(k)} - \min_{1 \leqslant i \leqslant N_k} \sum_{t=1}^{T} g_{t,i}^{(k)} \leqslant G^{(k)} \sqrt{2T \log N_k} \ .$$

substituting the last upper bound in the right-hand side of (5.31) concludes the proof.

5.C.2. An efficient γ-net for Lipschitz classes

Lemma 5.9. *Let $\mathcal{F}$ be the set of functions from $[0,1]$ to $[-B,B]$ that are 1-Lipschitz. Then:*

- *the set $\mathcal{F}^{(0)}$ defined in (5.16) is a γ-net of $\big(\mathcal{F}, \|\cdot\|_\infty\big)$;*
- *for all $M \geqslant 1$, the set $\mathcal{F}^{(M)}$ defined in (5.17) is a $\gamma/2^{M+1}$-net of $\big(\mathcal{F}, \|\cdot\|_\infty\big)$.*

Proof (of Lemma 5.9)
First claim: $\mathcal{F}^{(0)}$ is a γ-net of $\big(\mathcal{F}, \|\cdot\|_\infty\big)$.
Let $f \in \mathcal{F}$. We explain why there exist $c_1^{(0)}, \dots, c_{1/\gamma}^{(0)} \in C^{(0)}$ such that

$$f^{(0)}(x) = \sum_{a=1}^{1/\gamma} c_a^{(0)} \mathbb{1}_{x \in I_a}$$

satisfies $\big|f(x) - f^{(0)}(x)\big| \leqslant \gamma$ for all $x \in [0,1]$. We can choose $c_a^{(0)} \in \arg\min_{c \in C^{(0)}} \big|f(x_a) - c\big|$, where x_a is the center of the subinterval I_a. Indeed, since we can approximate $f(x_a)$ with precision $\gamma/2$ (the y-axis discretization is of width γ), and since f is 1-Lipschitz on I_a, we have that, for all $a \in \{1, \dots, 1/\gamma\}$ and all $x \in I_a$,

$$\big|f(x) - c_a^{(0)}\big| \leqslant \big|f(x) - f(x_a)\big| + \big|f(x_a) - c_a^{(0)}\big| \leqslant \frac{\gamma}{2} + \frac{\gamma}{2} = \gamma \ .$$

Since the subintervals I_a, $a \leqslant 1/\gamma$, form a partition of $[0,1]$, we just showed that $\|f - f^{(0)}\|_\infty \leqslant \gamma$.

Second claim: $\mathcal{F}^{(M)}$ is a $\gamma/2^{M+1}$-net of $\big(\mathcal{F}, \|\cdot\|_\infty\big)$.
Let $f \in \mathcal{F}$. We explain why there exist constants $c_a^{(0)} \in C^{(0)}$ and $c_a^{(m,n)} \in \big[-\gamma/2^{m-1}, \gamma/2^{m-1}\big]$ such that

$$f_{\boldsymbol{c}}(x) = \sum_{a=1}^{1/\gamma} c_a^{(0)} \mathbb{1}_{x \in I_a} + \sum_{m=1}^{M} \sum_{a=1}^{1/\gamma} \sum_{n=1}^{2^m} c_a^{(m,n)} \mathbb{1}_{x \in I_a^{(m,n)}}$$

satisfies $\big|f(x) - f_{\boldsymbol{c}}(x)\big| \leqslant \gamma/2^{M+1}$ for all $x \in [0,1]$. We argue below that it suffices to:

- choose the constants $c_a^{(0)} \in \arg\min_{c \in C^{(0)}} \big|f(x_a) - c\big|$ exactly as for $\mathcal{F}^{(0)}$ above;
- choose the constants $c_a^{(m,n)}$ in such a way that, for all levels $m \in \{1, \dots, M\}$, and for all positions $a \in \{1, \dots, 1/\gamma\}$ and $n \in \{1, \dots, 2^m\}$,

$$f\big(x_a^{(m,n)}\big) = c_a^{(0)} + \sum_{m'=1}^{m} c_a^{(m', n_{m'})} \ , \tag{5.32}$$

  where $x_a^{(m,n)}$ denotes the center of the subinterval $I_a^{(m,n)}$, and where $n_{m'}$ is the unique integer n' such that $I_a^{(m,n)} \subseteq I_a^{(m',n')}$. Such a choice can be done in a recursive way (induction on m). It is feasible since the functions in $\mathcal{F}$ are 1-Lipschitz (see Figure 5.1 for an illustration).

To conclude, it is now sufficient to use (5.32) with $m = M$. Note indeed from (5.17) that, on each

level-M subinterval $I_a^{(M,n)}$, the function $f_{\boldsymbol{c}}$ is equal to

$$f_{\boldsymbol{c}}(x) = c_a^{(0)} + \sum_{m=1}^{M} c_a^{(m,n_m)} \,,$$

where n_m is the unique integer n' such that $I_a^{(M,n)} \subseteq I_a^{(m,n')}$. Thus, by (5.32), we can see that $f_{\boldsymbol{c}}\big(x_a^{(M,n)}\big) = f\big(x_a^{(M,n)}\big)$ for all points $x_a^{(M,n)}$, $a \in \{1,\dots,1/\gamma\}$ and $n \in \{1,\dots,2^M\}$.

Now, if $x \in I_a^{(M,n)}$ is any point in $I_a^{(M,n)}$, then it is at most at a distance of $\gamma/2^{M+1}$ of the middle point $x_a^{(M,n)}$. Therefore, by 1-Lipschitzity of f, we have $\big|f(x) - f\big(x_a^{(M,n)}\big)\big| \leqslant \gamma/2^{M+1}$. Using the equality $f_{\boldsymbol{c}}\big(x_a^{(M,n)}\big) = f\big(x_a^{(M,n)}\big)$ proved above and the fact that $f_{\boldsymbol{c}}$ is constant on $I_a^{(M,n)}$, we get that

$$\forall a \in \{1,\dots,1/\gamma\}, \quad \forall n \in \{1,\dots,2^M\}, \quad \forall x \in I_a^{(M,n)} \,, \qquad \big|f(x) - f_{\boldsymbol{c}}(x)\big| \leqslant \gamma/2^{M+1} \,.$$

Since the level-M subintervals $I_a^{(M,n)}$, $a \in \{1,\dots,1/\gamma\}$ and $n \in \{1,\dots,2^M\}$, form a partition of $[0,1]$, we just showed that $\|f - f_{\boldsymbol{c}}\|_\infty \leqslant \gamma/2^{M+1}$, which concludes the proof. ∎

5.C.3. An efficient γ-net for Hölder classes (proof of Lemma 5.6)

First claim: $\mathcal{F}^{(0)}$ is a γ-net of $\big(\mathcal{F}, \|\cdot\|_\infty\big)$.
Let $f \in \mathcal{F}$. We explain why there exist $P_a^{(0)} \in \mathcal{P}_a^{(0)}$ for all $a \in \{1,\dots,1/\delta_x\}$ such that

$$f^{(0)}(x) = \sum_{a=1}^{1/\delta_x} P_a^{(0)}(x) \mathbb{1}_{x \in I_a}$$

satisfies $\big|f(x) - f^{(0)}(x)\big| \leqslant \gamma$ for all $x \in [0,1]$. Fix $a \in \{1,\dots,1/\delta_x\}$ and let x_a be the center of the subinterval I_a. By Taylor's formula for all $x \in I_a$ there exist $\xi \in I_a$ such that

$$f(x) = f(x_a) + f'(x_a)(x - x_a) + \frac{f''(x_a)}{2!}(x - x_a)^2 + \dots + \frac{f^{(q)}(x_a)}{q!}(x - x_a)^q + \frac{1}{q!}\Big(f^{(q)}(\xi) - f^{(q)}(x_a)\Big)(x - x_a)^q \,. \quad (5.33)$$

Thus, the function f can be written as the sum of a polynomial and a term (the last one) that will be proven to be small by the Hölder property (5.22). Now, for every derivative $i \in \{0,\dots,q\}$ we can choose $b_i \in \mathcal{Y}^{(0)}$ such that

$$|f^{(i)}(x_a) - b_i| \leqslant \delta_y/2 \,. \quad (5.34)$$

Indeed, the y-axis discretization $\mathcal{Y}^{(0)}$ of $[-B,B]$ is of width δ_y and $|f^{(i)}(x_a)| \leqslant B$ by definition of $\mathcal{F}$. Thus, setting

$$P_a^{(0)}(x) = b_0 + \frac{b_1}{1}(x - x_a) + \frac{b_2}{2!}(x - x_a)^2 + \dots + \frac{b_q}{q!}(x - x_a)^q \,,$$

the polynomial $P_a^{(0)}$ satisfies by (5.33) for all $x \in I_a$

$$\begin{aligned}\left|f(x) - P_a^{(0)}(x)\right| &\leqslant \sum_{i=0}^{q} \frac{\left|f^{(i)}(x_a) - b_i\right|}{i!}|x - x_a|^i + \frac{1}{q!}\left|f^{(q)}(\xi) - f^{(q)}(x_a)\right||x - x_a|^q \\ &\leqslant \sum_{i=0}^{q} \frac{\delta_y}{2i!}\underbrace{|x - x_a|^i}_{\leqslant 1} + \frac{\lambda}{q!}|\xi - x_a|^\alpha |x - x_a|^q ,\end{aligned}$$

where the second inequality is by (5.34) and because $f^{(q)}$ is α-Hölder with coefficient λ. Now, since $|\xi - x_a|$ and $|x - x_a|$ are bounded by $\delta_x/2$, it yields

$$\left|f(x) - P_a^{(0)}(x)\right| \leqslant \sum_{i=0}^{q} \frac{\delta_y}{2i!} + \frac{\lambda}{q!}\left(\frac{\delta_x}{2}\right)^{q+\alpha} \leqslant \frac{e}{2}\delta_y + \frac{\lambda}{q!}\left(\frac{\delta_x}{2}\right)^{\beta} .$$

The choices $\delta_x = 2(q!\gamma/(2\lambda))^{1/\beta}$ and $\delta_y = \gamma/e$ finally entail

$$\left|f(x) - \left[P_a^{(0)}(x)\right]_B\right| \leqslant \left|f(x) - P_a^{(0)}(x)\right| \leqslant \frac{\gamma}{2} + \frac{\gamma}{2} = \gamma .$$

This concludes the first part of the proof.

Second claim: $\mathcal{F}^{(M)}$ is a $\gamma/2^M$-net of $\left(\mathcal{F}, \|\cdot\|_\infty\right)$.
Let $f \in \mathcal{F}$. We explain why there exist clipped-polynomials $P_a^{(0)} \in \mathcal{P}^{(0)}$ and $Q_a^{(m,n)} \in \mathcal{Q}_a^{(m,n)}$ such that

$$f_{\boldsymbol{c}}(x) = \sum_{a=1}^{1/\delta_x} P_a^{(0)}(x)\mathbb{1}_{x\in I_a} + \sum_{m=1}^{M}\sum_{a=1}^{1/\delta_x}\sum_{n=1}^{4^m} Q_a^{(m,n)}(x)\mathbb{1}_{x\in I_a^{(m,n)}}$$

satisfies $\left|f(x) - f_{\boldsymbol{c}}(x)\right| \leqslant \gamma/2^M$ for all $x \in [0,1]$. To do so, we show first that there exist clipped polynomials $P_a^{(0)} \in \mathcal{P}^{(0)}$ and $P_a^{(m,n)} \in \mathcal{P}_a^{(m,n)}$ such that

$$\begin{aligned}\widetilde{f}_{\boldsymbol{c}}(x) = \sum_{a=1}^{1/\delta_x} P_a^{(0)}(x)\mathbb{1}_{x\in I_a} &+ \sum_{n=1}^{4}\sum_{a=1}^{1/\delta_x}\left(P_a^{(1,n)} - P_a^{(0)}\right)(x)\mathbb{1}_{x\in I_a^{(1,n)}} \\ &+ \sum_{m=2}^{M}\sum_{a=1}^{1/\delta_x}\sum_{n=1}^{4^m}\left(P_a^{(m,n)} - P_a^{(m-1,n_{m-1})}\right)(x)\mathbb{1}_{x\in I_a^{(m,n)}}\end{aligned}$$

satisfies $\left|f(x) - \widetilde{f}_{\boldsymbol{c}}(x)\right| \leqslant \gamma/2^M$ for all $x \in [0,1]$. We recall that n_{m-1} denotes the unique integer n' such that $I_a^{(m,n)} \subset I_a^{(m-1,n')}$. First we remark that the function $\widetilde{f}_c$ defined above equals $P_a^{(M,n)}$ on each level-M subinterval $I_a^{(M,n)}$.

Thus, it suffices to design clipped polynomials $P_a^{(m,n)} \in \mathcal{P}_a^{(m,n)}$, such that $\left|f(x) - P_a^{(m,n)}(x)\right| \leqslant \gamma/2^m$ for all $x \in I_a^{(m,n)}$. To do so, we reproduce the same proof as for $\mathcal{F}^{(0)}$ above. Because $\operatorname{diam} I_a^{(m,n)} = \delta_x/4^m \leqslant \delta_x/2^{m/\beta}$ (recall that $\beta \geqslant 1/2$), for every position $a \in \{1, \dots, 1/\delta_x\}$, every level $m \in \{1, \dots, M\}$, and every $n \in \{1, \dots, 4^m\}$, we can define as for $\mathcal{F}^{(0)}$ above a polynomial

$$\widetilde{P}_a^{(m,n)}(x) = b_0 + \frac{b_1}{1}\left(x - x_a^{(m,n)}\right) + \frac{b_2}{2!}\left(x - x_a^{(m,n)}\right)^2 + \cdots + \frac{b_q}{q!}\left(x - x_a^{(m,n)}\right)^q$$

(recall that $x_a^{(m,n)}$ is the center of $I_a^{(m,n)}$) such that all coefficients b_j have the form $-B + z_j \delta_y 2^{-m}$ for some $z_j \in \{0, \dots, 2^{m+1}B/\delta_y\}$ and

$$\left| f(x) - \left[\widetilde{P}_a^{(m,n)}(x)\right]_B \right| \leqslant \gamma/2^m \tag{5.35}$$

for all $x \in I_a^{(m,n)}$. To conclude, we choose the clipped polynomials $P_a^{(m,n)} = \left[\widetilde{P}_a^{(m,n)}\right]_B$.

To conclude the proof, we see that for all $x \in I_a^{(m,n)}$, by the triangle inequality

$$\left| P_a^{(m,n)}(x) - P_a^{(m-1,n_{m-1})}(x) \right| \leqslant \frac{\gamma}{2^m} + \frac{\gamma}{2^{m-1}} = \frac{3\gamma}{2^m} \,,$$

so that $f_c = \widetilde{f}_c$ for the choices $Q_a^{(m,n)} = \left[P_a^{(m,n)}(x) - P_a^{(m-1,n_{m-1})}(x)\right]_{3\gamma/2^m}$.

5.C.4. Proof of Theorem 5.4

We split our proof into two main parts. First, we explain why each functions $\widehat{f}_{t,a}$ incurs small cumulative regret inside each subinterval I_a. Second, we sum the previous regret bounds over all positions $a \in \{1, \dots, 1/\gamma\}$.

Part 1: focus on a subinterval I_a. In this part, we fix some $a \in \{1, \dots, 1/\gamma\}$ and we consider the a-th instance of the algorithm $\mathcal{A}$, whose local time is only incremented when a new x_t falls into I_a. As in Algorithm 9, our instance of algorithm $\mathcal{A}$ uses a combination of the EWA and the Multi-variable EG forecasters to perform high-scale and low-scale aggregation simultaneously. Thus, the proof closely follows the path of the one of Theorem 5.3. We split again the proof into two subparts: one for each level of aggregation.

Subpart 1: low-scale aggregation.

In this subpart, we fix $j \in \{0, \dots, 2B/\gamma\}$. The proof starts as the one of Theorem 5.3 except that $\mathcal{A}$ applies the adaptive version of the Multi-variable Exponentiated Gradient forecaster (Algorithm 10, Appendix 5.A) with the loss function ℓ_t defined in (5.18). We will thus apply Theorem 5.5 (available in Appendix 5.A) instead of Theorem 5.2. After checking its assumptions exactly as in the proof of Theorem 5.3, we can apply Theorem 5.5. The norms of the loss gradients $\left\|\nabla_{\widehat{u}_t^{(m,n)}} \ell_t\right\|_\infty$ are bounded by $16B\gamma/2^m$ if x_t falls in $I_a^{(m,n)}$ and by 0 otherwise. Setting $T_a^{(m,n)} = \sum_{t=1}^T \mathbb{1}_{x_t \in I_a^{(m,n)}}$, Theorem 5.5 yields as in (5.11):

$$\sum_{t=1}^T \left(y_t - \widehat{f}_{t,a,j}(x_t)\right)^2 \mathbb{1}_{x_t \in I_a} \tag{5.36}$$

$$\leqslant \inf_{c_a^{(m,n)},\ \forall (m,n)} \sum_{t=1}^T \left(y_t - \left(-B + j\gamma + \sum_{m=1}^M \sum_{n=1}^{2^m} c_a^{(m,n)} \mathbb{1}_{x_t \in I_a^{(m,n)}} \right) \right)^2 \mathbb{1}_{x_t \in I_a}$$

$$+ 2 \sum_{m=1}^M \sum_{n=1}^{2^m} 16B\gamma/2^m \sqrt{T_a^{(m,n)} \log 2} \,, \tag{5.37}$$

where the infimum is over all constants $c_a^{(m,n)} \in [-\gamma/2^{m-1}, \gamma/2^{m-1}]$ for every $m = 1, \dots, M$ and $n = 1, \dots, 2^m$. But, for each level $m = 1, \dots, M$, the point x_t only falls into one interval $I_a^{(m,n)}$.

Thus,

$$\sum_{n=1}^{2^m} T_a^{(m,n)} = T_a\,,$$

where $T_a = \sum_{t=1}^{T} \mathbb{1}_{x_t \in I_a}$ is the final local time of the a-th instance of $\mathcal{A}$. Therefore, using the concavity of the square root and applying Jensen's inequality, (5.37) entails

$$\begin{aligned}
&\sum_{t=1}^{T} \Big(y_t - \widehat{f}_{t,a,j}(x_t)\Big)^2 \mathbb{1}_{x_t \in I_a} \\
&\qquad \leqslant \inf_{c_a^{(m,n)},\ \forall (m,n)} \ \sum_{t=1}^{T} \left(y_t - \left(-B + j\gamma + \sum_{(m,n)} c_a^{(m,n)} \mathbb{1}_{x_t \in I_a^{(m,n)}} \right)\right)^2 \mathbb{1}_{x_t \in I_a} \\
&\qquad\qquad\qquad + 32B\gamma \sum_{m=1}^{M} 2^{-m} \sqrt{T_a 2^m \log 2} \\
&\qquad \leqslant \inf_{c_a^{(m,n)},\ \forall (m,n)} \ \sum_{t=1}^{T} \left(y_t - \left(-B + j\gamma + \sum_{(m,n)} c_a^{(m,n)} \mathbb{1}_{x_t \in I_a^{(m,n)}} \right)\right)^2 \mathbb{1}_{x_t \in I_a} \\
&\qquad\qquad\qquad + 32B\gamma(1+\sqrt{2}) \sqrt{T_a \log 2}\,. \qquad (5.38)
\end{aligned}$$

The second inequality is because $\sum_{m=1}^{\infty} 2^{-m/2} = 1 + \sqrt{2}$.

Subpart 2: high-scale aggregation.
Following the proof of Theorem 5.3, we apply EWA to the experts $\widehat{f}_{t,a,j}$ for $j \in \{0, \dots, 2B/\gamma\}$ with tuning parameter $\eta = 1/(2(4B)^2)$ because $\widehat{f}_{t,a,j} \in [-B-2\gamma, B+2\gamma] \subset [-3B, 3B]$ and $y_t \in [-B, B]$. We get from Proposition 3.1 and Page 46 of Cesa-Bianchi and Lugosi [43] that

$$\begin{aligned}
&\sum_{t=1}^{T} \Big(y_t - \widehat{f}_{t,a}(x_t)\Big)^2 \mathbb{1}_{x_t \in I_a} \\
&\qquad \leqslant \min_{0 \leqslant j \leqslant 2B/\gamma} \ \sum_{t=1}^{T} \Big(y_t - \widehat{f}_{t,a,j}(x_t)\Big)^2 \mathbb{1}_{x_t \in I_a} + \frac{\log\big(2B/\gamma + 1\big)}{\eta} \\
&\qquad \leqslant \min_{0 \leqslant j \leqslant 2B/\gamma} \ \inf_{c_a^{(m,n)}, \forall (m,n)} \ \sum_{t=1}^{T} \left(y_t - \left(-B + j\gamma + \sum_{(m,n)} c_a^{(m,n)} \mathbb{1}_{x \in I_a^{(m,n)}} \right)\right)^2 \mathbb{1}_{x_t \in I_a} \\
&\qquad\qquad\qquad + 32B\big(1+\sqrt{2}\big)\gamma \sqrt{T_a \log 2} \ + 32B^2 \log\big(2B/\gamma + 1\big)\ , \qquad (5.39)
\end{aligned}$$

where the infima are over all $j \in \{0, \dots, 2B/\gamma\}$ and all constants $c_a^{(m,n)} \in [-\gamma/2^{m-1}, \gamma/2^{m-1}]$, and where the second inequality follows from (5.38) and from $\eta = 1/(32B^2)$.

Part 2: we sum the regrets over all subintervals I_a By definition of $\widehat{f}_t$, we have

$$\begin{aligned}
\sum_{t=1}^{T} \big(y_t - \widehat{f}_t(x_t)\big)^2 &= \sum_{t=1}^{T} \left(y_t - \sum_{a=1}^{1/\gamma} \widehat{f}_{t,a}(x_t) \mathbb{1}_{x_t \in I_a}\right)^2 \\
&= \sum_{a=1}^{1/\gamma} \sum_{t=1}^{T} \Big(y_t - \widehat{f}_{t,a}(x_t)\Big)^2 \mathbb{1}_{x_t \in I_a}
\end{aligned}$$

Now, by definition of $\mathcal{F}^{(M)}$, summing (5.39) over all $a = 1, \dots, 1/\gamma$ leads to

$$\sum_{t=1}^{T} \big(y_t - \widehat{f}_t(x_t)\big)^2 \leqslant \inf_{f \in \mathcal{F}^{(M)}} \sum_{t=1}^{T} \big(y_t - f(x_t)\big)^2 + \frac{32B^2}{\gamma} \log\left(2B/\gamma + 1\right) + 32B\big(1+\sqrt{2}\big)\gamma\sqrt{\log 2}\left(\sum_{a=1}^{1/\gamma} \sqrt{T_a}\right) . \tag{5.40}$$

Then, using that $\sum_{a=1}^{1/\gamma} T_a = T$, since at every round t, the point x_t only falls into one subinterval I_a, and applying Jensen's inequality to the square root, we can see that

$$\sum_{a=1}^{1/\gamma} \sqrt{T_a} \leqslant \sqrt{\frac{T}{\gamma}} .$$

Therefore, substituting in (5.40), we obtain

$$\sum_{t=1}^{T} \big(y_t - \widehat{f}_t(x_t)\big)^2 \leqslant \inf_{f \in \mathcal{F}^{(M)}} \sum_{t=1}^{T} \big(y_t - f(x_t)\big)^2 + \frac{32B^2}{\gamma} \log\left(2B/\gamma + 1\right) + 32B\big(1+\sqrt{2}\big)\sqrt{\gamma T \log 2} . \tag{5.41}$$

But, $\mathcal{F}^{(M)}$ is by Lemma 5.9 a $\gamma/2^{M+1}$-net of $\mathcal{F}$. Using that $M = \lceil \log_2(\gamma T/B) \rceil$ and following the proof of (5.15), it entails

$$\inf_{f \in \mathcal{F}^{(M)}} \sum_{t=1}^{T} \big(y_t - f(x_t)\big)^2 \leqslant \inf_{f \in \mathcal{F}} \sum_{t=1}^{T} \big(y_t - f(x_t)\big)^2 + 2B^2 + \frac{B^2}{4T} .$$

Finally, from (5.41) we have

$$\sum_{t=1}^{T} \big(y_t - \widehat{f}_t(x_t)\big)^2 \leqslant \inf_{f \in \mathcal{F}} \sum_{t=1}^{T} \big(y_t - f(x_t)\big)^2 + \frac{32B^2}{\gamma} \log\left(2B/\gamma + 1\right) + 32B\big(1+\sqrt{2}\big)\sqrt{\gamma T \log 2} + 2B^2 + \frac{B^2}{4T} . \tag{5.42}$$

The above regret bound grows roughly as (we omit logarithmic factors and small additive terms):

$$\gamma^{-1} + \sqrt{\gamma T} .$$

Optimizing in γ would yield $\gamma \approx T^{-1/3}$ and a regret roughly of the order of $T^{1/3}$. More rigorously, taking $\gamma = BT^{-1/3}$ and substituting it in (5.42) concludes the proof.

5.C.5. Proof of Theorem 5.7

The proof closely follows the one of Theorem 5.4. It is split into two main parts. First, we explain why each function $\widehat{f}_{t,a}$ incurs a small cumulative regret inside each subinterval I_a. Second, we sum the previous regret bounds over all positions $a = 1, \dots, 1/\delta_x$.

Part 1: focus on a subinterval I_a In this part, we fix some $a \in \{1, \dots, 1/\delta_x\}$ and we consider the a-th instance of the algorithm $\mathcal{A}$, denoted $\mathcal{A}_a$, whose local time is only incremented when a new x_t falls into I_a. As in Algorithm 9, $\mathcal{A}_a$ uses a combination of the EWA and the Multi-variable EG forecasters to perform high-scale and low-scale aggregation simultaneously. We split again the proof into two subparts: one for each level of aggregation.

Subpart 1: low-scale aggregation.

In this subpart, we fix $j \in \{1, \dots, \operatorname{card} \mathcal{P}_a^{(0)}\}$. Similarly to the proof of Theorem 5.4, we start by applying Theorem 5.5. Since the elements in $\mathcal{Q}_a^{(m,n)}$ are bounded in supremum norm by $3\gamma/2^m$, and since the elements in $\mathcal{P}_a^{(0)}$ are bounded by B, the norms of the gradients of the loss function (defined in (5.27)) are bounded by 0 if $x_t \notin I_a^{(m,n)}$ and as follows otherwise:

$$\left\|\nabla_{\widehat{u}_{t,a,j}^{(m,n)}} \ell_t\right\|_\infty \leqslant 2\Big(|y_t| + \big\|\widehat{f}_{t,a,j}\big\|_\infty\Big)\left\|Q_{a,k}^{(m,n)}\right\|_\infty \leqslant 2(B+4B)3\gamma/2^m = 30B\gamma/2^m\,.$$

Here, we used that

$$\big|\widehat{f}_{t,a,j}(x)\big| \leqslant \big\|P_{a,j}^{(0)}\big\|_\infty + \sum_{m=1}^{M}\sum_{n=1}^{4^m}\sum_{k=1}^{\operatorname{card}\mathcal{Q}_a^{(m,n)}} \widehat{u}_{t,a,j,k}^{(m,n)}\Big|Q_{a,k}^{(m,n)}(x)\Big|\mathbb{1}_{x\in I_a^{(m,n)}} \leqslant B + \sum_{m=1}^{M}\frac{3\gamma}{2^m} \leqslant 4B\,. \quad (5.43)$$

Thus, setting $T_a^{(m,n)} = \sum_{t=1}^{T} \mathbb{1}_{x_t \in I_a^{(m,n)}}$, Theorem 5.5 yields:

$$\begin{aligned}
&\sum_{t=1}^{T}\Big(y_t - \widehat{f}_{t,a,j}(x_t)\Big)^2 \mathbb{1}_{x_t\in I_a} \\
&\quad\leqslant \inf_{Q_a^{(m,n)},\ \forall (m,n)} \sum_{t=1}^{T}\left(y_t - \left(P_{a,j}^{(0)} + \sum_{m=1}^{M}\sum_{n=1}^{4^m} Q_a^{(m,n)}\mathbb{1}_{x_t\in I_a^{(m,n)}}\right)(x_t)\right)^2 \mathbb{1}_{x_t\in I_a} \\
&\qquad\qquad + 2\sum_{m=1}^{M}\sum_{n=1}^{4^m} 30B\gamma/2^m\sqrt{T_a^{(m,n)}\log\Big(\operatorname{card}\mathcal{Q}_a^{(m,n)}\Big)}\,, \qquad (5.44)
\end{aligned}$$

where the infimum is over all polynomial functions $Q_a^{(m,n)} \in \mathcal{Q}_a^{(m,n)}$ for every $m = 1, \dots, M$ and $n = 1, \dots, 4^m$. But, for each level $m = 1, \dots, M$, the point x_t only falls into one interval $I_a^{(m,n)}$. Thus, $\sum_{n=1}^{4^m} T_a^{(m,n)} = T_a$, where $T_a = \sum_{t=1}^{T} \mathbb{1}_{x_t\in I_a}$ is the final local time of the a-th instance of $\mathcal{A}$. Therefore, using the concavity of the square root and applying Jensen's inequality, (5.44) entails

$$\begin{aligned}
&\sum_{t=1}^{T}\Big(y_t - \widehat{f}_{t,a,j}(x_t)\Big)^2 \mathbb{1}_{x_t\in I_a} \qquad (5.45)\\
&\quad\leqslant \inf_{Q_a^{(m,n)},\ \forall (m,n)} \sum_{t=1}^{T}\left(y_t - \left(P_{a,j}^{(0)} + \sum_{(m,n)} Q_a^{(m,n)}\mathbb{1}_{x_t\in I_a^{(m,n)}}\right)(x_t)\right)^2 \mathbb{1}_{x_t\in I_a} \\
&\qquad\qquad + 60B\gamma\sum_{m=1}^{M} 2^{-m}\sqrt{T_a 4^m \log\Big(\operatorname{card}\mathcal{Q}_a^{(m,n)}\Big)}\,. \qquad (5.46)
\end{aligned}$$

Now, by the definitions of $\mathcal{Q}_a^{(m,n)}$, $\mathcal{P}_a^{(m,n)}$, and $\mathcal{Y}^{(m)}$ (see Equations (5.24) and (5.25)), we get

$$\operatorname{card} \mathcal{Q}_a^{(m,n)} \leqslant \operatorname{card}\left(\mathcal{P}_a^{(m,n)}\right)^2 \leqslant \left(\operatorname{card} \mathcal{Y}^{(m)}\right)^{2(q+1)} = \left(2^{m+1}\frac{B}{\delta_y}+1\right)^{2(q+1)} = \left(2^{m+1}e\frac{B}{\gamma}+1\right)^{2(q+1)},$$

which yields

$$\begin{aligned}\sum_{m=1}^{M} 2^{-m}\sqrt{4^m \log\left(\operatorname{card} \mathcal{Q}_a^{(m,n)}\right)} &\leqslant \sum_{m=1}^{M}\sqrt{2(q+1)\log\left(2^{m+1}eB/\gamma+1\right)} \\ &\leqslant M\sqrt{2(q+1)\log\left(2^{M+1}eB/\gamma+1\right)}\,.\end{aligned}$$

Thus, using $M = \lceil \log_2(\gamma T/B)\rceil$, so that $2^M\gamma^{-1} \leqslant 2T/B$ and combining the above inequality with (5.46), we have

$$\sum_{t=1}^{T}\left(y_t - \widehat{f}_{t,a,j}(x_t)\right)^2 \mathbb{1}_{x_t\in I_a} \tag{5.47}$$

$$\begin{aligned}&\leqslant \inf_{Q_a^{(m,n)},\ \forall(m,n)} \sum_{t=1}^{T}\left(y_t - \left(P_{a,j}^{(0)} + \sum_{(m,n)} Q_a^{(m,n)}\mathbb{1}_{x_t\in I_a^{(m,n)}}\right)(x_t)\right)^2 \mathbb{1}_{x_t\in I_a} \\ &\qquad + 60B\gamma\lceil\log_2(\gamma T/B)\rceil\sqrt{2(q+1)T_a\log(4eT+1)}\,.\end{aligned} \tag{5.48}$$

Subpart 2: high-scale aggregation.

Following the proof of Theorem 5.4, we apply EWA to the experts $\widehat{f}_{t,a,j}$ for $j \in \{1,\dots,\operatorname{card}\mathcal{P}_a^{(0)}\}$ with tuning parameter $\eta = 1/(2(5B)^2) = 1/(50B^2)$ because $\widehat{f}_{t,a,j} \in [-4B, 4B]$ (see (5.43)). From (5.48) and using $\operatorname{card}\mathcal{P}_a^{(0)} \leqslant (2B/\delta_y+1)^{q+1} = (2eB/\gamma+1)^{q+1}$, we have

$$\begin{aligned}&\sum_{t=1}^{T}\left(y_t - \widehat{f}_{t,a}(x_t)\right)^2\mathbb{1}_{x_t\in I_a} \\ &\quad\leqslant \min_{1\leqslant j\leqslant \operatorname{card}\mathcal{P}_a^{(0)}} \sum_{t=1}^{T}\left(y_t - \widehat{f}_{t,a,j}(x_t)\right)^2\mathbb{1}_{x_t\in I_a} + \frac{\log\left(\operatorname{card}\mathcal{P}_a^{(0)}\right)}{\eta} \\ &\quad\leqslant \inf_{P_a^{(0)}\in\mathcal{P}_a^{(0)}}\ \inf_{Q_a^{(m,n)},\ \forall(m,n)} \sum_{t=1}^{T}\left(y_t - \left(P_a^{(0)} + \sum_{(m,n)} Q_a^{(m,n)}\mathbb{1}_{x_t\in I_a^{(m,n)}}\right)(x_t)\right)^2\mathbb{1}_{x_t\in I_a} \\ &\qquad\qquad + 60B\gamma\lceil\log_2(\gamma T/B)\rceil\sqrt{2(q+1)T_a\log(4eT+1)} \\ &\qquad\qquad + 50B^2(q+1)\log\left(2eB/\gamma+1\right)\ ,\end{aligned} \tag{5.49}$$

where the infimum is over all functions $P^{(0)} \in \mathcal{P}_a^{(0)}$ and $Q_a^{(m,n)} \in \mathcal{Q}_a^{(m,n)}$, and where the second inequality follows from $\eta = 1/(50B^2)$.

Part 2: we sum the regrets over all subintervals I_a By definition of $\widehat{f}_t$, we have

$$\sum_{t=1}^{T}\left(y_t - \widehat{f}_t(x_t)\right)^2 = \sum_{t=1}^{T}\left(y_t - \sum_{a=1}^{1/\delta_x}\widehat{f}_{t,a}(x_t)\mathbb{1}_{x_t\in I_a}\right)^2 = \sum_{a=1}^{1/\delta_x}\sum_{t=1}^{T}\left(y_t - \widehat{f}_{t,a}(x_t)\right)^2\mathbb{1}_{x_t\in I_a}$$

Now, by definition of $\mathcal{F}^{(M)}$, summing (5.49) over all $a = 1, \dots, 1/\delta_x$ leads to

$$\begin{aligned}\sum_{t=1}^{T} \big(y_t - \widehat{f}_t(x_t)\big)^2 \leqslant \inf_{f \in \mathcal{F}^{(M)}} \sum_{t=1}^{T} \big(y_t - f(x_t)\big)^2 \\ + 50B^2(q+1)\log(2eB/\gamma + 1)\delta_x^{-1} \\ + 60B\gamma\lceil \log_2(\gamma T/B)\rceil \sqrt{2(q+1)\log(4eT+1)} \left(\sum_{a=1}^{1/\delta_x} \sqrt{T_a}\right). \quad (5.50)\end{aligned}$$

Then, using that $\sum_{a=1}^{1/\delta_x} T_a = T$, since at every round t, the point x_t only falls into one subinterval I_a, and applying Jensen's inequality to the square root, we can see that

$$\sum_{a=1}^{1/\delta_x} \sqrt{T_a} \leqslant \sqrt{T/\delta_x}\,.$$

Therefore, substituting in (5.50) and because $\delta_x = 2(q!\gamma/(2\lambda))^{1/\beta}$, we have

$$\begin{aligned}\sum_{t=1}^{T} \big(y_t - \widehat{f}_t(x_t)\big)^2 \leqslant \inf_{f \in \mathcal{F}^{(M)}} \sum_{t=1}^{T} \big(y_t - f(x_t)\big)^2 \\ + 25B^2(q+1)\log(2eB/\gamma + 1)\big(q!\gamma/(2\lambda)\big)^{-1/\beta} \\ + 60B\gamma\lceil \log_2(\gamma T/B)\rceil \sqrt{(q+1)\log(4eT+1)T\big(q!\gamma/(2\lambda)\big)^{-1/\beta}}\,.\end{aligned}$$

But, $\mathcal{F}^{(M)}$ is by Lemma 5.6 a $\gamma/2^M$-net of $\mathcal{F}$. Using that $M = \lceil \log_2(\gamma T/B)\rceil$ and following the proof of (5.15), it entails

$$\inf_{f \in \mathcal{F}^{(M)}} \sum_{t=1}^{T} \big(y_t - f(x_t)\big)^2 \leqslant \inf_{f \in \mathcal{F}} \sum_{t=1}^{T} \big(y_t - f(x_t)\big)^2 + 4B^2 + \frac{B^2}{T}\,.$$

Finally, we have

$$\begin{aligned}\sum_{t=1}^{T} \big(y_t - \widehat{f}_t(x_t)\big)^2 \leqslant \inf_{f \in \mathcal{F}} \sum_{t=1}^{T} \big(y_t - f(x_t)\big)^2 \\ + 25B^2(q+1)\log(2eB/\gamma + 1)\big(q!\gamma/(2\lambda)\big)^{-1/\beta} + 4B^2 + \frac{B^2}{T} \\ + 60B\gamma\lceil \log_2(\gamma T/B)\rceil \sqrt{(q+1)\log(4eT+1)T\big(q!\gamma/(2\lambda)\big)^{-1/\beta}}\,. \quad (5.51)\end{aligned}$$

The above regret bound grows roughly as (we omit logarithmic factors and small additive terms):

$$\gamma^{-1/\beta} + \gamma^{1-1/(2\beta)}\sqrt{T}\,.$$

Optimizing in γ would yield $\gamma \approx T^{-\beta/(2\beta+1)}$ and a regret roughly of order $\mathcal{O}\big(T^{1/(2\beta+1)}\big)$. More rigorously, taking $\gamma = BT^{-\beta/(2\beta+1)}$ and substituting it in (5.51) concludes the proof.

5.C.6. Proof of Lemma 5.8

Storage complexity. Fix a position $a \in \{1, \dots, 1/\delta_x\}$. At round $t \geqslant 1$, the Nested Chaining Algorithm for Hölder functions needs to store:

- the high-level weights $\widehat{w}_{t,a,j}$ for every $j \in \big\{1, \dots, \operatorname{card} \mathcal{P}_a^{(0)}\big\}$;
- the low-level weights $\widehat{u}_{t,a,j,k}^{(m,n)}$ for every $j \in \big\{1, \dots, \operatorname{card} \mathcal{P}_a^{(0)}\big\}$, every $m \in \{1, \dots, M\}$, every $n \in \{1, \dots, 4^m\}$, and every $k \in \big\{1, \dots, \operatorname{card} \mathcal{Q}_a^{(m,n)}\big\}$.

The complexity of the ath instance of $\mathcal{A}$ is thus of order

$$\operatorname{card} \mathcal{P}_a^{(0)} \times M \times 4^M \times \operatorname{card} \mathcal{Q}_a^{(M,n)} .$$

Now, we bound each of these terms separately. First for $\gamma = BT^{-\beta/(2\beta+1)}$, we have

$$\begin{aligned} \operatorname{card} \mathcal{P}_a^{(0)} &\leqslant (2B/\delta_y + 1)^{q+1} = (2eB/\gamma + 1)^{q+1} = \big(2eT^{\beta/(2\beta+1)} + 1\big)^{q+1} \\ &= \mathcal{O}\big(T^{\beta(q+1)/(2\beta+1)}\big) , \end{aligned}$$

because $\delta_y = e/\gamma$. Second using $M = \lceil \log_2(\gamma T/B) \rceil$, we can see that

$$4^M = \big(2^M\big)^2 \leqslant (2\gamma T/B)^2 = \big(2T^{1-\beta/(2\beta+1)}\big)^2 = \mathcal{O}\big(T^{2-2\beta/(2\beta+1)}\big) ,$$

and that

$$\operatorname{card} \mathcal{Q}_a^{(M,n)} \leqslant \big(2^{M+1} eB/\gamma + 1\big)^{2(q+1)} \leqslant (4eT + 1)^{2(q+1)} = \mathcal{O}\big(T^{2(q+1)}\big) .$$

Putting all things together the space-complexity of the ath instance of $\mathcal{A}$ is of order

$$\mathcal{O}\big(T^{2q+4+\beta(q-1)/(2\beta+1)} \log T\big) .$$

The whole storage complexity of the algorithm is thus of order

$$\mathcal{O}\big(T^{2q+4+\beta(q-1)/(2\beta+1)}/\delta_x\big) = \mathcal{O}\big(T^{2q+4+(\beta(q-1)+1)/(2\beta+1)} \log T\big) ,$$

where we used that $\delta_x = 2(q!\gamma/(2\lambda))^{1/\beta} = \mathcal{O}\big(T^{-1/(2\beta+1)}\big)$

Time complexity. At round $t \geqslant 1$, x_t only falls into one subinterval I_a and one subinterval $I_a^{(m,n)}$ for each level $m = 1, \dots, M$. It thus needs to update

- the weights $\widehat{w}_{t,a,j}$ for a single position a and for every $j \in \big\{1, \dots, \operatorname{card} \mathcal{P}_a^{(0)}\big\}$,
- for every level $m = 1, \dots, M$ the weights $\widehat{u}_{t,a,j,k}^{(m,n)}$ for a single position a and a single n, but for all $j \in \big\{1, \dots, \operatorname{card} \mathcal{P}_a^{(0)}\big\}$ and all $k \in \big\{1, \dots, \operatorname{card} \mathcal{Q}_a^{(m,n)}\big\}$.

The time-complexity is thus bounded by

$$\mathcal{O}\Big(\operatorname{card} \mathcal{P}_a^{(0)} \times M \times \operatorname{card} \mathcal{Q}_a^{(M,n)} \Big) = \mathcal{O}\big(T^{(q+1)(2+\beta/(2\beta+1))} \log T\big) .$$

6

A consistent deterministic regression tree for nonparametric prediction of time series

We study online prediction of bounded stationary ergodic processes. To do so, we consider the setting of prediction of individual sequences and propose a deterministic regression tree that performs asymptotically as well as the best L-Lipschitz predictor. Then, we show why the obtained regret bound entails the asymptotic optimality with respect to the class of bounded stationary ergodic processes.

Contents

NOTA: This chapter is a joint work with Paul Baudin. It is based on the submitted paper [6].

6.1. Introduction

Consider that at each time step $t = 1, 2, \dots$, a learner is asked to form a prediction $\widehat{Y}_t$ of the next outcome $Y_t \in [0,1]$ of a bounded stationary ergodic process $(Y_t)_{t=-\infty,\dots,\infty}$ with knowledge of the past observations $Y_1, \dots, Y_{t-1}$. To evaluate the performance, a convex and M-Lipschitz (with respect to its first argument) loss function $\ell : [0,1]^2 \to [0,1]$ is considered. But, for any bounded and continuous loss functions ℓ, Algoet [15] proved the following fundamental limit. For any prediction strategy, almost surely

$$\liminf_{T\to\infty} \left\{ \frac{1}{T} \sum_{t=1}^{T} \ell\big(\widehat{Y}_t, Y_t\big) \right\} \geqslant L^\star \,,$$

where

$$L^\star = \mathbb{E}\Big[\inf_{f \in \mathcal{B}^\infty} \mathbb{E}\Big[\ell\big(f(Y_{-\infty}^{-1}), Y_0\big) \big| Y_{-\infty}^{-1} \Big] \Big]$$

is the expected minimal loss over all possible Borel estimations of the outcome Y_0 based on the infinite past ($\mathcal{B}^\infty$ denotes the set of Borel functions from $[0,1]^\infty$ to $[0,1]$). One may thus try to design *consistent* strategies that achieve the lower bound, that is,

$$\limsup_{T\to\infty} \left\{ \frac{1}{T} \sum_{t} \ell\big(\widehat{Y}_t, Y_t\big) \right\} \leqslant L^\star \,. \tag{6.1}$$

Our approach. To do so, we partition the analysis and the design into two separate layers: the setting of individual sequences and the one of stochastic time series. In Sections 6.2 and 6.3, we adopt first the point of view of individual sequences, where no stochastic assumption about the underlying process that generates the data is made, see the monograph of Cesa-Bianchi and Lugosi [43]. Only Section 6.4 assumes that the data comes from a stationary ergodic process and states that any strategy that satisfies some deterministic regret bound is consistent. Sections 6.2, 6.3, and 6.4 can be read independently.

Formally, our framework(s) is (are) the following. The setting of individual sequences (Section 6.2) assumes that a sequence $(\boldsymbol{x}_t, y_t) \in \mathcal{X} \times \mathcal{Y}$ is observed step by step, where $\mathcal{X} \subset [0,1]^d$ is the covariate space and $\mathcal{Y} \subset [0,1]$ a convex observation space*. The learner is asked at each time step t to predict the next observation y_t with knowledge of the past observations $y_1, \dots, y_{t-1}$ and of the past and present exogenous variables $\boldsymbol{x}_1, \dots, \boldsymbol{x}_t$. The accuracy of a prediction is measured by a loss function $\ell : \mathcal{X} \times \mathcal{Y} \to [0,1]$ that we assume to be convex and M-Lipschitz in its first argument (typically the square loss, the pinball loss, ...). Given a d-dimensional input space $\mathcal{X}$, the goal of the forecaster is to minimize its cumulative regret against the classes $\mathcal{L}_L^d$ of L-Lipschitz functions with respect to the ℓ^2-norm (for all fixed $L > 0$†) from $[0,1]^d$ to $[0,1]$,

$$\widehat{R}_{L,T} = \sum_{t=1}^{T} \ell(\widehat{y}_t, y_t) - \inf_{f \in \mathcal{L}_L^d} \sum_{t=1}^{T} \ell\big(f(\boldsymbol{x}_t), y_t\big) \,,$$

*In Section 6.3, $\boldsymbol{x}_t$ will be replaced by $y_{t-d}^{t-1} = y_{t-d}, \dots, y_{t-1}$, then, the deterministic elements y_t will be replaced by random variables Y_t in Section 6.4.

†The strategy of the forecaster is independent of the Lipschitz constant L, which is not known.

that is, to ensure

$$\forall L > 0\,, \qquad \limsup_{T\to\infty} \left\{ \sup_{(\boldsymbol{x}_1,y_1),\dots,(\boldsymbol{x}_T,y_T)} \frac{\widehat{R}_{L,T}}{T} \right\} \leqslant 0\,.$$

Section 6.4 addresses actually the more challenging goal of competing against all Borel functions.

Section 6.2 designs such a strategy. The nested EG strategy (Algorithm 12) follows the spirit of binary regression trees like CART (see Breiman et al. [36]), by performing a data-driven partition of the covariate space. Theorem 6.3 guarantees that the nested EG strategy satisfies for all $T \geqslant 1$ and all $L > 0$,

$$\widehat{R}_{L,T} \leqslant M(L+3)\Big(\sqrt{T} + 2(3d)^{\frac{d}{2(d+2)}} T^{\frac{d+1}{d+2}}\Big)\,, \tag{6.2}$$

in the worst case, that is, for all possible values of $(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_T, y_T)$.

In Section 6.3, we switch to an autoregressive setting: previous responses $y_{t-d}^{t-1} = (y_{t-d}, \dots, y_{t-1})$ are used as covariates to predict y_t. Basically, $\boldsymbol{x}_t$ is replaced by y_{t-d}^{t-1}. The number d of previous responses is the order of the autoregressive model. For each order $d \geqslant 1$, Inequality (6.2) remains valid. The challenge of Section 6.3 is to obtain the result simultaneously for all orders $d \geqslant 1$. This is done (Algorithm 13) by combining an increasing number of fixed order-d nested EG. Theorem 6.6 upper-bounds the regret of Algorithm 13 for all $d \leqslant T$, and for all $y_1, \dots, y_T \in [0,1]$,

$$\sum_{t=d+1}^{T} \ell(\widehat{y}_t, y_t) \leqslant \inf_{f\in\mathcal{L}_L^d} \sum_{t=d+1}^{T} \ell\Big(f\big(y_{t-d}^{t-1}\big), y_t\Big) + \sqrt{(T+1)\log(T+1)} \\ + M(L+3)\Big(\sqrt{T} + 2(3d)^{\frac{d}{2(d+2)}} T^{\frac{d+1}{d+2}}\Big)\,.$$

Consequently, for all $d \geqslant 0$ and all $L > 0$, and for all $y_1, \dots, y_T \in [0,1]$,

$$\limsup_{T\to\infty} \left\{ \frac{1}{T} \sum_{t=d+1}^{T} \ell\big(\widehat{y}_t, y_t\big) \right\} \leqslant \limsup_{T\to\infty} \left\{ \inf_{f\in\mathcal{L}_L^d} \frac{1}{T} \sum_{t=d+1}^{T} \ell\Big(f(y_{t-d}^{t-1}), y_t\Big) \right\}\,. \tag{6.3}$$

Finally, we assume in Section 6.4 that the sequence of observations $y_1, \dots, y_T$ actually comes from a stationary ergodic process (Y_t): y_t is replaced by Y_t. Our main result is Theorem 6.7 which states that any strategy achieving (6.3) is actually consistent, i.e., satisfies (6.1).

Literature review. Many consistent forecasting strategies have been designed to achieve (6.1). The vast majority of these strategies are based on statistical techniques used for time-series prediction, ranging from parametric models like autoregressive models (see Brockwell and Davis [38]) to nonparametric methods (see the surveys of Györfi et al. [84], Bosq [30], Merhav and Feder [114]). In recent years, another collection of algorithms resolving related problems have been designed in Györfi et al. [85], Györfi and Ottucsák [83], Biau et al. [25], Biau and Patra [24]. At their cores, all these algorithms use some machine learning nonparametric prediction scheme (like histogram, kernel, or nearest neighbor estimation). These algorithms rely on two parameters: a window parameter h taking values in a discrete countable set $\mathcal{H}$ (e.g., the number of the bins of the uniform histograms or the number of neighbors), and the order d of the autoregressive model to consider. Then, they output predictions by mixing the countably infinite set of experts indexed by $(h,d) \in \mathcal{H} \times \mathbb{N}^\star$ corresponding to strategies with fixed values of these two parameters. To do so, a prior distribution π on this infinite set of experts $\mathcal{H} \times \mathbb{N}$ has to be considered. All these studies only perform asymp-

totic analysis and thus only require that the prior π assign positive weight to each expert (h, d) without taking into account the complexity of the experts. The practical choice of π is however left to the user and not calibrated online. No finite-time analysis is performed. Besides, computational purposes require to consider nothing but finite grids in numerical experiments, which results in practice in approximation of the algorithm studied theoretically. In all these studies, assumption are made from the beginning about the underlying process generating the sequence of observations. In our method, consistency only comes as an additional feature of the algorithm via Theorem 6.7 in the special case of stochastic time series.

In contrast to statistics, little research has been devoted to online nonparametric prediction in the setting of individual sequences. Recently, Rakhlin and Sridharan [124] provided the first optimal rates for the square loss for arbitrary classes of regression functions. Surprisingly, these rates match the ones for statistical learning with square loss. However, the algorithm provided is generally not computationally feasible. Vovk [143, 144] considered vast classes of regression function (such as Besov space) and proposed two approaches: defensive forecasting that uses the convexity of the functional space and perform directly a method of defensive forecasting (such as gradient descent), and another one based on aggregating a set of basis functions, that approximates well the space. However, Vovk does not explains how to build efficiently the set of basis functions to aggregate, and how to calibrate the precision of approximation. Our method nested EG can be seen as a data-driven procedure to do so in the particular case where the comparison class are Lipschitz functions. Theorem 6.7 provides the first link between this literature, and the statistical goal of getting consistent strategies that achieve (6.1).

Contributions. First, we clean up the standard analysis of prediction of ergodic processes by carrying out the aforementioned separation in two layers: first we perform a worst-case deterministic analysis, then in the particular case where data comes from stationary ergodic process, consistency is obtained as a direct consequence via Theorem 6.7. Our second main contribution is to build an efficient data-driven and fully automatic procedure (nested EG) to compete against Lipschitz functions. Our forecaster comes with robust (worst-case) finite-time guarantees. In Section 6.2.3, we consider two simulation studies which confirm that nested EG performs much better than simpler methods like uniform histograms. Our algorithm, which is purely sequential (unlike nearest neighbor estimation for instance), is computationally efficient in contrast to the one of Rakhlin and Sridharan [124]. Its time complexity can indeed be chosen arbitrarily close to linear in the number of time steps. A last benefit of our approach is to be valid for a general class of loss functions when previous papers to our knowledge only treat particular cases like the square loss or the pinball loss.

6.2. A strategy that competes against Lipschitz functions

The nested EG strategy (Algorithm 12) incrementally builds an estimate of the best Lipschitz function (denoted by $f^\star$) with respect to the ℓ^2-norm. The core idea is to estimate $f^\star$ precisely in areas of the covariate space $\mathcal{X}$ with many occurrences of covariates $\boldsymbol{x}_t$, while estimating it loosely in other parts of the space. To implement this idea, Algorithm 12 maintains a deterministic binary tree whose nodes are associated with regions of the covariate space, such that the regions with nodes deeper in the tree represent increasingly smaller subsets of $\mathcal{X}$ (see Figure 6.1).

In the sequel, we assume for simplicity that $\mathcal{X} = [0, 1]^d$ and $\mathcal{Y} = [0, 1]$ and that the loss function ℓ

is from $[0,1]^2$ to $[0,1]$. The case of unknown bounded sets $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} \subset \mathbb{R}$ will be treated later in remarks.

This section is divided into three subsections. Section 6.2.2 is the core of the algorithm. It explains how to partition the space of covariates $\boldsymbol{x}_t$ in a clever data-driven fashion. Inside each region of the space, the algorithm estimates the best constant prediction by running a defensive forecasting. Section 6.2.1 and Section 6.2.1 control the error suffered by the defensive algorithm inside each region: Section 6.2.1 bounds the approximation error of a Lipschitz function by a constant, and Section 6.2.1 controls the estimation error of learning this best constant online.

6.2.1. Performing almost as well as the best constant

Approximation error of the best constant

If the number of observations such that $\boldsymbol{x}_t$ belong to a subset $\mathcal{X}^{\text{node}} \subset \mathcal{X}$ is small enough, one does not need to estimate $f^\star$ precisely over $\mathcal{X}^{\text{node}}$. Lemma 6.1 formalizes this idea by controlling the approximation error suffered by approximating $f^\star$ by the best constant in $[0,1]$. The control is expressed in terms of the number of observations T^{node} and of the size of the set $\mathcal{X}^{\text{node}}$, which is measured by its diameter defined as $\operatorname{diam}\big(\mathcal{X}^{\text{node}}\big) = \max_{\boldsymbol{x},\boldsymbol{x}' \in \mathcal{X}^{\text{node}}} \|\boldsymbol{x} - \boldsymbol{x}'\|_2$.

Lemma 6.1. — Approximation error. *Let $T^{\text{node}} \geqslant 1$ and suppose that ℓ is M-Lipschitz in its first argument. Then, for all $(\boldsymbol{x}_1, y_1), \dots, (\boldsymbol{x}_{T^{\text{node}}}, y_{T^{\text{node}}}) \in [0,1]^2$ and all $L > 0$, the cumulative loss of the best constant is upper bounded as*

$$\inf_{y \in [0,1]} \sum_{t=1}^{T^{\text{node}}} \ell(y, y_t) \leqslant \inf_{f \in \mathcal{L}_L^d} \sum_{t=1}^{T^{\text{node}}} \ell\big(f(\boldsymbol{x}_t), y_t\big) + MLT^{\text{node}} \operatorname{diam}\big(\mathcal{X}^{\text{node}}\big),$$

where $\mathcal{X}^{\text{node}} \subset [0,1]^d$ is such that $\boldsymbol{x}_t \in \mathcal{X}^{\text{node}}$ for all $t = 1, \dots, T^{\text{node}}$.

Proof Let $t \geqslant 1$. Using that ℓ is M-Lipschitz and f is L-Lipschitz with respect to the ℓ^2 norm, we get

$$\begin{aligned} \ell\big(f(\boldsymbol{x}_1), y_t\big) - \ell\big(f(\boldsymbol{x}_t), y_t\big) &\leqslant M\big|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_t)\big| \\ &\leqslant ML\big\|\boldsymbol{x}_1 - \boldsymbol{x}_t\big\|_2 \\ &\leqslant ML \operatorname{diam}\big(\mathcal{X}^{\text{node}}\big) . \end{aligned}$$

Summing over t and noting that $\inf_y \sum_t \ell(y, y_t) \leqslant \sum_t \ell\big(f(\boldsymbol{x}_1), y_t\big)$ concludes. ∎

Estimation error of the best constant online

Lemma 6.1 implies that considering constant predictions is not bad when either the covariate region is small, or the number of observations is small. The next step consists thus in estimating online the best constant prediction in $[0,1]$.

To do so, among many existing methods, we consider the well-known *gradient-based exponentially weighted average forecaster* (EG), introduced by Kivinen and Warmuth [97]. In the setting of pre-

Algorithm 11: The gradient-based exponentially weighted average forecaster (EG) with two constant experts that predict respectively 0 and 1.

Parameter: $M > 0$

For time step $t = 1, 2, \ldots$

1. Define the learning parameter $\eta_t = M^{-1}\sqrt{(\log 2)/t}$
2. Predict
$$\widehat{y}_t = \frac{\exp\Big(-\eta_t \sum_{s=1}^{t-1} \ell'(\widehat{y}_s, y_s)\Big)}{1 + \exp\Big(-\eta_t \sum_{s=1}^{t-1} \ell'(\widehat{y}_s, y_s)\Big)} \in [0,1]\,,$$
where ℓ' denotes the (sub)gradient of ℓ with respect to its first argument
3. Observe y_t

diction of individual sequences with expert advice—see the monograph by Cesa-Bianchi and Lugosi [43], EG competes with the best fixed convex combination of experts. In the case where two experts predict constant predictions respectively 0 and 1 at all time steps, EG ensures vanishing average regret with respect to any constant prediction in $[0,1]$. Algorithm 11 implements this particular case of EG and Lemma 6.2 provides the associated regret bound. Lemma 6.2 is a particular case of the standard regret bound of EG, whose proof is available for instance in Cesa-Bianchi and Lugosi [43].

Lemma 6.2. — Estimation error. *Let $T^{\text{node}} \geqslant 1$. We assume that the loss function ℓ is convex and M-Lipschitz in its first argument. Then, for all $y_1, \ldots y_{T^{\text{node}}} \in [0,1]$, the cumulative loss of Algorithm 11 is upper bounded as follows:*

$$\sum_{t=1}^{T^{\text{node}}} \ell(\widehat{y}_t, y_t) \leqslant \inf_{y \in [0,1]} \sum_{t=1}^{T^{\text{node}}} \ell(y, y_t) + 2M\sqrt{T^{\text{node}} \log 2}\,.$$

Unknown value of M. Note that Algorithm 11 needs to know in advance a uniform bound M on ℓ'. This is the case if one considers (as we do) a bounded observation space $[0,1]$ with the absolute loss function, defined for all $y, y' \in [0,1]$ by $\ell(y', y) = |y - y'|$; the pinball loss, defined by $\ell_\alpha(y', y) = (\alpha - \mathbb{1}_{\{y \geqslant x\}})(y - y')$; or the square loss, defined by $\ell(y', y) = (y - y')^2$. However, in the case of an unknown observation space $\mathcal{Y}$ the bound on the gradient of the square loss is unknown and needs to be calibrated online at the small cost of the additional term $2M(2 + 4(\log 2)/3)$ in the regret bound, see de Rooij et al. [58].

6.2.2. Performing almost as well as the best Lipschitz function: the nested EG strategy

The nested EG strategy (Algorithm 12) implements the idea of Lemma 6.1 and Lemma 6.2. It maintains a binary tree whose nodes are associated with regions of the covariate space $[0,1]^d$. The nodes in the tree are indexed by pairs of integers (h, i); where the first index $h \geqslant 0$ denotes the distance of the node to the root (also referred to as the depth of the node) and the second index i belongs to $\{1, \ldots, 2^h\}$. The root is thus denoted by $(0, 1)$. By convention, $(h+1, 2i-1)$ and

$(h+1, 2i)$ are used to refer to the two children of node (h, i). Let $\mathcal{X}^{(h,i)} \subset [0,1]^d$ be the region associated with node (h, i). By construction, these regions are hyper-rectangles and satisfy the constraints

$$\mathcal{X}^{(0,1)} = [0,1]^d \qquad \text{and} \qquad \mathcal{X}^{(h,i)} = \mathcal{X}^{(h+1,2i-1)} \sqcup \mathcal{X}^{(h+1,2i)} ,$$

where $\sqcup$ denotes the disjoint union. The set of regions associated with terminal nodes (or leaves) thus forms a partition of $[0,1]^d$.

At time step t, when a new covariate $\boldsymbol{x}_t$ is observed, Algorithm 12 first selects the associated leaf (h_t, i_t) such that $\boldsymbol{x}_t \in \mathcal{X}^{(h_t,i_t)}$ (Step 2). The leaf (h_t, i_t) then predicts the next observation y_t by updating a local version of Algorithm 11 (Step 3). Namely, for node (h_t, i_t), Algorithm 12 runs Algorithm 11 on the sub-sequence of past observations $(\boldsymbol{x}_s, y_s)$ such that the associated leaf is (h_t, i_t), that is, $E^{(h_t,i_t)} \stackrel{\text{def}}{=} \{1 \leqslant s \leqslant t-1 : (h_s, i_s) = (h_t, i_t)\}$; then it forms the prediction (Step 3) and finally updates the set of observations predicted by node (h_t, i_t): $E^{(h_t,i_t)} = E^{(h_t,i_t)} \cup \{t\}$ (Step 4). When the number of observations $T^{(h_t,i_t)} \stackrel{\text{def}}{=} \#E^{(h_t,i_t)}$ received and predicted by leaf (h_t, i_t) becomes too large compared to the size of the region $\mathcal{X}^{(h_t,i_t)}$ (Step 5), the tree is updated. To do so, the region $\mathcal{X}^{(h_t,i_t)}$ is divided in two sub-regions of equal volume by cutting along one given coordinate.

The coordinate $r_t + 1$ to be split is chosen in a deterministic order, where $r_t = (h_t \mod d)$ and mod denotes the modulo operation. Thus, at the root node $(0, 1)$ the first coordinate is split, then by going down in the tree we split the second one, then the third one and so on until we reach the depth d, in which case we split the first coordinate for the second time. Each sub-region is associated with a child of node (h_t, i_t). Consequently, (h_t, i_t) becomes an inner node and is thus no longer used to form predictions.

To facilitate the formal study of the algorithm, we will need some additional notation. In particular, we will introduce time-indexed versions of several quantities. $\mathcal{T}_t$ denotes the tree stored by Algorithm 12 at the beginning of time step t. The initial tree is thus the root $\mathcal{T}_0 = \{(0, 1)\}$ and it is expanded when the splitting condition (Step 5) holds, as

$$\mathcal{T}_{t+1} = \mathcal{T}_t \cup \big\{(h_t + 1, 2i_t - 1), (h_t + 1, 2i_t)\big\}$$

(Step 5.2.3) and remains unchanged otherwise. We denote by N_t the number of nodes of $\mathcal{T}_t$ and by H_t the height of $\mathcal{T}_t$, that is, the maximal depth of the leaves of $\mathcal{T}_t$. A performance bound for Algorithm 12 is provided below.

Theorem 6.3. *Let $T \geqslant 1$ and $d \geqslant 1$. Then, the cumulative regret $\widehat{R}_{L,T}$ of Algorithm 12 is*

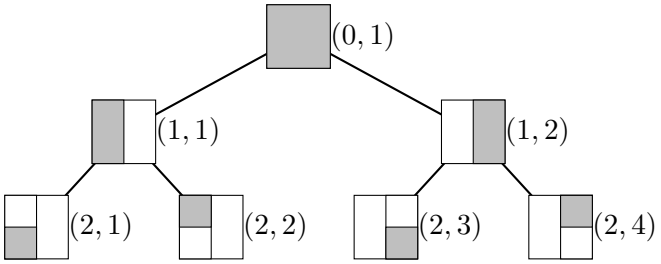


Figure 6.1.: Representation of the binary tree in dimension $d = 2$. In this case, the regions associated to nodes $(0, 1)$ is $\mathcal{X}^{(0,1)} = [0,1]^2$ and to node $(2, 3)$ is $\mathcal{X}^{(2,3)} = [0.5, 1] \times [0, 0.5]$.

Algorithm 12: Sequential prediction of function via Nested EG

Initialization:

- $\mathcal{T} = \big\{(0,1)\big\}$ a tree (for now reduced at a root node)
- Define the bin $\mathcal{X}^{(0,1)} = [0,1]^d$ and the corresponding set of points $E^{(0,1)} = \emptyset$.

For $t = 1, \dots, T$

1. Observe $\boldsymbol{x}_t \in [0,1]^d$
2. Select the leaf (h_t, i_t) such that $\boldsymbol{x}_t \in \mathcal{X}^{(h_t,i_t)}$
3. Predict
$$\widehat{y}_t = \frac{\exp\big(-\eta^{(h_t,i_t)} \sum_{s \in E^{(h_t,i_t)}} \ell'(\widehat{y}_s, y_s)\big)}{1 + \exp\big(-\eta^{(h_t,i_t)} \sum_{s \in E^{(h_t,i_t)}} \ell'(\widehat{y}_s, y_s)\big)} \in [0,1]\,,$$
where $\eta^{(h_t,i_t)} = \sqrt{(\log 2)/\big(\#E^{(h_t,i_t)} + 1\big)}$.
4. Update the set of observations predicted by node (h_t, i_t)
$$E^{(h_t,i_t)} \leftarrow \big\{1 \leqslant s \leqslant t, \quad (h_s, i_s) = (h_t, i_t)\big\}$$
5. **If** the splitting condition $\#E^{(h_t,i_t)} + 1 \geqslant \Big(\operatorname{diam}\big(\mathcal{X}^{(h_t,i_t)}\big)\Big)^{-2}$ holds **then** extend the binary tree $\mathcal{T}$ as follows:
   5.1. Compute the decomposition $h_t = k_t d + r_t$ with $r_t \in \{0, \dots, d-1\}$
   5.2. Split coordinate $r_t + 1$ for node (h_t, i_t)
      5.2.1. Define the splitting threshold $\tau = \big(x^- + x^+\big)/2$, where $x^- = \inf_{\boldsymbol{x} \in \mathcal{X}^{(h_t,i_t)}} \{x_{r_t+1}\}$ and $x^+ = \sup_{\boldsymbol{x} \in \mathcal{X}^{(h_t,i_t)}} \{x_{r_t+1}\}$.
      5.2.2. Define two children leaves for node (h_t, i_t):
      - the left leaf $(h_t + 1, 2i_t - 1)$ with corresponding bin
$$\mathcal{X}^{(h_t+1,2i_t-1)} = \{\boldsymbol{x} \in \mathcal{X}^{(h_t,i_t)} \,:\, x_{r_t+1} \in [x^-, \tau[\}$$
      - the right leaf $(h_t + 1, 2i_t)$ with corresponding bin
$$\mathcal{X}^{(h_t+1,2i_t-1)} = \left\{\boldsymbol{x} \in \mathcal{X}^{(h_t,i_t)} \,:\, \begin{array}{ll} x_{r_t+1} \in [\tau, x^+[& \text{if } x_+ < 1 \\ x_{r_t+1} \in [\tau, 1] & \text{if } x_+ = 1 \end{array}\right\}$$
      5.2.3. Initialize their sets of observations
$$E^{(h_t+1,2i_t-1)} = E^{(h_t+1,2i_t-1)} = \emptyset.$$
      5.2.3. Update $\mathcal{T} \leftarrow \mathcal{T} \cup \big\{(h_t + 1, 2i_t - 1), (h_t + 1, 2i_t)\big\}$

upper bounded as

$$\sum_{t=1}^{T} \ell(\widehat{y}_t, y_t) - \inf_{f \in \mathcal{L}_L^d} \sum_{t=1}^{T} \ell\big(f(\boldsymbol{x}_t), y_t\big) \leqslant M\,(L+3)\,\sqrt{N_T T}$$
$$\leqslant M(L+3)\Big(\sqrt{T} + 2(3d)^{\frac{d}{2(d+2)}} T^{\frac{d+1}{d+2}}\Big)\,.$$

The proof of Theorem 6.3 is deferred to Section 6.A.1.2.

Time and storage complexity. The following lemma, whose proof is postponed to Section 6.A.1.1, provides time and storage complexity guarantees for Algorithm 12. It upper bounds the maximal size of $\mathcal{T}_T$, that is, its number of nodes N_T and its depth H_T, which yields in particular the regret

bound of order $\mathcal{O}\big(T^{(d+1)/(d+2)}\big)$ stated in Theorem 6.3.

Lemma 6.4. *Let $T \geqslant 1$ and $d \geqslant 1$. Then the depth H_T and the number of nodes N_T of the binary tree $\mathcal{T}_T$ stored by Algorithm 12 after T time steps are upper bounded as follows:*

$$H_T \leqslant 1 + \frac{d}{2}\log_2(4dT) \qquad \text{and} \qquad N_T \leqslant 1 + 8\,(dT)^{\frac{d}{d+2}}\,.$$

Indeed, Algorithm 12 needs to store a constant number of parameters at each node of the tree. Thus the space complexity is of order $\mathcal{O}(N_T) = \mathcal{O}\big(T^{d/(d+2)}\big)$. Besides at each time step t, Algorithm 12 needs to perform $\mathcal{O}(H_t) = \mathcal{O}(\log t)$ binary test operations in order to select the leaf (h_t, i_t). It then only needs constant time to update both the local version of Algorithm 11 associated with node (h_t, i_t) and the tree $\mathcal{T}$. Thus the per-round time complexity of Algorithm 12 is of order $\mathcal{O}(\log t)$ and the global time complexity is of order $\mathcal{O}(T \log T)$. Therefore, we can summarize:

$$\text{Storage complexity:} \quad O\big(T^{d/(d+2)}\big)\,, \qquad \text{Time complexity:} \quad O\big(T \log T\big)\,.$$

Unknown bounded sets $\mathcal{X} \subset \mathbb{R}^d$ and $\mathcal{Y} \subset \mathbb{R}$. As we mentioned in the end of Section 6.2.1, the generalization of Algorithm 11 and thus of Algorithm 12 to an unknown set $\mathcal{Y} \subset \mathbb{R}$ can be obtained by using standard tools of individual sequences—see for instance de Rooij et al. [58]. To adapt Algorithm 12 to any unknown compact set $\mathcal{X} \subset \mathbb{R}^d$, one can first divide the covariate space $\mathbb{R}^d$ in hyper-rectangle subregions of the form $[n_1, n_1 + 1] \times \cdots \times [n_d, n_d + 1]$ and then run independent versions of Algorithm 12 on all of these subregions. If $\operatorname{diam}(\mathcal{X}) \leqslant \sqrt{d}B$ with an unknown value of $B > 0$, then the number of initial subregions is upper-bounded by $\lceil B \rceil^d$ and by Jensen's inequality, this adaptation would lead to a multiplicative cost of $\lceil B \rceil^{d/(d+2)}$ in the upper-bound of Theorem 6.3.

Comparison with other methods. One may want to obtain similar guarantees by considering other strategies like uniform histograms, kernel regression, or nearest neighbors, which were studied in the context of stationary ergodic processes by Györfi et al. [85], Györfi and Ottucsák [83], Biau et al. [25], Biau and Patra [24]. We were unfortunately unable to provide any finite-time and worst-case analysis neither for kernel regression nor for nearest neighbors estimation.

The regret bound of Theorem 6.3 can however be obtained in an easier manner with uniform histograms. To do so, one can consider the class of uniform histograms $\mathcal{H}_N$. We divide the covariate space $[0,1]^d$ in a partition $(I_j)_{j=1,\dots,N}$ of N hyper-rectangle subregions of equal size. $\mathcal{H}_N$ is the set of functions that takes constant values in $[0,1]$ in each subregion I_j. We consider the class of 2^N prediction strategies that predict the constant values 0 or 1 in each bin of the partition. Competing with this class of 2^N functions by resorting for instance to EG gives the regret bound

$$\sum_{t=1}^{T} \ell\big(\widehat{y}_t, y_t\big) \leqslant \min_{h \in \mathcal{H}_N} \sum_{t=1}^{T} \ell\Big(h(\boldsymbol{x}_t), y_t\Big) + 2M\sqrt{TN\log 2}\,.$$

Now, considering the approximation of $\mathcal{H}_N$ to capture $\mathcal{L}_L^d$ and optimizing the number N of bins in hindsight setting $N \approx (dT)^{\frac{d}{d+2}}$ (or by resorting to the doubling trick, see Cesa-Bianchi and Lugosi [43]) provides a regret bound similar to the one of Theorem 6.3 of order $d^{d/(4d+4)}T^{(d+1)/(d+2)}$ against any Lipschitz function. The details are provided in the appendix.

In the worst case the nested EG strategy provides no better guarantee. Such worst case occurs for large number N_T of nodes, which happens in particular when the trees are height-balanced, that is, when the covariates $\boldsymbol{x}_t$ are uniformly distributed in $[0,1]^d$. But the nested EG strategy adapts better to data as it is depicted in the simulation studies below. If the covariates $\boldsymbol{x}_t$ are non-uniformly allocated (with regions of the space $[0,1]^d$ associated with much more observations than in other regions of similar size), the resulting tree $\mathcal{T}_T$ will be un-balanced, leading to a smaller number of nodes. In the best case, $N_T = \mathcal{O}(H_T)$, which yields a regret of order $\mathcal{O}(\sqrt{T \log T})$.

By improving the definition of Algorithm 12, one can even obtain the optimal and expected $\mathcal{O}(\sqrt{T})$ regret if $(\boldsymbol{x}_t)$ is constant. To do so, it only needs to compute online the effective range of the data that belongs to each node (h,i),

$$\delta_t^{(h,i)} = \operatorname{diam}\{\boldsymbol{x}_s, \quad 0 \leqslant s \leqslant t \text{ and } (h_s, i_s) = (h,i)\}$$

and substitute the diameter $\operatorname{diam} \mathcal{X}^{(h,i)}$ by $\delta_{t+1}^{(h,i)}$ in the splitting condition of the algorithm (Step 5).

6.2.3. Simulation studies

We consider in this section two simulated data sets so as to compare the performance obtained by three procedures:

- histEG corresponds to run Algorithm 11 on uniform histograms, whose size is theoretically calibrated by doubling trick with approximatively $T^{1/3}$ bins (this follows from the theoretical tuning of the previous section with $d = 1$);
- nestedEG is Algorithm 12;
- nestedEG(+) corresponds to Algorithm 12 where the splitting condition (Step 5) is replaced by the condition:

$$2 \min_{c \in [0,1]} \left\{ \sum_{s \in E^{(h_t,i_t)}} (y_s - c)^2 \right\} > \sum_{s \in E^{(h_t,i_t)}} (y_s - \widehat{y}_s)^2$$

  which can be rephrased as: "the approximation error of the best constant in the local region is larger than the estimation error". This condition is better than the one suggested in Algorithm 12 since it adapts not only to the structure of covariates $(\boldsymbol{x}_t)$ but also to the structure of the objective variable (y_t). In particular, it adapts to easy areas of the space where the link function g between $\boldsymbol{x}_t$ and y_t does not vary much. We conjecture that the theoretical guarantees can be retrieved, the proof is however left for future research.

We performed two simulations studies. In the first one, the data are independent and identically distributed, while in the second one the data are distributed from an Hidden Markov Model (HMM), see the monograph of Cappé et al. [40]. In both studies, we sampled sequences $\{(X_t, Y_t)\}_{t=1,\dots,1000}$ of $T = 1000$ observations.

Experiment 1: I.I.D. data. (X_t) is independent and identically distributed from a mixture of two Gaussian distribution restricted to $[0,1]$. Its density is proportional to

$$\frac{1}{2}\Big(\mathcal{N}_{(0,0.1)}(t) + \mathcal{N}_{(0.7,0.1)}(t)\Big)\, \mathbb{1}_{t \in [0,1]}\,,$$

where $\mathcal{N}_{(\mu,\sigma)}$ is the density function of the normal distribution of mean μ and standard deviation σ. (Y_t) is independent and identically distributed and follows the normal distribution $\mathcal{N}\big(g(X_t), 0.1\big)$ restricted to $[0,1]$, where $g(x) = (\cos(10x) + \sin(15x) + \cos(20x) + \sin(25x) + \cos(30x) + 2)/6$. The data are represented in Figure 6.3. The choices of the distribution of X and Y are quite arbitrary. The goal was to obtain X not uniformly distributed over $[0,1]$ and to have a function g with large variations in some areas and small variations in others.

Experiment 2: HMM. (X_t, Y_t) follows a 2-states HMM with transition probabilities $a_{11} = a_{22} = 0.9$ and $a_{12} = a_{21} = 0.1$ and uniform initial distribution. For each $t \geqslant 1$, $X_t \sim \text{Beta}(2,5)$ for state 1 and $X_t \sim \text{Beta}(0.7, 0.3)$ for state 2. Besides, for state $i \in \{1,2\}$, $Y_t \sim \mathcal{N}\big(g_i(X_t), 0.1\big)$ restricted to $[0,1]$, where $g_1(x) = x$ and $g_2(x) = \big|\cos(2\pi x)\big|$.

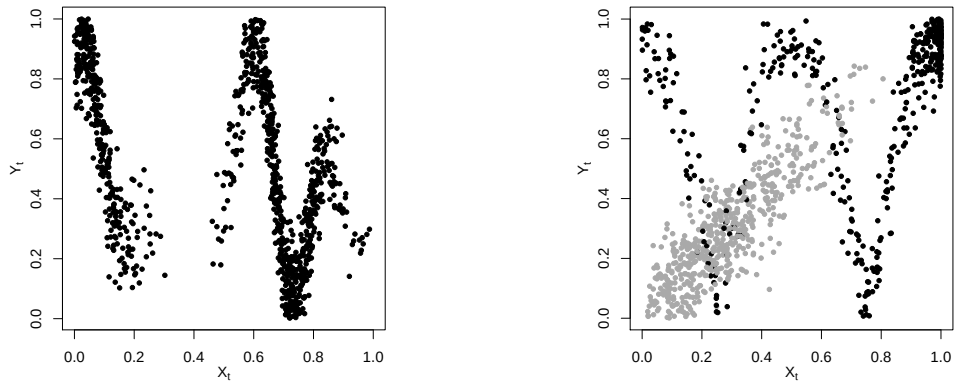


Figure 6.2.: Plots of the observations (X_t, Y_t) of Experiment 1 [left] and Experiment 2 [right]. In Experiment 2, gray points corresponds to state 1 and black points to state 2.

The data are depicted in Figure 6.2. Figure 6.3 displays the boxplots (over 100 independent runs of the experiments) of the root mean square errors (RMSEs) obtained by the three forecasting procedures. We observe much better performance of the data-driven calibration procedures in comparison to uniform histograms with theoretical calibration of the number of bins. We could observe that in area with large variations of the link function and many data, the bins of nested EG are significantly smaller than in areas with few points and low variation of the link function g. Our method however suffers from the non-smoothness of histogram procedures. The extension of our deterministic analysis to smoother strategies (such as nearest neighbors, or kernel regression) is left for future research.

6.3. Autoregressive framework

We present in this section a technical result that will be useful for later purposes. Here, the forecaster still sequentially observes from time $t = 1$ an arbitrary bounded sequence $(y_t)_{t=-\infty,\dots,+\infty}$. However, at time step t, it is asked to forecast the next outcome $y_t \in [0,1]$ with knowledge of the past observations $y_1^{t-1} = y_1, \dots, y_{t-1}$ only.

We are interested in a strategy that performs asymptotically as well as the best model that considers the last d observations to form the predictions, and does this simultaneously for all values of $d \geqslant 1$.

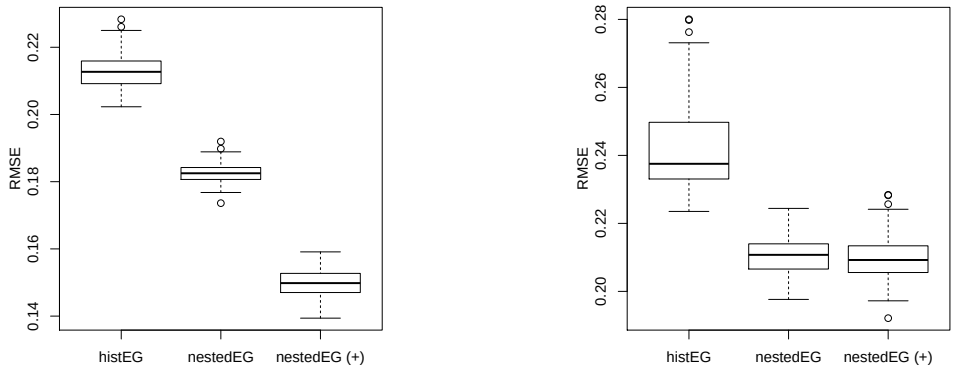


Figure 6.3.: Boxplots of the RMSE*s* suffered by the three forecasting procedures over 100 independent replications of Experiment 1 [left] and Experiment 2 [right].

More formally, we denote

$$\widehat{R}_{L,T}^{d} \stackrel{\text{def}}{=} \sum_{t=d+1}^{T} \ell(\widehat{y}_t, y_t) - \inf_{f \in \mathcal{L}_L^d} \sum_{t=d+1}^{T} \ell\big(f(y_{t-d}^{t-1}), y_t\big)\,,$$

and we want that for all d, the average regrets $\widehat{R}_{L,T}^{d}/T$ vanish as $T \to \infty$. We show how it can be obtained via a meta-algorithm (Algorithm 13) that combines an increasing sequence of orders d of autoregressive nested EG forecasters.

Fixed order *d*. For each order $d \geqslant 0$, let $\mathcal{A}_d$ denote the autoregressive forecaster of order d that forms prediction for $t \geqslant d+1$ by performing nested EG (Algorithm 12) on the sequence $\big\{\big(y_t, y_{t-d}^{t-1}\big)\big\}_{t \geqslant d+1}$. We denote by $f_{d,t}$ the prediction provided by $\mathcal{A}_d$ at time step $t \geqslant d+1$. By substituting $\boldsymbol{x}_t = y_{t-d}^{t-1}$ in Theorem 6.3, $\mathcal{A}_d$ satisfies the following regret bound

$$\begin{aligned}\sum_{t=d+1}^{T} \ell(f_{d,t}, y_t) - \inf_{f \in \mathcal{L}_L^d} \sum_{t=d+1}^{T} \ell\big(f(y_{t-d}^{t-1}), y_t\big) \\ \leqslant M(L+3)\Big(\sqrt{T} + 2(3d)^{\frac{d}{2(d+2)}} T^{\frac{d+1}{d+2}}\Big)\,,\end{aligned} \tag{6.4}$$

which is valid for all $T \geqslant 1$, all $L > 0$ and for all $y_1, \ldots, y_T \in [0,1]$.

All orders *d*. Now, we show how to obtain the above regret bound simultaneously for all orders $d \geqslant 1$. To do so, Algorithm 13 combines via EG the predictions formed by all forecasters $\mathcal{A}_d$ for $d \geqslant 0$. Note that at time step t, only the t first forecasters $\mathcal{A}_0, \ldots, \mathcal{A}_{t-1}$ suggest predictions. Lemma 6.5 controls the cumulative loss of Algorithm 13 by the cumulative loss of the best strategy $\mathcal{A}_d$.

Lemma 6.5. *Let $T \geqslant 1$. Then, Algorithm 13 satisfies for all $d \in 1, \ldots, T$, for all $L > 0$, and for all $y_1, \ldots, y_T \in [0,1]$,*

$$\sum_{t=d+1}^{T} \ell(\widehat{y}_t, y_t) - \ell(f_{d,t}, y_t) \leqslant \sqrt{(T+1)\log(T+1)}\,.$$

Algorithm 13: Extension of the Algorithm 12 to unknown order d of autoregressive model.

Parameter:
- $(\mathcal{A}_d)_{d\geqslant 1}$ a sequence of forecasters such that $\mathcal{A}_d$ forms predictions for time steps $t \geqslant d+1$

For $t = 1, \dots, T$
1. **For** each $d = 0, \dots, t-1$, denote by $f_{d,t}$ the prediction formed by $\mathcal{A}_d$
2. predict $\widehat{y}_t = \sum_{d=0}^{t-1} \widehat{p}_{d,t} f_{d,t}$
3. initialize the weight of the new forecaster: $p_{t,t+1} = 1/(t+1)$
4. observe Y_t and perform exponential weight update component-wise for $d = 0, \dots, t-1$ as

$$\widehat{p}_{d,t+1} = \frac{t}{t+1} \, \frac{\widehat{p}_{d,t}^{\eta_{t+1}/\eta_t} e^{-\eta_{t+1}\ell(f_{d,t},y_t)}}{\sum_{k=1}^{t} \widehat{p}_{k,t}^{\eta_{t+1}/\eta_t} e^{-\eta_{t+1}\ell(f_{k,t},y_t)}},$$

where $\eta_t = 2\sqrt{(\log t)/t}$ for all $t \geqslant 1$.

The proof of Lemma 6.5 follows the standard one of the exponentially weighted average forecaster. It is postponed to Section 6.A.2. It could also be recovered by noting that our setting with starting experts is a particular case of the setting of sleeping experts introduced in Freund et al. [71].

Theorem 6.6. *Let $T \geqslant 1$, $L > 0$. Then, for all $d \in \{0, \dots, T-1\}$, Algorithm 13 run with the sequence of forecasters $(\mathcal{A}_d)$ satisfies for all $L > 0$ and for all $y_1, \dots, y_T \in [0,1]$,*

$$\begin{aligned}\widehat{R}_{L,T}^d &= \sum_{t=d+1}^{T} \ell(\widehat{y}_t, y_t) - \inf_{f \in \mathcal{L}_L^d} \sum_{t=d+1}^{T} \ell(f(y_{t-d}^{t-1}), y_t) \\ &\leqslant \sqrt{(T+1)\log(T+1)} + M(L+3)\left(\sqrt{T} + 2(3d)^{\frac{d}{2(d+2)}} T^{\frac{d+1}{d+2}}\right).\end{aligned}$$

Consequently, for all $d \geqslant 1$, $\limsup_{T\to\infty} \left(\widehat{R}_{L,T}^d / T\right) \leqslant 0$.

Proof The regret bound is by combining (6.4) and Lemma 6.5. The second part is obtained by dividing by T and letting T grow to infinity. ∎

6.4. From individual sequences to ergodic processes: convergence to $L^\star$

In this section, we present our main result by deriving from Theorem 6.6 similar results obtained in a stochastic setting by Györfi et al. [85], Györfi and Ottucsák [83], Biau et al. [25], Biau and Patra [24].

We leave here the setting of individual sequences of the previous sections and we assume that the sequence of observations $y_1, \dots, y_T$ is now generated by some stationary ergodic process. More formally, we assume that a stationary bounded ergodic process $(Y_t)_{t=-\infty,\dots,\infty}$ is sequentially observed. At time step t, the learner is asked to form a prediction $\widehat{Y}_t$ of the next outcome $Y_t \in [0,1]$ of the sequence with knowledge of the past observations $Y_1^{t-1} = Y_1, \dots, Y_{t-1}$. The nested EG strategy, as

a consequence of the deterministic regret bound of Theorem 6.3, will be shown to be consistent, i.e., satisfies Equation (6.1).

Theorem 6.7 shows that any strategy that achieves a deterministic regret bound for individual sequences as in Theorem 6.6 predicts asymptotically as well as the best strategy defined by a Borel function.

Theorem 6.7. *Let $(Y_t)_{t=-\infty,\dots,\infty}$ be a stationary bounded ergodic process. We assume that for all t, $Y_t \in [0,1]$ almost surely and that for all $d \geqslant 1$ the law of $Y_{-d}^{-1} = (Y_{-d}, \dots, Y_{-1})$ is regular. Let $\ell : [0,1]^2 \to [0,1]$ be a loss function M-Lipschitz in its first argument. Assume that a prediction strategy satisfies for all $d \geqslant 1$, almost surely,*

$$\forall L \geqslant 0 \qquad \limsup_{T\to\infty} \left(\frac{1}{T} \sum_{t=1}^{T} \ell\Big(\widehat{Y}_t, Y_t\Big) \right) \leqslant \limsup_{T\to\infty} \left(\inf_{f \in \mathcal{L}_L^d} \frac{1}{T} \sum_{t=1}^{T} \ell\Big(f(Y_{t-d}^{t-1}), Y_t\Big) \right) ,$$

then, almost surely,

$$\limsup_{T\to\infty} \left(\frac{1}{T} \sum_{t=1}^{T} \ell\Big(\widehat{Y}_t, Y_t\Big) \right) \leqslant L^\star .$$

By Theorem 6.6, Algorithm 13 satisfies the assumption of Theorem 6.7 (by replacing the deterministic terms y_t by the random variables Y_t). Our deterministic strategy is thus asymptotically optimal for any stationary bounded ergodic process satisfying the assumptions of Theorem 6.7. Here we only give the main ideas in the proof of Theorem 3. The complete argument is given in Section 6.A.3.2.

Proof (sketch) Basically, the proof of Theorem 6.7 consists first in applying Breiman's generalized ergodic theorem[‡] (see Breiman [33]), so that

$$\limsup_{T\to\infty} \left(\frac{1}{T} \sum_{t=1}^{T} \ell\Big(f(Y_{t-d}^{t-1}), Y_t\Big) \right) = \mathbb{E}\Big[\ell\big(f(Y_{-d}^{-1}), Y_0\big)\Big] .$$

Then, by exchanging lim sup and inf in the right-term of the assumption and by letting $L \to \infty$, we can compete against any Lipschitz function:

$$\limsup_{T\to\infty} \left(\frac{1}{T} \sum_{t=1}^{T} \ell\Big(\widehat{Y}_t, Y_t\Big) \right) \leqslant \inf_{f \in \mathcal{L}_L^d} \mathbb{E}\Big[\ell\big(f(Y_{-d}^{-1}), Y_0\big)\Big] .$$

The proof is then completed by approximating the best Borel function by the best Lipschitz function (see Lemma 6.8 below). ∎

Lemma 6.8. *Let $\mathcal{X}$ be a convex and compact subset of a normed space. Let $\ell : [0,1]^2 \to [0,1]$ be a loss function M-Lipschitz in its first argument. Let X be a random variable on $\mathcal{X}$ with a regular law $\mathbb{P}_X$ and let Y be a random variable on $[0,1]$. Then,*

$$\inf_{f \in \mathcal{L}^{\mathcal{X}}} \mathbb{E}\big[\ell\big(f(X), Y\big)\big] = \inf_{f \in \mathcal{B}^{\mathcal{X}}} \mathbb{E}\big[\ell\big(f(X), Y\big)\big] ,$$

[‡]Here, we use the assumption that (Y_t) is a stationary ergodic process.

where $\mathcal{L}^{\mathcal{X}}$ denotes the set of Lipschitz functions from $\mathcal{X}$ to $\mathbb{R}$ and $\mathcal{B}^{\mathcal{X}}$ the one of Borel functions from $\mathcal{X}$ to $\mathbb{R}$.

The proof of Lemma 6.8 is postponed to Section 6.A.3.1. It follows from the Stone-Weierstrass theorem, used to approximate continuous functions, and from Lusin's theorem, to approximate Borel functions.

The assumptions. Theorem 6.7 makes two main assumptions on the ergodic sequence to be predicted. First, the sequence is supposed to lie in $[0,1]$. As earlier, this assumption can be easily relaxed to any bounded subset of $\mathbb{R}$—see remarks of Sections 6.2.1 and 6.2.2. The generalization to unbounded sequence is left to future work and should follow from the same techniques as in Györfi and Ottucsák [83]. Second, Theorem 6.7 assumes that for all $d \geqslant 1$ the law of Y_{-d}^{-1} is regular, that is, for any Borel set $S \subset [0,1]^d$ and for any $\varepsilon > 0$, one can find a compact set K and an open set V such that

$$K \subset S \subset V, \qquad \text{and} \qquad \mathbb{P}_{Y_{-d}^{-1}}(V \backslash K) \leqslant \varepsilon \,.$$

This second assumption is considerably weaker than the assumptions required by Biau and Patra [24] for quantile prediction. The authors indeed imposed that the random variables $\|Y_{-d}^{-1} - s\|$ have continuous distribution functions for all $s \in \mathbb{R}^d$ and the conditional distribution function $F_{Y_0|Y_{-\infty}^{-1}}$ to be increasing. One can however argue that their assumptions are thus hardly comparable with ours because they consider unbounded ergodic processes. We aim at obtaining in the future minimal assumptions for any generic convex loss function ℓ in the case of unbounded ergodic process, see Morvai and Weiss [116].

Computational efficiency. The space complexity of Algorithm 13 is $\mathcal{O}\big(T^2\big)$. Previous algorithms of Györfi et al. [85], Györfi and Ottucsák [83], Biau et al. [25], Biau and Patra [24] exhibit consistent strategies as well. However, in practice, these algorithms involve choices of parameters somewhere in their design (by choosing the a priori weight of the infinite set of experts). Then, the consideration of an infinite set of experts makes the exact algorithm computationally inefficient. For practical purpose, it needs to be approximated. This can be obtained by MCMC or for instance by restricting the set of experts to some finite subset at the cost, however, of loosing theoretical guarantees, see Biau and Patra [24].

Generic loss function. Theorem 6.7 assumes ℓ to be bounded, convex, and M-Lipschitz in its first argument. In contrast, the results of Györfi et al. [85], Györfi and Ottucsák [83], Biau et al. [25], Rakhlin and Sridharan [124] only hold for the square loss (while Biau and Patra [24] extend them to the pinball-loss).

Appendices for Chapter 6

6.A. Technical proofs

We gather in this section the proofs, which were omitted from the previous sections.

6.A.1. Proofs of Section 2

The proofs of Theorem 6.3 and Lemma 6.4 are based on the following lemma, which controls the size of the regions associated with nodes located at depth h in the tree $\mathcal{T}_T$.

Lemma 6.9. *Let $h \geqslant 0$. Then, for all indices $i = 1, \dots, 2^h$, the diameter of the region $\mathcal{X}^{(h,i)}$ associated with node (h,i) in Algorithm 12 is upper bounded as*

$$\operatorname{diam}\left(\mathcal{X}^{(h,i)}\right) \leqslant \sqrt{2d}2^{-h/d}\,.$$

Proof Basically, the proof of Lemma 6.9 consists of an induction on the depth h. It suffices to prove that for all $h \geqslant 0$, for all indexes $i \in \{1, \dots, 2^h\}$ and all coordinates $j \in \{1, \dots, d\}$, the ranges $\delta_j^{(h,i)} \stackrel{\text{def}}{=} \max_{\boldsymbol{x},\boldsymbol{x}' \in \mathcal{X}^{(h,i)}} \left|x_j - x'_j\right|$ satisfies

$$\delta_j^{(h,i)} = \begin{cases} 2^{-(k+1)} & \text{if } j \leqslant r \\ 2^{-k} & \text{otherwise} \end{cases}\,, \tag{6.5}$$

where $h = kd + r$ is the decomposition with $r \in \{0, \dots, d-1\}$. Indeed, we then have

$$\operatorname{diam}\left(\mathcal{X}^{(h,i)}\right) = \max_{\boldsymbol{x},\boldsymbol{x}' \in \mathcal{X}^{(h,i)}} \left\|\boldsymbol{x} - \boldsymbol{x}'\right\|_2 \leqslant \sqrt{\sum_{j=1}^{d} \left(\delta_j^{(h,i)}\right)^2}\,.$$

But by (6.5), for r coordinates $j \in \{1, \dots, r\}$ among the d coordinates $\delta_j^{(h,i)}$ equals $2^{-(k+1)}$ while the $d-r$ remaining coordinates $j \in \{r+1, \dots, d\}$ satisfy $\delta_j^{(h,i)} = 2^{-k}$. Thus, by routine calculations

$$\begin{aligned} \operatorname{diam}\left(\mathcal{X}^{(h,i)}\right) &\leqslant \sqrt{r\left(2^{-(k+1)}\right)^2 + (d-r)\left(2^{-k}\right)^2} \\ &= 2^{-k}\sqrt{\frac{r}{4} + d - r} \\ &= \sqrt{d}2^{-k}\sqrt{1 - \frac{3r}{4d}} \\ &= \sqrt{d}\left(2^{1/d}\right)^{-(dk+r)} 2^{r/d}\sqrt{1 - \frac{3r}{4d}}\,. \end{aligned}$$

But,

$$2^{r/d}\sqrt{1-\frac{3r}{4d}} \leqslant \max_{0\leqslant u\leqslant 1}\left\{2^u\sqrt{1-\frac{3u}{4}}\right\} \approx 1.12 \leqslant \sqrt{2}\,.$$

The proof is concluded by substituting in the previous bound.

Now, we prove (6.5) by induction on the depth h. This is true for $h = 0$ as the bin of the root node $\mathcal{X}^{(0,1)}$ equals $[0,1]^d$ by definition. Besides, let $h \geqslant 0$ and $i \in \{1,\dots,2^h\}$. We compute the decomposition $h = kd + r$ with $r \in \{0,\dots,d-1\}$. We have by Step 5.4 of Algorithm 12 that the range of each coordinate $j \neq r+1$ of the bin of the child node $(h+1, 2i)$ remains the same

$$\delta_j^{(h+1,2i)} = \delta_j^{(h,i)} = \begin{cases} 2^{-(k+1)} & \text{if } j \leqslant r \\ 2^{-k} & \text{if } j \geqslant r+2 \end{cases}, \tag{6.6}$$

and the range of coordinate $r+1$ is divided by 2,

$$\delta_{r+1}^{(h+1,2i)} = \delta_{r+1}^{(h,i)}/2 = 2^{-(k+1)}\,. \tag{6.7}$$

Equations (6.6) and (6.7) are also true for the second child $(h+1, 2i-1)$, and this concludes the induction. ∎

6.A.1.1. Proof of Lemma 6.4

Upper bound for N_T. For each node (h,i), we recall that $T^{(h,i)} = \sum_{t=1}^T \mathbb{1}_{\{(h_t,i_t)=(h,i)\}}$ denotes the number of observations predicted by using the local version of Algorithm 11 associated with the sequence of observations $E^{(h,i)}$. The total number of observations T is the sum of $T^{(h,i)}$ over all nodes (h,i). That is,

$$T = \sum_{h=0}^{H_T}\sum_{i=1}^{2^h} T^{(h,i)}\mathbb{1}_{\{(h,i)\in\mathcal{T}_T\}} \geqslant \sum_{h=0}^{H_T}\sum_{i=1}^{2^h} T^{(h,i)}\ \mathbb{1}_{\left\{(h,i) \text{ is an inner node in } \mathcal{T}_T\right\}}.$$

Now we use the fact that each inner node (h,i) has reached its splitting condition (Step 5 of Algorithm 12), that is,

$$T^{(h,i)} + 1 \geqslant \left(\operatorname{diam}\left(\mathcal{X}^{(h,i)}\right)\right)^{-2}.$$

Using that $\operatorname{diam}\left(\mathcal{X}^{(h,i)}\right) \leqslant \sqrt{2d}2^{-h/d}$ by Lemma 6.9, we get

$$\begin{aligned} T &\geqslant \sum_{h=0}^{H_T}\sum_{i=1}^{2^h}\left[-1+\left(\operatorname{diam}\left(\mathcal{X}^{(h,i)}\right)\right)^{-2}\right]\mathbb{1}_{\{(h,i) \text{ is an inner node }\}} \\ &\geqslant \sum_{h=0}^{H_T}\underbrace{\left(-1+\frac{2^{2h/d}}{2d}\right)}_{g(h)}\underbrace{\sum_{i=1}^{2^h}\mathbb{1}_{\{(h,i) \text{ is an inner node }\}}}_{n_h}. \end{aligned} \tag{6.8}$$

Because $g : \mathbb{R}_+ \to \mathbb{R}$ is convex in h, by Jensen's inequality

$$T \geqslant N_T^{\text{in}}\ g\left(\frac{1}{N_T^{\text{in}}}\sum_{h=0}^{H_T} h n_h\right),$$

where $N_T^{\text{in}} = \sum_h n_h$ is the total number of inner nodes. Now, by Lemma 6.10 available in Section 6.A.1.3, because $\mathcal{T}_T$ is a binary tree with N_T nodes in total, it has exactly $N_T^{\text{in}} = (N_T - 1)/2$ inner nodes and the average depth of its inner nodes is lower-bounded as

$$\frac{1}{N_T^{\text{in}}} \sum_{h=0}^{H_T} h n_h \geqslant \log_2\left(\frac{N_T - 1}{8}\right) .$$

Substituting in the previous bound, it implies

$$\begin{aligned} T &\geqslant \frac{N_T - 1}{2} g\left(\log_2\left(\frac{N_T - 1}{8}\right)\right) = \frac{N_T - 1}{2}\left(-1 + \frac{1}{2d} 2^{\frac{2}{d}\log_2\left((N_T-1)/8\right)}\right) \\ &= -\frac{N_T - 1}{2} + \frac{N_T - 1}{4d}\left(\frac{N_T - 1}{8}\right)^{2/d} \\ &\geqslant \underbrace{-\frac{N_T - 1}{2}}_{\geqslant -T/2} + \frac{2}{d}\left(\frac{N_T - 1}{8}\right)^{1+2/d} . \end{aligned}$$

By reorganizing the terms, it entails $dT \geqslant (3/4)dT \geqslant \big((N_T - 1)/8\big)^{1+2/d}$. Thus, $(N_T - 1)/8 \leqslant (dT)^{d/(d+2)}$, which yields the desired bound for N_T.

Upper bound for H_T . We start from (6.8) and we use the fact that for all $h = 0, \dots, H_T - 1$, there exists at least one inner node of depth h in $\mathcal{T}$. Thus,

$$T \geqslant \sum_{h=0}^{H_T - 1}\left(-1 + \frac{2^{2h/d}}{2d}\right) = -H_T + \frac{1}{2d}\frac{2^{2H_T/d} - 1}{2^{2/d} - 1} \geqslant -H_T + \frac{2^{2(H_T-1)/d}}{2d}$$

where the last inequality is because $(a - 1)/(b - 1) \geqslant a/b$ for all numbers $a \geqslant b > 1$. Therefore, by upper-bounding $T \geqslant H_T$, we get $4T \geqslant 2^{2(H_T-1)/d}/d$ and thus $2(H_T - 1)/d \leqslant \log_2(4dT)$ which concludes the proof.

6.A.1.2. Proof of Theorem 6.3

The cumulative regret suffered by Algorithm 12 is controlled by the sum of all cumulative regrets incurred by all local versions of Algorithm 11, each associated with a subsequence of observations $E^{(h,i)}$. That is,

$$\widehat{R}_{L,T} \leqslant \sum_{(h,i)\in\mathcal{T}_T}\left[\sum_{t\in E^{(h,i)}} \ell\big(\widehat{y}_t, y_t\big) - \inf_{f\in\mathcal{L}_L^d} \sum_{t\in E^{(h,i)}} \ell\big(f(\boldsymbol{x}_t), y_t\big)\right] ,$$

where $E^{(h,i)} = \big\{1 \leqslant t \leqslant T \,:\, (h_t, i_t) = (h, i)\big\}$ is the set of time steps assigned to node (h, i). Now, by Lemma 6.2, the cumulative loss incurred by nested EG associated with node (h, i) satisfies

$$\begin{aligned} \sum_{t\in S^{(h,i)}} \ell\big(\widehat{y}_t, y_t\big) &\leqslant \inf_{y\in[0,1]} \sum_{t\in S^{(h,i)}} \ell\big(y, y_t\big) + M\sqrt{T^{(h,i)}\log 2} \\ &\leqslant \inf_{f\in\mathcal{L}_L^d} \sum_{t\in S^{(h,i)}} \ell\big(f(\boldsymbol{x}_t), y_t\big) + ML \underbrace{\operatorname{diam}\Big(\mathcal{X}^{(h,i)}\Big)}_{\leqslant 1/\sqrt{T^{(h,i)}} \text{ by Step 5 of Algorithm 12}} T^{(h,i)} + 2M\sqrt{T^{(h,i)}\log 2} \end{aligned}$$

where the second inequality is by Lemma 6.1. Thus,

$$\widehat{R}_{L,T} \leqslant M\Big(L + \underbrace{2\sqrt{\log 2}}_{\leqslant 3}\Big) \sum_{(h,i)\in\mathcal{T}_T} \sqrt{T^{(h,i)}}\,.$$

Then, by Jensen's inequality,

$$\frac{1}{N_T} \sum_{(h,i)\in\mathcal{T}_T} \sqrt{T^{(h,i)}} \leqslant \sqrt{\frac{1}{N_T} \sum_{(h,i)} T^{(h,i)}} = \sqrt{\frac{T}{N_T}}\,,$$

which concludes the first statement of the theorem. The second statement follows from Lemma 6.4 and because for all $a, b \geqslant 0$, $\sqrt{a+b} \leqslant \sqrt{a} + \sqrt{b}$,

$$\begin{aligned} M(L+3)\sqrt{N_T T} &\leqslant M(L+3)\sqrt{\Big(1 + 4(3dT)^{d/(d+2)}\Big)T} \\ &\leqslant M(L+3)\Big(\sqrt{T} + \sqrt{4(3dT)^{d/(d+2)}T}\Big) \\ &= M(L+3)\Big(\sqrt{T} + 2(3d)^{\frac{d}{2(d+2)}} T^{\frac{d+1}{d+2}}\Big)\,. \end{aligned}$$

6.A.1.3. Lemma 6.10 and its proof

Lemma 6.10. *Let $N \geqslant 1$ be an odd integer. Let $\mathcal{T}$ be a binary tree with N nodes. Then,*

- *its number of inner-nodes equals $N^{\mathrm{in}} = (N-1)/2$.*
- *the average depth (i.e., distance to the root) of its inner nodes is lower-bounded as*

$$\frac{1}{N^{\mathrm{in}}} \sum_{h=0}^{\infty} h\ \#\{\textit{inner nodes in } \mathcal{T} \textit{ of depth } h\} \geqslant \log_2\left(\frac{N-1}{8}\right)\,.$$

Proof *First statement.* We proceed by induction. If $N = 1$, there is only one binary tree with one node, the lone leaf, so that $N^{\mathrm{in}} = 0$. Now, if $\mathcal{T}$ is a binary tree with $N \geqslant 3$ nodes, select an inner node n which is parent of two leaf nodes. Then, replaces the subtree rooted at n by a leaf node. The resulting subtree $\mathcal{T}'$ of $\mathcal{T}$ has $N-2$ nodes, so that by induction hypothesis $\mathcal{T}'$ has $(N-3)/2$ inner nodes. But, $\mathcal{T}'$ has also $N^{\mathrm{in}} - 1$ inner nodes. Therefore $N^{\mathrm{in}} = (N-1)/2$.

Second statement. We note that the average depth is minimized for the equilibrated binary trees, that are such that

- all depths $h \in \{0, \dots, \lfloor \log_2 N^{\mathrm{in}} \rfloor\}$ have exactly 2^h inner nodes;
- no inner nodes has depth $h > \lceil \log_2 N^{\mathrm{in}} \rceil$.

Therefore,

$$\frac{1}{N^{\mathrm{in}}} \sum_{h=0}^{\infty} h\ \#\{\text{inner nodes in } \mathcal{T} \text{ of depth } h\} \geqslant \frac{1}{N^{\mathrm{in}}} \sum_{h=0}^{\lfloor log_2 N^{\mathrm{in}} \rfloor} h 2^h$$

Now, we use that $\sum_{i=0}^{n-1} i 2^i = 2^n(n-2) + 2$ for all $n \geqslant 1$, which implies because $\lfloor \log_2 N^{\mathrm{in}} \rfloor \geqslant$

$\log_2 N^{\text{in}} - 1$ and by substituting in the previous bound,

$$\frac{1}{N^{\text{in}}} \sum_{h=0}^{\infty} h \, \#\{\text{inner nodes in } \mathcal{T} \text{ of depth } h\} \geqslant \underbrace{\frac{2^{\log_2 N^{\text{in}}}}{N^{\text{in}}}}_{=1} \left(\log_2 N^{\text{in}} - 2\right) + \underbrace{\frac{2}{N^{\text{in}}}}_{\geqslant 0} .$$

This concludes the proof by substituting $N^{\text{in}} = (N-1)/2$. ∎

6.A.2. Proofs of Section 6.3

The proof of Lemma 6.5 follows from a simple adaptation of the proof of the regret bound of the exponentially weighted average forecaster—see for instance Cesa-Bianchi and Lugosi [43]. By convexity of ℓ and by Hoeffding's inequality, we have at each time step t

$$\ell(\widehat{y}_t, y_t) \leqslant \sum_{d=0}^{t-1} \widehat{p}_{d,t} \ell(f_{d,t}, y_t) \leqslant -\frac{1}{\eta_t} \log \sum_{d=0}^{t-1} \widehat{p}_{d,t} e^{-\eta_t \ell(f_{d,t}, y_t)} + \frac{\eta_t}{8}$$

By Jensen's inequality, since $\eta_{t+1} \leqslant \eta_t$ and thus $x \mapsto x^{\eta_t/\eta_{t+1}}$ is convex

$$\begin{aligned} \frac{1}{t} \sum_{d=0}^{t-1} \widehat{p}_{d,t} e^{-\eta_t \ell(f_{d,t}, y_t)} &= \frac{1}{t} \sum_{d=0}^{t-1} \left(\widehat{p}_{d,t}^{\frac{\eta_{t+1}}{\eta_t}} e^{-\eta_{t+1} \ell(f_{d,t}, y_t)} \right)^{\frac{\eta_t}{\eta_{t+1}}} \\ &\geqslant \left(\frac{1}{t} \sum_{d=1}^{t-1} \widehat{p}_{d,t}^{\frac{\eta_{t+1}}{\eta_t}} e^{-\eta_{t+1} \ell(f_{d,t}, y_t)} \right)^{\frac{\eta_t}{\eta_{t+1}}} \end{aligned}$$

Substituting in Hoeffding's bound we get

$$\ell(\widehat{y}_t, y_t) \leqslant \left(\frac{1}{\eta_{t+1}} - \frac{1}{\eta_t} \right) \log t - \frac{1}{\eta_{t+1}} \log \left(\sum_{d=0}^{t-1} \widehat{p}_{d,t}^{\frac{\eta_{t+1}}{\eta_t}} e^{-\eta_{t+1} \ell(f_{d,t}, y_t)} \right) + \frac{\eta_t}{8}$$

Now, by definition of the loss update in Step 3 of Algorithm 13, for all $d = 0, \ldots, t-1$

$$\sum_{k=0}^{t-1} \widehat{p}_{k,t}^{\frac{\eta_{t+1}}{\eta_t}} e^{-\eta_{t+1} \ell(f_{k,t}, y_t)} = \frac{t}{t+1} \frac{\widehat{p}_{d,t}^{\frac{\eta_{t+1}}{\eta_t}} e^{-\eta_{t+1} \ell(f_{d,t}, y_t)}}{\widehat{p}_{d,t+1}}$$

which after substitution in the previous bound leads to the inequality

$$\ell(\widehat{y}_t, y_t) \leqslant \ell(f_{d,t}, y_t) + \frac{1}{\eta_{t+1}} \log \left((t+1) \, \widehat{p}_{d,t+1} \right) - \frac{1}{\eta_t} \log(t \, \widehat{p}_{d,t}) + \frac{\eta_t}{8} .$$

By summing over $t = d+1, \ldots, T$, the sum telescopes; using that $\widehat{p}_{d,d+1} = 1/(d+1)$ by Step 3.1.

$$\begin{aligned} &\sum_{t=d+1}^{T} \ell(\widehat{y}_t, y_t) - \sum_{t=d+1}^{T} \ell(f_{d,t}, y_t) \\ &\qquad \leqslant \frac{1}{\eta_{T+1}} \log \left((T+1) \underbrace{\widehat{p}_{d,T+1}}_{\leqslant 1} \right) - \frac{1}{\eta_t} \log \left(\underbrace{(d+1) \, \widehat{p}_{d,d+1}}_{=1} \right) + \frac{1}{8} \sum_{t=d+1}^{T} \eta_t \end{aligned}$$

$$\leqslant \frac{1}{\eta_{T+1}} \log\big(T+1\big) + \frac{1}{8} \sum_{t=d+1}^{T} \eta_t \,.$$

Finally, by routine calculation

$$\begin{aligned}
\sum_{t=d+1}^{T} \eta_t \leqslant 2 \sum_{t=1}^{T} \sqrt{\frac{\log t}{t}} &\leqslant 2\sqrt{\log T} \sum_{t=1}^{T} \frac{1}{\sqrt{t}} \\
&= 2\sqrt{\log T} \left(1 + \sum_{t=2}^{T} \frac{1}{\sqrt{t}}\right) \\
&\leqslant 2\sqrt{\log T} \left(1 + \int_1^T \frac{1}{\sqrt{t}} \,\mathrm{d}t\right) \\
&\leqslant 4\sqrt{T(\log T)} \,,
\end{aligned}$$

which concludes the proof.

6.A.3. Proofs of Section 6.4

6.A.3.1. Proof of Lemma 6.8

The proof is performed in two steps.

Step 1: Lipschitz → Continuous. The set $\mathcal{L}$ of Lipschitz functions, from a compact metric space $\mathcal{X}$ to $\mathbb{R}$, is a subalgebra of the set $\mathcal{C}$ of continuous functions. Besides, $\mathcal{L}$ contains the constant functions and separates the points of $\mathcal{X}$. Therefore, the Stone-Weierstrass theorem, recalled in Theorem 6.11, entails that any continuous function $f : \mathcal{X} \to \mathbb{R}$ is the uniform limit of Lipschitz functions. Thus, the dominated convergence theorem yields

$$\inf_{f \in \mathcal{L}} \mathbb{E}\Big[\ell\big(f(X), Y\big)\Big] = \inf_{f \in \mathcal{C}} \mathbb{E}\Big[\ell\big(f(X), Y\big)\Big] \,.$$

.

Step 2: Continuous → Borel. Second, by the version of Lusin's theorem stated in Theorem 6.12, we can approximate any measurable function by continuous functions (this is where regularity is used).

Let $\delta, \varepsilon > 0$ and $f : \mathcal{X} \to [0,1]$ be a Borel function. By Theorem 6.12, there exists a continuous function $g : \mathcal{X} \to [0,1]$ such that

$$\mathbb{P}_X\Big\{|f - g| \geqslant \delta\Big\} \leqslant \varepsilon \,.$$

Then by Jensen's inequality, and since

$$\begin{aligned}
\Delta \stackrel{\text{def}}{=} \left|\mathbb{E}\Big[\ell\big(f(X), Y\big)\Big] - \mathbb{E}\Big[\ell\big(g(X), Y\big)\Big]\right| &\leqslant \mathbb{E}\Big[\Big|\ell\big(f(X), Y\big) - \ell\big(g(X), Y\big)\Big|\Big] \\
\leqslant \underbrace{\mathbb{P}_X\Big\{|f - g| \geqslant \delta\Big\}}_{\leqslant \varepsilon} &+ \underbrace{\mathbb{E}\Big[M \big|f(X) - g(X)\big| \; \mathbb{1}_{\{|f(X) - g(X)| \leqslant \delta\}}\Big]}_{\leqslant M\delta} \,,
\end{aligned}$$

where the second inequality is because ℓ takes values in $[0,1]$ and is M-Lipschitz in its first argument. Thus $\Delta \leqslant \varepsilon + M\delta$, which concludes the proof since this is true for arbitrary small values of ε and δ.

Theorem 6.11. — Stone-Weierstrass. *Let $\mathcal{C}(X,\mathbb{R})$ be the ring of continuous function on a compact X with the topology of uniform convergence i.e. the topology generated by the norm*

$$\big\|f\big\| = \max_{x\in X} \big|f(x)\big| \qquad f \in \mathcal{C}(X,\mathbb{R})\,.$$

Let $A \subseteq \mathcal{C}(X,\mathbb{R})$ be a subring containing all constant functions and separating the points of X, that is for any two different points $x_1, x_2 \in X$, there exists a function $f \in A$ for which $f(x_1) \neq f(x_2)$. Then A is dense in $\mathcal{C}(X,\mathbb{R})$: every continuous function on X is the limit of a uniformly converging sequence of functions in A.

Proof The proof is carried out in several references, for instance Rudin [130] ■

Theorem 6.12. — Lusin. *If $\mathcal{X}$ is a convex and compact subset of a normed space, equipped with a regular probability measure μ, then for every measurable function $f : \mathcal{X} \to [0,1]$ and for every $\delta, \varepsilon > 0$, there exists a continuous function $g : \mathcal{X} \to [0,1]$ such that*

$$\mu\big\{\big|f-g\big| \geqslant \delta\big\} \leqslant \varepsilon\,.$$

Proof The proof of Theorem 6.12 can be easily derived from the proof of Stoltz and Lugosi [133, Proposition 25]. ■

6.A.3.2. Proof of Theorem 6.7

In this proof, apart from the use of Breiman's generalized ergodic theorem in the beginning and the martingale convergence theorem in the end (as exhibited in Györfi et al. [85], Györfi and Ottucsák [83], Biau et al. [25], Biau and Patra [24]), we resort to new arguments.

Let $d \geqslant 1$ and $L \geqslant 0$. Then, by assumption and by exchanging lim sup and inf,

$$\limsup_{T\to\infty} \frac{1}{T}\left(\sum_{t=1}^{T} \ell\big(\widehat{Y}_t, Y_t\big)\right) \leqslant \inf_{f\in\mathcal{L}_L^d} \limsup_{T\to\infty} \left(\frac{1}{T}\sum_{t=1}^{T} \ell\big(f(Y_{t-d}^{t-1}), Y_t\big)\right)\,.$$

Because ℓ is bounded over $[0,1]^2$ and thus integrable, Breiman's generalized ergodic theorem (see Breiman [33]) entails that the right-term converges: almost surely,

$$\lim_{T\to\infty} \left(\frac{1}{T}\sum_{t=1}^{T} \ell\big(f(Y_{t-d}^{t-1}), Y_t\big)\right) = \mathbb{E}\big[\ell\big(f(Y_{-d}^{-1}), Y_0\big)\big]$$

and thus,

$$\limsup_{T\to\infty} \left(\frac{1}{T}\sum_{t=1}^{T} \ell\big(\widehat{Y}_t, Y_t\big)\right) \leqslant \inf_{f\in\mathcal{L}_L^d} \mathbb{E}\Big[\ell\big(f(Y_{-d}^{-1}), Y_0\big)\Big]\,.$$

By letting $L \to \infty$ in the inequality above, we get

$$\limsup_{T\to\infty} \left(\frac{1}{T}\sum_{t=1}^{T} \ell\big(\widehat{Y}_t, Y_t\big)\right) \leqslant \inf_{f\in\mathcal{L}^d} \mathbb{E}\Big[\ell\big(f(Y_{-d}^{-1}), Y_0\big)\Big]\,.$$

By Lemma 6.8 the infimum over all continuous functions equals the infimum over the set $\mathcal{B}^d$ of

Borel functions. Therefore,

$$\limsup_{T\to\infty}\left(\frac{1}{T}\sum_{t=1}^{T}\ell\big(\widehat{Y}_t,Y_t\big)\right) \leqslant \inf_{f\in\mathcal{B}^d}\mathbb{E}\Big[\ell\big(f(Y_{-d}^{-1}),Y_0\big)\Big] \\ \leqslant \mathbb{E}\Bigg[\underbrace{\inf_{f\in\mathcal{B}^d}\mathbb{E}\Big[\ell\big(f(Y_{-d}^{-1}),Y_0\big)\Big|Y_{-d}^{-1}\Big]}_{\stackrel{\text{def}}{=}Z_d}\Bigg],$$

where the second inequality is by the measurable selection theorem—see Theorem 8 in Appendix I of Algoet [15]. Now, we remark that $\big(Z_d\big)$ is a bounded super-martingale with respect to the family of sigma algebras $\big(\sigma(Y_{-d}^{-1})\big)_{d\geqslant 1}$. Indeed, the function $\inf_{f\in\mathcal{B}^{d+1}}(.)$ is concave, thus conditional Jensen's inequality

$$\mathbb{E}\big[Z_{d+1}\big|Y_{-d}^{-1}\big] \leqslant \inf_{f\in\mathcal{B}^{d+1}}\mathbb{E}\Bigg[\mathbb{E}\Big[\ell\Big(f\big(Y_{-(d+1)}^{-1}\big),Y_0\Big)\Big|Y_{-(d+1)}^{-1}\Big]\Bigg|Y_{-d}^{-1}\Bigg] \\ = \inf_{f\in\mathcal{B}^{d+1}}\mathbb{E}\Big[\ell\Big(f\big(Y_{-(d+1)}^{-1}\big),Y_0\Big)\Big|Y_{-d}^{-1}\Big]$$

Now, we note that

$$\inf_{f\in\mathcal{B}^{d+1}}\mathbb{E}\Big[\ell\Big(f\big(Y_{-(d+1)}^{-1}\big),Y_0\Big)\Big|Y_{-d}^{-1}\Big] \leqslant \inf_{f'\in\mathcal{B}^{d}}\mathbb{E}\Big[\ell\Big(f'\big(Y_{-d}^{-1}\big),Y_0\Big)\Big|Y_{-d}^{-1}\Big] = Z_d\,,$$

which yields $\mathbb{E}\big[Z_{d+1}\big|Y_{-d}^{-1}\big] \leqslant Z_d$. Thus, the martingale convergence theorem (see e.g. Chow [52]) implies that Z_d converges almost surely and in $\mathbb{L}_1$. Thus,

$$\lim_{d\to\infty}\mathbb{E}\big[Z_d\big] = \mathbb{E}\Bigg[\inf_{f\in\mathcal{B}^\infty}\mathbb{E}\Big[\ell\big(f(Y_{-\infty}^{-1}),Y_0\big)\Big|Y_{-\infty}^{-1}\Big]\Bigg] = L^\star\,,$$

which yields the stated result $\limsup_T\sum_{t=1}^{T}\ell\big(\widehat{Y}_t,Y_t\big)/T = L^\star$.

6.B. Uniform histograms

We detail in this appendix the performance bound obtained by competing against uniform histograms over the input space $\mathcal{X}=[0,1]^d$. We denote by $\mathcal{H}_N$ the class of uniform histograms with N hyper-rectangle subregions of equal size. Note that this class exists for $N\in\{i^d,\text{ such that }i\geqslant 1\}$.

Approximation error. First, we bound the approximation error made by the best uniform histogram in $\mathcal{H}_N$ to approximate the unknown best L-Lipschitz objective function $f^\star:[0,1]^d\to[0,1]$. The diameter $\operatorname{diam}(\mathcal{H}_N)$ with respect to the ℓ^2-norm of the bins of a uniform histogram in $\mathcal{H}_N$ equals

$$\operatorname{diam}(\mathcal{H}_N) \stackrel{\text{def}}{=} \max\Big\{\|\boldsymbol{x}_i-\boldsymbol{x}_j\|_2\,:\quad \boldsymbol{x}_i,\boldsymbol{x}_j\in[0,1]^d\,,\quad \forall h\in\mathcal{H}_N\ h(\boldsymbol{x}_i)=h(\boldsymbol{x}_j)\Big\} = \sqrt{d}\ N^{-1/d}\,.$$

Therefore, by applying Lemma 6.1 on each bin and summing over all N bins, the cumulative approximation error of $\mathcal{H}_N$ satisfies for all $L>0$

$$\begin{aligned}\inf_{h\in\mathcal{H}_N}\sum_{t=1}^{T}\ell\big(h(\boldsymbol{x}_t),y_t\big) &\leqslant \inf_{f\in\mathcal{L}_L^d}\sum_{t=1}^{T}\ell\big(f(\boldsymbol{x}_t),y_t\big)+MLT\,\mathrm{diam}\big(\mathcal{H}_N\big)\\ &= \inf_{f\in\mathcal{L}_L^d}\sum_{t=1}^{T}\ell\big(f(\boldsymbol{x}_t),y_t\big)+MLT\sqrt{d}\;N^{-1/d}\,. \end{aligned}\tag{6.9}$$

Estimation error. Now, we bound the additional error obtained by estimating the best histogram in $\mathcal{H}_N$ online. To do so, we resort to EG on the set of 2^N functions that predict the constant values 0 or 1 in each bin of the partition of $\mathcal{H}_N$, we obtain the upper-bound

$$\sum_{t=1}^{T}\ell\big(\widehat{y}_t,y_t\big)\leqslant\inf_{h\in\mathcal{H}_N}\sum_{t=1}^{T}\ell\Big(h(\boldsymbol{x}_t),y_t\Big)+2M\sqrt{T\log\big(2^N\big)}\,.\tag{6.10}$$

Total error. By summing the approximation error (6.9) and the estimation error (6.10), we finally get the regret bound

$$\sum_{t=1}^{T}\ell\big(\widehat{y}_t,y_t\big)\leqslant\inf_{f\in\mathcal{L}_L^d}\sum_{t=1}^{T}\ell\big(f(\boldsymbol{x}_t),y_t\big)+\underbrace{MLT\sqrt{d}\;N^{-1/d}}_{\text{Approximation}}+\underbrace{2M\sqrt{TN\log 2}}_{\text{Estimation}}\,.\tag{6.11}$$

The optimal number of bins N that balances the approximation and the estimation errors need to be optimized. Solving the equality

$$MLT\sqrt{d}\;N^{-1/d}=2M\sqrt{TN\log 2}\,,$$

in N yields the optimal value

$$N=\left(\frac{L}{2}\right)^{\frac{2d}{d+2}}\left(\frac{dT}{\log 2}\right)^{\frac{d}{d+2}}\,.$$

Substituting in (6.11), we get

$$\sum_{t=1}^{T}\ell\big(\widehat{y}_t,y_t\big)\leqslant\inf_{f\in\mathcal{L}_L^d}\sum_{t=1}^{T}\ell\big(f(\boldsymbol{x}_t),y_t\big)+\underbrace{2(4L\log 2)^{\frac{d}{d+2}}}_{\leqslant 6L}Md^{\frac{d}{2(d+2)}}T^{\frac{d+1}{d+2}}\,.$$

Online calibration of N. To achieve the above regret bound, the forecaster need to know the Lipschitz constant L and the time horizon T in advance. We could not find a solution for the calibration of the Lipschitz constant. Therefore, we assumed the constant to be $L=1$ which resulted in the suboptimal linear dependency on L instead of $L^{\frac{d}{d+2}}$. However, we can avoid the assumption that T is known in advance at the cost of a constant factor by resorting to the well-known doubling trick, see Cesa-Bianchi and Lugosi [43]. The idea is to restart the algorithm whenever we reach a time step t such that t is a power of 2. At each restart, we forget all the information gained in the past and we set $N\approx(dt)^{\frac{d}{d+2}}$[§].

[§]We recall that for uniform histograms the number of bins should lie in $\{i^d,\text{ such that }i\geqslant 1\}$

III

Online probabilistic predictions

Introduction

In this part we study probabilistic prediction in the setting of sequential prediction of arbitrary sequences. We are not interested anymore in producing single valued forecasts only but the learner is asked to couple his point prediction with a measure of uncertainty (i.e., interval forecasts or distribution forecasts).

In Chapter 7 we investigate several formulations of the above setting. We highlight the challenge raised by the following question: how to handle uncertainties in a deterministic world? We show that the main difficulty relies on designing the good criterion to evaluate the player. We gather many attempts to do so in this chapter. However in each case either the criterion is impossible to achieve, or it is trivially realized, or it is resolvable by resorting to standard techniques of the setting of individual sequences. Be that as it may, we unfortunately do not make significant contributions to the theory of individual sequences.

Chapter 8 tackles the problem from an empirical point of view. From August to December 2014, I took part in two tracks of the online competition GEFCom14, that aimed at designing state-of-the-art methods to provide probabilistic forecasts of energy data (electricity consumption and electric price). After twelve weeks of intense competition with academic and industrial participants from around the world, our team ranked first for both tracks. In Chapter 8 I summarize the methodology used during the competition. It is mostly based on ideas from generalized additive models, however we also investigate expert aggregation techniques with promising results.

7

How to handle uncertainties in a deterministic world?

In this chapter, I gather some attempts to handle uncertainties in the setting of online prediction of arbitrary sequences with expert advice, see Cesa-Bianchi and Lugosi [43]. More precisely, I address the task of not only producing point forecasts $\widehat{y}_t \in \mathbb{R}$ of the observation y_t to be predicted (such as the electricity consumption, the heat load, or the electricity price) but I aim at coupling $\widehat{y}_t$ with a measure of its uncertainty (i.e., producing interval forecasts). The main aim here is to produce probabilistic forecasts ex-nihilo, in the sense that the experts do not provide uncertainty measures of their forecasts. However, we also look at slightly different settings and we consider several performance criteria. The key message of this chapter is that it is not an easy task to design a correct criterion to evaluate the quality of the uncertainty forecasts. In each case considered, either the criterion is already solved by standard methods of individual sequences or it is impossible to satisfy by the forecaster.

Contents

Nota: This chapter is published here for the first time.

The main purpose of this chapter is to address a request from the operational decision-makers of EDF. When they receive the predictions of the experts, if these are significantly different from each other, the operational decision-makers expect a bigger risk in their prediction. On the other hand, if the individual predictions match, the risk seems to be smaller. Therefore, the decision-makers desire to receive simultaneously the combined forecast $\widehat{y}_t$ and a measure of its uncertainty that matches this basic idea.

In today's competitive and dynamic environment, more and more decision making processes are relying on probabilistic forecasts so as to handle risks. Therefore the interest in probabilistic forecasting is steadily growing in the recent years. An example is the competition Global Energy Forecasting Competition 2014 (see Chapter 8) that aimed at building state-of-the-art techniques for energy probabilistic forecasting. The stochastic literature is rich and many methods of density estimation (such as kernel density estimation) or combining methods (such as Bayesian model averaging) encounter success in a stochastic environment. Some of these methods were already transferred to the setting of individual sequence (such as online density estimation with log loss, cf. Section 7.1.1); see also Dawid and Vovk [57] and Shafer and Vovk [131] that bridge the individual sequences and the deterministic world.

Literature of probabilistic forecast in individual sequences In the setting of prediction of arbitrary sequences $y_1, \dots, y_n \in \mathbb{R}$ a large part of the literature focuses on predicting only single valued forecasts $\widehat{y}_1, \dots, \widehat{y}_n \in \mathbb{R}$ without looking at the uncertainties of the forecasts (see the monograph of Cesa-Bianchi and Lugosi [43]). However, several works already looked at probabilistic forecasts.

Among this work we can cite the notion of calibration introduced by Brier [37] in the weather forecasting literature (see also Cesa-Bianchi and Lugosi [43, Chapter 4.5]). To summarize, the notion of calibration considers sequences of observations y_t in $\{0, 1\}$ (such as sunny and rainy days). The learner aims at each time step at predicting by $\widehat{p}_t$ the probability of rain, i.e., $y_t = 1$. A sequence of forecasts is called well calibrated when it is consistent in hindsight. For instance, we want that the proportion of rainy days such that the prediction of rain was about 30% (i.e., $\widehat{p}_t \approx 0.3$) turn out to be approximatively 30% as well. More formally, a well-calibrated forecaster satisfies for all probabilities $p \in [0, 1]$ and for all $\varepsilon > 0$,

$$\lim_{n\to\infty} \left| \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}_{\{|\widehat{p}_t - p| \leqslant \varepsilon\}} (\widehat{p}_t - y_t) \right| = 0 \,.$$

Quite surprisingly, calibrated forecasters exist (see Foster and Vohra [67], or Mannor and Stoltz [112] for a nice geometric proof).

Another area of research focuses on estimating online the density of the deterministic sequence $y_1, \dots, y_n$ by minimizing the cumulative logarithmic loss (i.e., the negative log-likelihood). This problem was formally introduced in data compression literature by Shtarkov [132] and early suggested by Rissanen [128] (see also the monograph of Grünwald [82] for an overview). The aim was to produce a code that is as good as the best code in some reference set (i.e., the set of experts). The cumulative log loss corresponded to the incurred code length and their strategy led to well used data compression programs like context-tree weighting (CTW). To do so, they considered various strategies such as the Bayesian strategy (i.e., the exponentially weighted average forecaster with log loss and with learning rate $\eta = 1$) or smoothed maximum likelihood strategies. Online density

estimation of arbitrary sequences with log loss has then been extensively studied in the learning literature. One can cite for instance Azoury and Warmuth [21], Freund [69], Dasgupta and Hsu [56], Raginsky et al. [123], and Kotlowski and Grünwald [104]. Together they constructed strategies (often based on Follow the Leader algorithm) to compete against the best of an exponential family of distributions, including Bernoulli, Gamma, or Gaussian families.

In this chapter, we propose a brief overview of attempts to produce probabilistic forecasts when the learner has access to a set of expert advice to be combined. We recall how to solve this problem under a log-scoring rule and investigate other possible performance criteria inspired from the stochastic literature. We see that the main difficulty relies on building the proper criterion to evaluate the quality of the sequence of probabilistic forecasts. For all the criteria we thought about either the player could easily minimize the criterion by applying well known techniques directly (e.g., the cumulative logarithmic loss) or the criterion is impossible to meet. To formalize the problem, we distinguish the following three settings according to the information provided on their uncertainty by the individual forecasters (experts) to be combined.

Full uncertainty measure. Here we suppose that at each time step $t \geqslant 1$ the experts supply probability forecasts of y_t by producing probability density functions $f_{k,t} : \mathbb{R} \to \mathbb{R}_+$. We intend to combine these distributions sequentially. We consider this setting in Section 7.1. Borrowing from the stochastic literature, we consider three goals for the player, that aims at minimizing:

- the cumulative logarithmic loss (Section 7.1.1): already studied and solved by the literature;
- the Kolmogorov statistic (Section 7.1.2): trivially satisfied;
- the cumulative pinball loss (Section 7.1.3): resolvable by standard algorithms.

Partial uncertainty measure. Here the experts and the learner predict intervals. This setting is addressed in Section 7.2. We successively attempt to design and patch a criterion that balances the empirical level of the predicted intervals and their average width. The solution we came to, in order get a correct criterion here, is to consider randomized forecasts. The setting is then revolvable by standard tools.

No uncertainty measure. In this setting, the experts do not supply any measure of uncertainty of their forecasts $x_{k,t} \in \mathbb{R}$. The aim is to produce a measure of uncertainty of $\widehat{y}_t$ ex-nihilo. This setting is considered in Section 7.3. It was the primary goal of this chapter, which was motivated by the intuition of the operational decision-makers of EDF described earlier. In the end, we retain two methods based on:

- the cumulative log loss: already analyzed by the literature, see Dasgupta and Hsu [56];
- the cumulative pinball loss: resolvable by standard algorithms. We use it in Chapter 8.

7.1. The experts provide probability distributions

We consider the setting of sequential robust aggregation of distributions, which is described in Figure 7.1.

The goal of the learner is to minimize some criterion that may depend on the sequence of observations $(y_t)_{1\leqslant t\leqslant n}$, the sequences of expert forecasts $(f_{k,t})_{1\leqslant t\leqslant n}$ for $1 \leqslant k \leqslant K$, and the sequence of distribution forecasts of the learner $(\widehat{f}_t)_{1\leqslant t\leqslant n}$. Hereafter we recall or investigate three criteria inspired by the stochastic literature.

At each time step $t = 1, \dots, n$
1. the experts supply density forecasts $f_{k,t}$ for $k \in \{1, \dots, K\}$;
2. the learner produces his own density forecast $\widehat{f}_t$;
3. the nature reveals $y_t \in \mathbb{R}$.

Figure 7.1.: Generic setting of online aggregation of probability distributions

7.1.1. Cumulative logarithmic loss (well studied)

This criterion is similar to the goal of maximizing the log-likelihood in the i.i.d. batch setting. More formally the accuracy of the predicted distribution $\widehat{f}_t$ proposed at round t for the observation y_t is evaluated by the logarithmic loss $\ell\big(\widehat{f}_t, y_t\big) = -\log \widehat{f}_t(y_t)$. The goal of the learner is to minimize his cumulative logarithmic loss defined as:

$$\sum_{t=1}^{n} \ell\big(\widehat{f}_t, y_t\big) = \sum_{t=1}^{n} -\log \widehat{f}_t(y_t)\,.$$

Remark 7.1. As we recall in the introduction, log-scoring rules have already been well studied in the online learning literature (see, e.g., Grünwald [82]). Thus we only explain below in a few lines how to solve this criterion by resorting to standard algorithms.

Using the fact that the logarithmic loss is 1-exp-concave (i.e., $f \mapsto \exp\big(-\ell(f, y)\big) = f(y)$ is concave for all $y \in \mathbb{R}$) and from Cesa-Bianchi and Lugosi [43, Section 9.1] the exponentially weighted average forecaster with learning rate $\eta = 1$ achieves the following regret bound

$$\sum_{t=1}^{n} -\log \widehat{f}_t(y_t) \leqslant \min_{1 \leqslant k \leqslant K} \sum_{t=1}^{n} -\log f_{k,t}(y_t) \;\; + \log K\,.$$

In other words, the average loss suffered by the forecaster converges to the average performance of the best expert at rate at most $\mathcal{O}(1/n)$.

7.1.2. Kolmogorov's criterion (new but trivially satisfied)

Another axis of research is to look at the quality of the cumulative distribution function $\widehat{F}_t : x \mapsto \int_{\infty}^{x} \widehat{f}_t(u)\,\mathrm{d}u$ instead of the quality of the density function $\widehat{f}_t$. However as we shall see, none of the attempts will be acceptable: the player will always find a way to satisfy the criterion while producing useless forecasts.

Starting point: Kolmogorov's statistic in the i.i.d setting. The idea is to adapt Kolmogorov's statistic that aims at comparing a sample of i.i.d. observations $(Y_t)_{1 \leqslant t \leqslant n}$ with a reference distribution F_{ref}. It is defined as follows:

$$T_n \overset{\mathrm{def}}{=} \big\| F_{\mathrm{ref}} - F_n^{(\mathrm{emp})} \big\|_{\infty}\,,$$

where $F_n^{(\mathrm{emp})}(x) \triangleq \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}_{\{Y_t \leqslant x\}}$ is the empirical distribution of $(Y_t)_{1 \leqslant t \leqslant n}$. From the Glivenko-Cantelli theorem, if the Y_i are distributed with cumulative distribution F_{ref} then $T_n \to 0$ almost surely. The criterion T_n can thus be used in a batch setting to assess the quality of a cumulative

distribution F_{ref} on the i.i.d. sample (Y_i).

Adaptation to a sequential setting. Here, we adapt Kolmogorov's statistic to a sequential setting. We still consider that the sequence of outcomes $(Y_t)_{t\geqslant 1}$ is generated by some underlying stochastic process however we do not assume anymore that the observations are identically distributed. For all $t \geqslant 1$, we denote by F_t the cumulative distribution of Y_t knowing the past. That is,

$$F_t : \quad x \mapsto P(Y_t \leqslant x | \mathcal{F}_{t-1}) \,,$$

where $\mathcal{F}_{t-1}$ is the set of past observations Y_s until time $s = t-1$. Then, if the Y_i are independent and if the F_t are continuous for all t, we have $F_t(Y_t) \sim \mathcal{U}_{[0,1]}$. Therefore applying the Glivenko-Cantelli theorem, the sequence of empirical distribution of $(F_t(Y_t))_{1\leqslant t\leqslant n}$ converges almost surely in supremum norm to the one of the uniform distribution over $[0,1]$, i.e., to the identity function on $[0,1]$. That means

$$T_n = \sup_{\alpha\in[0,1]} \left| \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}_{\{F_t(Y_t)\leqslant\alpha\}} - \alpha \right| \underset{n\to\infty}{\longrightarrow} 0 \qquad \text{a.s.}$$

The above convergence may still be valid in case where the Y_t are not independent by extending the Glivenko-Cantelli theorem to conditional distributions.

Returning to the setting of arbitrary sequences. Kolmogorov's statistic yields a criterion to evaluate the quality of a sequence of predicted cumulative distributions $\widehat{F}_1, \ldots, \widehat{F}_n$ for any arbitrary sequence of real observations $y_1, \ldots, y_n$:

$$\widehat{T}_n \overset{\text{def}}{=} \sup_{\alpha\in[0,1]} \left| \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}_{\left\{\widehat{F}_t(y_t)\leqslant\alpha\right\}} - \alpha \right| .$$

We call it Kolmogorov's criterion. Note that it is a continuous version (in α) of the rank diagrams in the ensemble forecasting literature (see Mallet [109]). Because there is no underlying convexity, we might well think that the player cannot easily solve Kolmogorov's criterion.

Why this criterion is meaningless. However, the player can easily trick the criterion by making $T_n \to 0$ at rate $\mathcal{O}(1/n)$ without having to look neither at the expert forecasts nor at the past observations. To do so, it suffices to play at time t, the cumulative distribution

$$\widehat{F}_t : x \mapsto \frac{t}{n} \mathbb{1}_{\{-B\leqslant x<B\}} + \mathbb{1}_{\{B\leqslant x\}} \,,$$

where B is some bound on the observations (it can be any sufficiently large number). Thus, for all times $t \in \{1, \ldots, n\}$ and for all possible observations y_t we have $\widehat{F}_t(y_t) = t/n$ which entails

$$\widehat{F}_n^{(\text{emp})}(\alpha) \overset{\text{def}}{=} \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}_{\left\{\widehat{F}_t(y_t)\leqslant\alpha\right\}} = \alpha$$

for all $\alpha \in \{1/n, 2/n, \ldots, 1\}$ and therefore $\widehat{T}_n \leqslant 1/n$ because $\widehat{F}_n^{(\text{emp})}$ is non decreasing. The adaptation in n can be performed by doubling trick or by considering dyadic refinements of $[0,1]$.

The same argument is valid for Cramer-von-Mises-type and absolute-error-type criteria, respectively

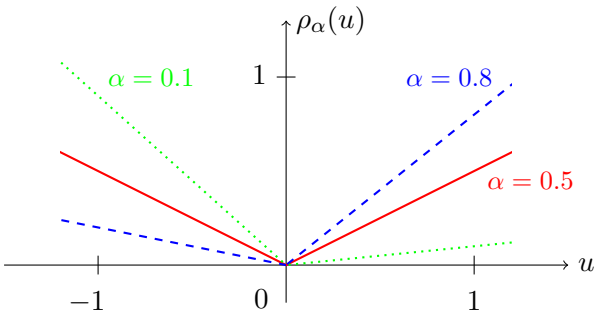


Figure 7.2.: The pinball loss for quantiles $\alpha \in \{0.1, 0.5, 0.8\}$

defined by

$$\int_0^1 \left(\frac{1}{n} \sum_{t=1}^{n} \mathbb{1}_{\{\widehat{F}_t(y_t) \leqslant \alpha\}} - \alpha \right)^2 \mathrm{d}\alpha \quad \text{and} \quad \int_0^1 \left| \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}_{\{\widehat{F}_t(y_t) \leqslant \alpha\}} - \alpha \right| \mathrm{d}\alpha \,.$$

7.1.3. Cumulative pinball loss (solved by standard tools)

A third way to perform probabilistic prediction is to look at quantiles. Once again we borrow ideas from the i.i.d. setting. This approach can be solved by standard tools of individual sequences and enjoys good forecasting performance. This is the solution followed in our practical experiments (see Chapter 8).

Starting point: the pinball loss in the i.i.d. setting. Here we show that in the i.i.d setting a powerful tool exists to perform quantile regression: the pinball loss.

Definition 7.1. *Let Y be a real-valued random variable with cumulative distribution F. Then the quantile q_α of order $\alpha \in (0,1)$ of Y is defined as the generalized inverse of F:*

$$q_\alpha \stackrel{\text{def}}{=} F^{-1}(\alpha) = \inf \{ y \in \mathbb{R}, F(y) \geqslant \alpha \} \,. \tag{7.1}$$

Lemma 7.2 shows that the quantile q_α minimizes in expectation a loss function, that we call the pinball loss.

Lemma 7.2. *Let $\alpha \in (0,1)$. Let Y be an integrable real-valued random variable. Then the quantile q_α satisfies*

$$q_\alpha = \inf \left\{ \arg\min_{\theta \in \mathbb{R}} \mathbb{E}\big[\rho_\alpha(Y - \theta)\big] \right\} ,$$

where ρ_α is the pinball loss defined for all $u \in \mathbb{R}$ by $\rho_\alpha(u) = u(\alpha - \mathbb{1}_{\{u<0\}})$.

A proof of Lemma 7.2 is available in Biau and Patra [24, Lemma 2.1]. The pinball loss is plotted in Figure 7.2 for three different levels of α. In particular for $\alpha = 0.5$ it yields back to the absolute value. Lemma 7.2 states in this case that the median (i.e., $q_{0.5}$) minimizes the expected absolute error (i.e., $2\rho_{0.5}$).

Proposition 7.3 explains why it is interesting to minimize the cumulative pinball loss in the i.i.d

setting. It expresses the fact that the minimizer of the pinball loss (i.e., the empirical quantiles $\widehat{q}_{\alpha,n}$) converges to the true quantile q_α at rate $\mathcal{O}(1/\sqrt{n})$ with high probability.

Proposition 7.3. *Let $\alpha \in (0,1)$. Let $(Y_t)_{1\leqslant t\leqslant n}$ be a sample of i.i.d real-valued random variables with density function f which is bounded from below by a positive constant c in the γ-neighborhood of q_α. Then, the minimization problem*

$$\arg\min_{\theta\in\mathbb{R}} \sum_{t=1}^{n} \rho_\alpha(Y_t - \theta)$$

has a minimal solution, denoted $\widehat{q}_{\alpha,n}$, which satisfies with probability at least $1-\delta$

$$|\widehat{q}_{\alpha,n} - q_\alpha| \leqslant \frac{1}{c}\sqrt{\frac{\log(2/\delta)}{2n}} \qquad \text{for all } \delta \in [2\exp(-2nc^2\gamma^2), 1].$$

Proof Let $F_n^{(\mathrm{emp})}$ be the empirical cumulative distribution of $(Y_t)_{1\leqslant t\leqslant n}$. First, we remark that by Lemma 7.2,

$$\widehat{q}_{\alpha,n} = \big(F_n^{(\mathrm{emp})}\big)^{-1}(\alpha)$$

is the α-quantile of the empirical distribution. Then, we follow the proof of Rivoirard and Stoltz [129, Theorem 8.13]. Let $\varepsilon \geqslant \gamma > 0$. We have

$$\begin{aligned}
\mathbb{P}\big(q_\alpha + \varepsilon < \widehat{q}_{\alpha,n}\big) &= \mathbb{P}\Big(F_n^{(\mathrm{emp})}(q_\alpha+\varepsilon) < \alpha\Big) = \mathbb{P}\bigg(\sum_{t=1}^{n} \mathbb{1}_{\{Y_t \leqslant q_\alpha+\varepsilon\}} < n\alpha\bigg) \\
&= \mathbb{P}\bigg(\sum_{t=1}^{n} \Big(\mathbb{1}_{\{Y_t \leqslant q_\alpha+\varepsilon\}} - \mathbb{P}\big(Y_t \leqslant q_\alpha+\varepsilon\big)\Big) < n\big(\varepsilon - F(q_\alpha+\varepsilon)\big)\bigg) \\
&= \mathbb{P}\bigg(\sum_{t=1}^{n} \Big(\mathbb{1}_{\{Y_t \leqslant q_\alpha+\varepsilon\}} - \mathbb{P}\big(Y_t \leqslant q_\alpha+\varepsilon\big)\Big) < nc\varepsilon\bigg),
\end{aligned}$$

where the last inequality is because the density function f is lower-bounded by $c>0$ on $[q_\alpha-\gamma, q_\alpha+\gamma]$. Performing the same calculations with $\mathbb{P}\big(q_\alpha - \varepsilon > \widehat{q}_{\alpha,n}\big)$ and applying Hoeffding's inequality, we obtain

$$\mathbb{P}\Big(\big|\widehat{q}_{\alpha,n} - q_\alpha\big| > \varepsilon\Big) \leqslant 2\exp\big(-2nc^2\varepsilon^2\big),$$

which concludes the proof. ∎

Back to the setting of arbitrary sequences. We return to the setting detailed in Figure 7.1. Fix a quantile $\alpha \in (0,1)$. For each density function $f_{k,t}$ we can define a matching quantile $q_{k,t}^{(\alpha)} \stackrel{\mathrm{def}}{=} \inf\{y\in\mathbb{R} : \int_{-\infty}^{y} f_{k,t}(x)\,\mathrm{d}x \geqslant \alpha\}$. The performance of the predicted quantile $\widehat{q}_t^{(\alpha)}$ (that corresponds to $\widehat{f}_t$) proposed at round t for observation y_t is assessed by the pinball loss $\ell(\widehat{q}_t^{(\alpha)}, y_t) = \rho_\alpha\big(y_t - \widehat{q}_t^{(\alpha)}\big)$. The goal of the forecaster is then to minimize his cumulative loss

$$\sum_{t=1}^{n} \rho_\alpha\Big(y_t - \widehat{q}_t^{(\alpha)}\Big).$$

By convexity of the pinball loss, by applying standard convex optimization algorithm (such as the exponentially gradient forecaster of Cesa-Bianchi and Lugosi [43] or ML-Prod of Chapter 2) the

learner can guarantee a cumulative pinball-loss bounded as follows

$$\sum_{t=1}^{n} \rho_\alpha(y_t - \widehat{q}_{\alpha,t}) \;\leqslant\; \min_{\boldsymbol{w}\in\Delta_K} \sum_{t=1}^{n} \rho_\alpha(y_t - \boldsymbol{w}\cdot\boldsymbol{q}_t) \;+\; \Xi\sqrt{n\log K}\,,$$

where Ξ is a constant as a function of K and n; $\boldsymbol{q}_t \overset{\text{def}}{=} \left(q_{1,t}^{(\alpha)}, \ldots, q_{K,t}^{(\alpha)}\right)$ are the α-quantiles predicted by the experts; and $\Delta_K \overset{\text{def}}{=} \left\{\boldsymbol{x} \in \mathbb{R}_+^d : \sum_i x_i = 1\right\}$ is the simplex.

7.2. The experts provide prediction intervals

We now turn to the partial information setting. The player still has to make online sequential probabilistic prediction over a series of rounds, with the help of K experts. However in each round $t = 1, \ldots, n$, the experts do not provide full probability distributions but they only supply prediction intervals of level $1-\alpha$. The setting is summarized in Figure 7.3.

At each time step $t = 1, \ldots, n$

1. the experts supply prediction intervals $I_{k,t}$ for $k \in \{1, \ldots, K\}$ of level $1-\alpha$;
2. the learner produces his own prediction interval $\widehat{I}_t$ of level $1-\alpha$;
3. the nature reveals $y_t \in \mathbb{R}$.

Figure 7.3.: Generic setting of online aggregation of prediction intervals

Similarly to the previous section, we investigate here several criteria that can be used to assess the quality of the sequence of forecasts formed by the player. However as we will show all our tentatives to design a good criterion fail: they all suffer weak points that the learner can exploit to succeed without worrying about making good forecasts.

Tentative 1 (failed: trivially satisfied). First we can require that the proportion of time the y_t fall in the $\widehat{I}_t$ turn out to be approximatively $1-\alpha$. A natural criterion would thus be to control the following quantity

$$\left| \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}_{\left\{y_t \in \widehat{I}_t\right\}} - (1-\alpha) \right| .$$

Yet this is not sufficient since the player can easily fool the environment by choosing $\widehat{I}_t = \emptyset$ for $t \leqslant \alpha\, n$ and $\widehat{I}_t = \mathbb{R}$ for the remaining times.

Tentative 2 (failed: impossible to satisfy or trivially satisfied). To patch the performance criterion we add a regularization term that penalizes large intervals. We obtain the criterion

$$\left| \frac{1}{n} \sum_{t=1}^{n} \mathbb{1}_{\left\{y_t \in \widehat{I}_t\right\}} - (1-\alpha) \right| + \frac{\lambda}{n} \sum_{t=1}^{n} \left|\widehat{I}_t\right| , \tag{7.2}$$

where $\lambda > 0$ is a regularization parameter and $|I| = \max_{x_1,x_2\in I} |x_2 - x_1|$ is the diameter of interval I. Now we can assume that the term inside the absolute value is nonpositive because if $(1/t)\sum_{s=1}^{t-1} \mathbb{1}_{\left\{y_s \in \widehat{I}_s\right\}} \geqslant 1-\alpha$ the player can play $\widehat{I}_t = \emptyset$ at the following round to fool the criterion

and improve his performance. In order to avoid this trick we modify the criterion by removing the absolute value from (7.2). The new goal is then to control the quantity

$$\underbrace{\frac{1}{n}\sum_{t=1}^{n}\mathbb{1}_{\{y_t\notin\widehat{I}_t\}}}_{\text{empirical level}} \;+\; \lambda\cdot\underbrace{\frac{1}{n}\sum_{t=1}^{n}\big|\widehat{I}_t\big|}_{\text{average width}} \;=\; \underbrace{\frac{1}{n}\sum_{t=1}^{n}\mathbb{1}_{\{y_t\notin\widehat{I}_t\}}+\lambda\big|\widehat{I}_t\big|}_{\text{instantaneous loss}}\;. \tag{7.3}$$

Quite surprisingly the desired level of confidence $\alpha\in(0,1)$ does not appear in (7.3) because the quantity to control is independent of α. Criterion (7.3) highlights the trade-off that arises between the empirical level of the prediction interval and its average width. The regularization parameter λ balances this compromise. It needs to be calibrated by the forecast-makers according to the level of confidence they desire. Unfortunately Criterion 7.3 makes once again no sense. If the player cannot use random predictions (i.e., $\widehat{I}_t$ is a random variable unknown from the nature), according to the value of λ either the environment can keep the cumulative regret

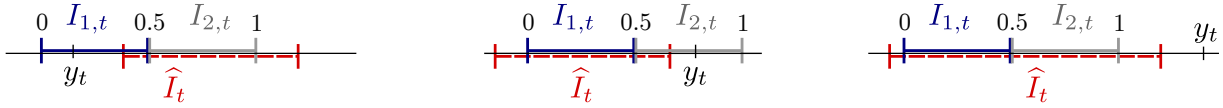
$$\widehat{R}_n=\sum_{t=1}^{n}\Big(\mathbb{1}_{\{y_t\notin\widehat{I}_t\}}+\lambda\big|\widehat{I}_t\big|\Big)-\min_{k=1,\dots,K}\sum_{t=1}^{n}\Big(\mathbb{1}_{\{y_t\notin I_{k,t}\}}+\lambda\big|I_{k,t}\big|\Big)$$

linear in n, or the player trivially succeeds (by making $\widehat{R}_n$ nonpositive). We provide below an illustration of this fact.

Example 7.2. Let $K=2$ (i.e., we have two experts). At each time step $t\geqslant 1$ the first expert predicts $I_{1,t}=[0,1/2]$ and the second expert outputs $I_{2,t}=[1/2,1]$. Now we show that, except for trivial case, it is impossible for the player to guarantee a sub-linear regret uniformly over all outcome sequences. To do so, we consider a sequence of observations $y_t\in\mathbb{R}$ such that

$$y_t\notin\widehat{I}_t \qquad\text{and}\qquad y_t\in\begin{cases} I_{1,t} & \text{if } I_{1,t}\not\subset\widehat{I}_t\\ I_{2,t} & \text{if } I_{1,t}\subset\widehat{I}_t \text{ and } I_{2,t}\not\subset\widehat{I}_t\end{cases} \tag{7.4}$$

Note that if $I_{1,t}\subset\widehat{I}_t$ and $I_{2,t}\subset\widehat{I}_t$, the observation y_t can be any point that does not fall into $\widehat{I}_t$. The possible scenarios are depicted in Figure 7.4. The instantaneous loss incurred for each scenario by the player and the experts are summarized in Table 7.1.



(a) $I_{1,t}\not\subset\widehat{I}_t$ (b) $I_{1,t}\subset\widehat{I}_t$ and $I_{2,t}\not\subset\widehat{I}_t$ (c) $I_{1,t}\cup I_{2,t}\subset\widehat{I}_t$

Figure 7.4.: Graphical representation of the three possible scenarios of (7.4)

	Scenario	Learner ($\widehat{\ell}_t$)	Expert 1 ($\ell_{1,t}$)	Expert 2 ($\ell_{2,t}$)
(a)	$I_{1,t}\not\subset\widehat{I}_t$	$1+\lambda\big\lvert I_{k,t}\big\rvert\geqslant 1$	$\lambda/2$	$1+\lambda/2$
(b)	$I_{1,t}\subset\widehat{I}_t,\ I_{2,t}\not\subset\widehat{I}_t$	$1+\lambda\big\lvert I_{k,t}\big\rvert\geqslant 1+\lambda/2$	$1+\lambda/2$	$\lambda/2$
(c)	$I_{1,t},I_{2,t}\subset\widehat{I}_t$	$1+\lambda\big\lvert I_{k,t}\big\rvert\geqslant 1+\lambda$	$1+\lambda/2$	$1+\lambda/2$

Table 7.1.: Instantaneous loss suffered by the expert and by the players according to the situation.

Then for any value of $\lambda > 0$ either the player can fool the environment or the opposite.

- If $\lambda \geqslant 2$, the player trivially succeeds. It only needs to predict $\widehat{I}_t = \emptyset$. It is thus in scenario (a) at all rounds and it suffers instantaneous loss 1, when expert 1 and 2 will respectively incur losses 1 and 2. The cumulative regret of the learner is then nonpositive.
- If $\lambda < 2$, no matter what the player does the cumulative regret grows linearly in n. Let us respectively denote by n_a, n_b, and n_c the number of rounds scenario (a), (b), and (c) of Table 7.1 occur. From Table 7.1, we get that the cumulative regret with respect to expert 1 is at least
$$\Big(1 - \frac{\lambda}{2}\Big) n_a + \frac{\lambda}{2} n_c \tag{7.5}$$
and with respect to the second expert using $n_a + n_b + n_c = n$
$$-\frac{\lambda}{2} n_a + n_b + \frac{\lambda}{2} n_c = n - \Big(1 + \frac{\lambda}{2}\Big) n_a + \Big(\frac{\lambda}{2} - 1\Big) n_c \,. \tag{7.6}$$
Solving the minimization problem
$$\min_{n_a, n_c \in \mathbb{R}_+} \min \left\{ \Big(1 - \frac{\lambda}{2}\Big) n_a + \frac{\lambda}{2} n_c, n - \Big(1 + \frac{\lambda}{2}\Big) n_a + \Big(\frac{\lambda}{2} - 1\Big) n_c \right\}$$
(by equalizing (7.5) and (7.6)) we obtain the following lower bound on the regret
$$\widehat{R}_n \geqslant \frac{\lambda(2-\lambda)}{4 + 2\lambda}\, n \,,$$
for the optimal values $n_a = 2\lambda n/(2+\lambda)$ and $n_c = (2-\lambda)n/(2+\lambda)$. This concludes the example.

Randomized prediction (solved through standard tools). A solution is to allow the learner to output randomized predictions, see Cesa-Bianchi and Lugosi [43, Chapter 4]. We suppose that at each time step $t \geqslant 1$ the player chooses a distribution $\widehat{\boldsymbol{p}}_t = (\widehat{p}_{1,t}, \dots, \widehat{p}_{K,t})$ over the set of experts and predicts the interval $I_{k,t}$ with probability $p_{k,t}$. We call $\widehat{I}_t$ the player's prediction at time t. Furthermore we assume that the environment is an oblivious opponent: the sequence of observations y_t is determined before the start of the game (i.e., y_t is a deterministic number that cannot depend on the random variable $\widehat{I}_t$). The instantaneous loss suffered by the player at time t is then $\ell(\widehat{I}_t, y_t) = \mathbb{1}_{\{y_t \notin \widehat{I}_t\}} + \lambda |\widehat{I}_t|$. It is also a random variable. Then from Cesa-Bianchi and Lugosi [43, Corollary 4.2] for any $\delta \in (0,1)$ the exponentially average forecaster with $\eta = (1 + \lambda B)^{-1} \sqrt{8 \log K / n}$ satisfies with probability at least $1 - \delta$
$$\widehat{R}_n \leqslant (1 + \lambda B) \sqrt{n/2} \left(\sqrt{\log K} + \sqrt{\log(1/\delta)} \right) .$$
Here, we used the fact that the instantaneous losses belong to $[0, 1 + \lambda B]$ where B is an upper-bound on $\max_{k,t} |I_{k,t}|$.

Other similar trade-off criteria can be considered. Unfortunately, we could only find criteria that were either trivially fooled by the player, or impossible to achieve, or convex and thus solved by standard techniques. For instance, another way to assess the quality of a sequence of prediction intervals is to look at the cumulative pinball loss of their extremities.

7.3. The experts only provide point forecasts

We now look at our last setting where the experts do not provide any measure of uncertainty. At each time $t \geqslant 1$, the player observes a vector of K point predictions $\boldsymbol{x}_t = (x_{1,t}, \dots, x_{k,t}) \in \mathbb{R}^K$, the player is then asked to predict the next probability distribution (e.g., by providing a density $\widehat{f}_t$) and the environment reveals the observation $\widehat{y}_t$. The goal of this section is again to find a good criterion to assess the quality of the sequence of predictions $\widehat{f}_t$.

The starting point of this setting was an observation of the operational team that forecasts the electric load in France. When the operational forecasters receive the individual predictions, if these are significantly different from each other, they conclude to a higher predictive risk than if the individual forecasts are similar. This idea can be formalized in the standard setting of online prediction with expert advice by the following quantity

$$v_t = \sum_{k=1}^{K} p_k^\star \left(\boldsymbol{p}^\star \cdot \boldsymbol{x}_t - x_{k,t}\right)^2 = \mathrm{Var}_{\boldsymbol{p}^\star}[Y_t]\,,$$

where $\boldsymbol{p}^\star = (p_1^\star, \dots, p_K^\star) \in \Delta_K$ is the best weight combination and Y_t is a random variable that equals $x_{k,t}$ with probability $p_k^\star$. Indeed, if several experts with large weight disagree, the random prediction Y_t will have a large variance and the prediction will be uncertain. Besides experts with bad performance will have a small weight $p_k^\star$ and will not impact much the value of v_t if their prediction is far away from the predictions of other experts.

This quantity leads to the so-called second order bounds (see Chapter 2 or Cesa-Bianchi et al. [45]) on the regret. It is possible achieve the following regret bound

$$\sum_{t=1}^{n} \widehat{\ell}_t \leqslant \min_k \left\{ \sum_{t=1}^{n} \ell_{k,t} + \Xi \sqrt{(\log K) \textstyle\sum_{t=1}^{n} \widehat{v}_t} \right\},$$

through a version of the exponentially weighted average forecaster with variable learning rate (see Cesa-Bianchi et al. [45]). Here, Ξ is a constant as a function of K and n; $\widehat{\ell}_t$ and $\ell_{k,t}$ are the losses respectively suffered by the algorithm and by the experts at time t; and $\widehat{v}_t = \sum_{k=1}^{K} \widehat{p}_{k,t} \big(\widehat{\ell}_t - \ell_{k,t}\big)^2$ is an approximation of v_t computed by the algorithm. In some sense, $\widehat{v}_t$ thus controls the uncertainty of the regret of the combining algorithm. But it does not include the uncertainty due to the approximation error, i.e., the uncertainty about the performance of the best expert. If the experts are highly correlated (e.g., because they are designed by using the same covariates), they may be mistaken all together at the same time and $\widehat{v}_t$ will not catch this error.

Example 7.3. For instance, the environment may choose $\boldsymbol{x}_t = (0, \dots, 0) \in \mathbb{R}^K$ and $y_t = 1$ so that the square loss incurred by the player and by the experts is 1 although $\widehat{v}_t = 0$ predicts a very small risk. A small value of $\widehat{v}_t$ actually means that the regret of the algorithm with respect to the best expert will not increase a lot.

7.3.1. Cumulative logarithmic loss (well studied)

In order to build a satisfactory criterion, we need to consider the uncertainty about the approximation error as well. A solution is to compare the performance of the predicted distributions $\widehat{f}_t$

formed by the player with the performance of a family of distributions. To do so, we consider the logarithmic loss $\ell(\widehat{f}_t, y_t) = -\log \widehat{f}_t(y_t)$. This opens the toolbox of online density estimation with logarithmic loss, see Section 7.1.1. We evaluate the performance of the player relatively to the comparison class parametrized by $k = 1, \dots, K$ and $\sigma > 0$ that produces at time t the density

$$\widehat{f}_{k,t}^{(\sigma)} : y \mapsto K\left(\frac{y - x_{k,t}}{\sigma}\right) \Big/ \sigma \,,$$

where K is a kernel (i.e., a nonnegative, integrable, and even function such that $\int_{\mathbb{R}} K(u)\,\mathrm{d}u = 1$). For instance we can choose the family of Gaussian kernels $K : x \mapsto (1/\sqrt{2\pi})\exp(-x^2/2)$ or the sliding windows kernels $K : x \mapsto \mathbb{1}_{\{|x| \leqslant 1/2\}}$. The windows parameter σ needs to be optimized. The goal of the player is to minimize his cumulative regret defined as

$$\widehat{R}_n \stackrel{\text{def}}{=} \sum_{t=1}^{n} -\log \widehat{f}_t(y_t) - \min_{k=1,\dots,K} \left\{ \inf_{\sigma>0} \sum_{t=1}^{n} -\log f_{k,t}^{(\sigma)}(y_t) \right\} . \tag{7.7}$$

In other words the player aims at competing in cumulative logarithmic loss with the best distribution centered in an expert and with fixed variance (see Section 7.1.1).

To our knowledge previous literature (e.g., [132, 128, 82, 21, 56, 104]) in online density estimation with a log-scoring rule did not consider this exact setting with multiple experts. However, we show below that the above setting can easily be reduced to standard online density estimation (with no expert) and thus solved by resorting to known techniques.

Simplification to a single expert. Now, we note that the above setting can be simplified by considering a single expert only and by focusing on the calibration of the window parameter $\sigma > 0$. Indeed, we can partition the analysis in two layers. First, for each expert $k \in \{1, \dots, K\}$ we predict sequentially $\widehat{f}_{j,t}$ that guarantees small regret

$$\widehat{R}_{k,n} \stackrel{\text{def}}{=} \sum_{t=1}^{n} -\log \widehat{f}_{k,t}(y_t) - \inf_{\sigma>0} \sum_{i=1}^{n} -\log f_{k,t}^{(\sigma)}(y_t) \,.$$

Then, we apply the algorithm presented in Section 7.1.1 in order to compete with the best sequence of density forecasts $(\widehat{f}_{k,t})_{t\geqslant 1}$. We then have

$$\widehat{R}_n \leqslant \max_{k=1,\dots,K} \widehat{R}_{k,n} \; + \log K \,.$$

Therefore, if we can control each $\widehat{R}_{k,n}$ individually, $\widehat{R}_n$ can be controlled easily.

The Gaussian kernel. Dasgupta and Hsu [56] analyze a similar setting of online Gaussian density estimation which captures the above setting (with an additional bias removal of x_t) for the particular case of the Gaussian kernel defined for all $x > 0$ by $K(x) = (1/\sqrt{2\pi})\exp(-x^2/2)$. They conclude the following statements:

- $\widehat{R}_{k,n}$ is difficult to control uniformly over all $\sigma > 0$.
- $\widehat{R}_{k,t}$ is easily bounded from below through an adapted version of the Follow the Leader forecaster for $\sigma > \sigma_0$. The regret if then of order $\mathcal{O}(1/(n\sigma_0^2) + 1/\sigma_0 + \log n)$ which is sub-linear as soon as $\sigma_0 \gg 1/n$. They also discuss the case $\sigma > 1/n$.

The extension to other kernels is left for future research.

7.3.2. Cumulative pinball loss (the criterion for the next chapter)

Another solution to form probabilistic forecasts is to minimize the cumulative pinball loss of the combined predictions. The idea is substantially similar to the one of Section 7.1.3. Therefore, we give only a brief description here. The only difference is that the experts do not output quantiles $q_{k,t}$ but provide only point forecasts $x_{k,t}$ (of the average). The expert vectors $\boldsymbol{q}_t = (q_{1,t}, \ldots, q_{K,t})$ of Section 7.1.3 are substituted with $\boldsymbol{x}_t = (x_{1,t}, \ldots, x_{K,t})$.

For any $\alpha \in (0,1)$, and any bounded convex set $B \subset \mathbb{R}^K$, we can use any standard combination rule (such as Ridge) which guarantees

$$\sum_{t=1}^{n} \rho_\alpha(y_t - \widehat{q}_{\alpha,t}) \;\leqslant\; \inf_{\boldsymbol{w}\in B} \sum_{t=1}^{n} \rho_\alpha(y_t - \boldsymbol{w}\cdot\boldsymbol{x}_t) \;+\; o(n)\,, \qquad \text{when } n \to \infty,$$

where $\widehat{q}_{\alpha,t}$ are the α-quantiles predicted by the algorithm.

By performing multiple combining algorithms in parallel for several values of $\alpha \in [0,1]$, we can forecast complete distributions $\widehat{f}_t$. Such a methodology is used in the next chapter so as to forecast the electricity price.

8

Semi-parametric models and robust aggregation for GEFCom2014 probabilistic electric load and electricity price forecasting

We sum up the methodology of the team Tololo on the Global Energy Forecasting Competition 2014 for the electric load and electricity price forecasting tracks. During the competition, we used and tested many statistical and machine learning methods such as random forests, gradient boosting machines, or generalized additive models. In this paper, we only present the methods that have shown the best results. For electric load forecasting, our strategy consists in producing temperature scenarios that we plug into a probabilistic forecasting load model. Both steps are performed by fitting a quantile generalized additive model (quantGAM). Concerning the electricity price forecasting, we investigate three methods that we used during the competition. The first method follows the spirit of the one used for electric load. The second one is based on combining a set of individual predictors. The last one fits a sparse linear regression on a large set of covariates. We chose to present in this paper these three methods because they show good performance and have a nice potential of improvements for future research.

Contents

NOTA: This chapter is a joint work with Yannig Goude and Raphaël Nedellec. It is based on the submitted paper [7].

8.1. Introduction

We present in this paper the methodology employed for the probabilistic electric load and electricity price forecasting tracks of the Global Energy Forecasting Competition 2014 (GEFCom2014). We participated in both tracks but with different intensity and motivation. Load forecasting was a familiar field of research for us before the competition, whereas we were inexperienced with price forecasting. As a consequence, we converged rapidly to a unique solution for load forecasting, but we constantly changed our method as we were learning and improving our knowledge of electricity price forecasting.

Quantile regression based on pinball loss minimization (see Koenker and Bassett 101) and generalized additive models (see Hastie and Tibshirani 88, Wood 147) are the main tools of our work. To our knowledge, there was no off-the-shelf program achieving quantile generalized additive models and we implemented our own solution for that. We designed it originally for load forecasting but at the end it turned out to be the most efficient method for both tasks. We present it in Section 8.2. We tested a wide range of other approaches for the price forecasting task. Among those we describe those which deserve to be shared. To our opinion, they have a potential for improvement and can be applied to other forecasting problems. Aggregation of experts is considered in Section 8.5. We were inspired by the work of Nowotarski and Weron [118] and extend it to the case where the weights of the combination can vary over time. More precisely, we adapt to quantile regression the setting of robust online aggregation of experts (see Cesa-Bianchi and Lugosi 43) which has already been applied successively for point wise load forecasting in Devaine et al. [60] and Gaillard and Goude [75]. Our set of 13 experts consists of forecasters from the price forecasting literature (AR, TAR, ARX, TARX, PAR as presented in Weron and Misiorek 145), GAMs, and popular machine learning: random forest (see Breiman 35) and gradient boosting machines (see Friedman 72). The third approach presented in Section 8.6 is based on covariate selection with ℓ_1 selection procedure, commonly known as Lasso regression introduced in Tibshirani [136]. It was motivated by the fact that we generated a lot of covariates (192) from the original ones (for price forecasting). Thus, we were curious at the end of the competition to see how an automatic procedure could select an optimal subset among them. ℓ_1 selection and quantile regression were studied in Belloni and Chernozhukov [23] but never applied neither to price nor to load forecasting. To our experience, no open source code exists that satisfies our needs for the competition. We present in Section 8.6 a kernel based

approach we developed at this occasion. For the price forecasting task, the results obtained during the competition differ slightly (sometimes better, sometimes worse) from those obtained by the three methods (quantile GAM, quantile mixture, quantile GLM). This is largely due to the other approaches that we used along the competition, and to hybrid variants of those presented here. We deliberately focus on these three methods in this paper for conciseness.

The Global Energy Forecasting Competition 2014

First, let us provide a brief description of the Global Energy Forecasting Competition 2014 (GEFCom2014). More details are available in Hong et al. [95]. GEFCom2014 was held from August to December 2014. It aimed at bringing together the state-of-the-art techniques and methodologies for probabilistic energy forecasting.

The competition attracted hundreds of participants from around the worlds including academic and industrial teams. The competition featured four tracks to predict electric load, electricity price, wind power, and solar power. In the sequel, we focus on the first two tracks in which we participated.

Each challenge was designed on a rolling basis. The data was parsed into sixteen batches. The first batch was the estimation set. It was used by the participants to train their forecasting models and was provided at the start of the contest. The following fifteen batches (called validation sets) were sequentially predicted by the participants, then released by the competition organizers, and added to the estimation set so as to forecast the next batch.

More formally, we set $\{t_1, \dots, t_{16}\}$ an ordered sequence of integers such that $1 < t_1 < \cdots < t_{16} = n$ where n is the total number of data. We define the set $S_k = \{t_k + 1, \dots, t_k + 1\}$ for $k \in \{1, \dots, 15\}$ and $S_0 = \{1, \dots, t_1\}$. The set S_k is called the k-th batch. During the competition period*, the contestants were asked each week $k = 1, \dots, 15$ to form probabilistic forecasts of the next data by providing the percentiles $\tau \in A \overset{\text{def}}{=} \{0.01, \dots, 0.99\}$ of future observations y_t for $t \in S_k$. To do so the participants had knowledge of previous data in $S_0 \cup \cdots \cup S_{k-1}$ only. At the end of each week, the new data S_k were revealed and the scores of the participants were computed as:

$$\frac{1}{t_{k+1} - t_k} \sum_{t=t_k+1}^{t_{k+1}} \frac{1}{99} \sum_{\tau \in A} \rho_\tau(y_t - \widehat{q}_{\tau,t}),$$

where ρ_τ is the pinball loss defined for all $u \in \mathbb{R}$ by $\rho_\tau(u) = u(\tau - \mathbb{1}_{\{u<0\}})$. Its interest in quantile regression is motivated in Section 7.1.3.

Track 1: electric load. 362 contestants grouped into 34 teams took part in this forecasting track. The estimation set S_0 ranged from January 1, 2006 to August 31, 2010. The fifteen validation sets consist of every month from September 2010 to December 2011.

Track 2: electricity price. This contest brought together 287 participants grouped into 19 teams. The estimation set S_0 included data from January 1, 2011 to June 5, 2013. The starting time of the fifteen validation sets are displayed in Table 8.2. Because of the high volatility of electricity price the participants were only asked to provide forecasts for the first 24 hours of each validation set.

*from August 15 to December 6, 2014

8.2. Quantile regression with Generalized Additive Models

We consider the supervised regression setting where we are asked to forecast an univariate response variable $Y_t \in \mathbb{R}$ (such as the load) according to several covariates $\boldsymbol{X}_t = (X_{t,1}, \ldots, X_{t,d}) \in \mathbb{R}^d$ (such as the temperature). A training sample $\{(\boldsymbol{X}_t, Y_t)\}_{t=1}^n$ is available.

8.2.1. Generalized Additive Models

Generalized additive models (GAMs) were introduced by Hastie and Tibshirani [88]. GAMs explain the output Y_t as a sum of smooth functions of the different covariates $X_{t,j}$. More formally, we assume that for all time $t = 1, \ldots, n$, $Y_t = \mu(\boldsymbol{X}_t) + \varepsilon_t$ where μ is the unknown function to be estimated and the ε_t denote zero mean random variables that are independent and identically distributed (i.i.d.) from some exponential family distribution[†]. GAMs assume that it exists a link function g such that

$$g\big(\mu(\boldsymbol{X}_t)\big) = f_1(X_{t,1}) + f_2(X_{t,2}) + f_3(X_{t,3}, X_{t,4}) + \ldots \tag{8.1}$$

where the f_j are smooth functions of the covariates $X_{t,k} \in \mathbb{R}$. In the following, the link function g is the identity and the smooth functions f_j are cubic splines (unless specified otherwise). Basically, cubic splines are polynomials of degree 3 that are joined at points known as "knots" by satisfying some continuity constraints (see Wood 147 for details). We call $\mathcal{S}(K_i)$ the class of cubic splines for some fixed set K_i of knots.

We fit the smooth effects f_i with penalized regression methods. To do so, we first choose the knots K_i for each effect f_i. Then, we use the ridge regression that minimizes over all effects $f_1 \in \mathcal{S}(K_1), f_2 \in \mathcal{S}(K_2), \ldots$ the following criterion:

$$\sum_{t=1}^{n} \left(Y_t - \sum_{i=1}^{p} f_i\Big(X_t^i\Big) \right)^2 + \sum_{i=1}^{p} \lambda_i \int \big\| f_i''(x) \big\|_2^2 dx, \tag{8.2}$$

where for each effect X_t^i are one or two covariates of $\boldsymbol{X}_t$ corresponding to effect f_i. Here $\lambda_1, ..., \lambda_p > 0$ are regularization parameters that control the degree of smoothness of each effect (the higher λ_i the smoother f_i is). They have to be optimized. The knots K_i are uniformly distributed over the range of the covariate(s) X_t^i corresponding to effect f_i. The number of knots (i.e., the cardinal of K_i) is another way to control the smoothness of the effect f_i and should be optimized as well. These problems are solved by using the methodology presented in Wood [147] which consists in minimizing the Generalized Cross Validation criterion (GCV). The method is implemented in the R package `mgcv` (see 147).

8.2.2. Quantile regression

Quantile regression was introduced by Koenker and Bassett [101]. Let Y be a real value random variable and let $\boldsymbol{X}$ be a set of explanatory variables. If $F_{Y|\boldsymbol{X}}$ denotes the conditional cumulative distribution of Y given $\boldsymbol{X}$, then the conditional quantile q_τ of order $\tau \in [0, 1]$ of Y knowing X is defined as the generalized inverse of $F_{Y|\mathbf{X}}$:

$$q_\tau(Y|\boldsymbol{X}) = F^{-1}_{Y|\boldsymbol{X}}(\tau) = \inf\left\{ y \in \mathbb{R}, F_{Y|\boldsymbol{X}}(y) \geqslant \tau \right\}. \tag{8.3}$$

[†]Throughout the paper, ε_t are i.i.d. error terms but their distribution may change from a display to another

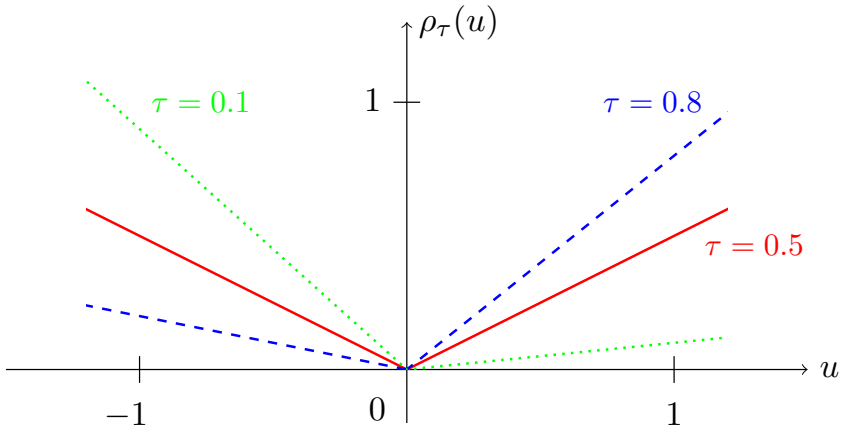


Figure 8.1.: The pinball loss for quantiles $\tau \in \{0.1, 0.5, 0.8\}$

Now, the idea of quantile estimation arises from the observation that the median (i.e., $q_{0.5}(Y|\boldsymbol{X})$) minimizes the expected absolute error. More generally, it can be shown that the conditional quantile $q_\tau(Y|\boldsymbol{X})$ is the solution of the minimization problem:

$$q_\tau(Y|\boldsymbol{X}) \in \arg\min_g \mathbb{E}[\rho_\tau(Y - g(\boldsymbol{X}))|\boldsymbol{X}]\,, \tag{8.4}$$

where ρ_τ is the pinball loss defined for all $u \in \mathbb{R}$ by $\rho_\tau(u) = u(\tau - \mathbf{1}_{\{u<0\}})$. The pinball loss is plotted in Figure 8.1.

Linear quantile regression is implemented in the `R`-package `quantreg` (see Koenker 100). It assumes that $\{(\mathbf{X}_t, Y_t)\}_{t=1,\dots,n}$ are i.i.d. such that $Y_t = \mathbf{X_t}^\top \boldsymbol{\beta} + \varepsilon_t$, where $\boldsymbol{\beta} \in \mathbb{R}^d$ is a vector of unknown parameters. Linear quantile regression solves the following convex minimization problem:

$$\widehat{\boldsymbol{\beta}}_\tau \in \arg\min_{\boldsymbol{\beta}\in\mathbb{R}^d} \sum_{t=1}^{n} \rho_\tau\big(Y_t - \mathbf{X}_t^\top \boldsymbol{\beta}\big)\,. \tag{8.5}$$

Then, it estimates the conditional quantile q_τ with $\widehat{q}_\tau : \boldsymbol{x} \mapsto \boldsymbol{x}^\top \widehat{\boldsymbol{\beta}}_\tau$.

8.2.3. Mixed approach: use smooth effects estimated by GAM as input for linear quantile regression

In this section, we introduce a generic procedure to perform quantile regression by using GAMs. An obvious patch would be to substitute in (8.2) the square loss with the pinball loss. However, the optimization problem is harder and we faced numerical problems trying to solve it. During the competition, we used a multiple steps approach to deal with these computational problems. We detail the methodology below.

1. We linearize the problem by fitting GAM. To do so, we perform the two following steps.
   1.a. *Fit the mean.* We fit GAM by minimizing Criterion (8.2) over the non-linear effects $f_1 \in S(K_1), f_2 \in S(K_2), \dots$. We get estimates of the effects (denoted $\widehat{f}_i$ for all i) that capture the non-linear relationships between the conditional mean of Y_t and the covariates. We denote by $\widehat{Y}_t$ the fitted estimate of observation Y_t formed by GAM.
   1.b. *Fit the variance* (optional). We perform a residual analysis to model the deviations from

the mean knowing the covariates. To do so, we fit GAM to predict the square residuals $\left(Y_t - \widehat{Y}_t\right)^2$. We obtain estimates of the effects (denoted $\widehat{g}_i$) that trap the second-order variation of Y_t. This step is optional.

2. Finally, for each percentile $\tau \in \{0.01, 0.02, ..., 0.99\}$, we perform a linear quantile regression by using the estimated effects $\boldsymbol{Z}_t = \left(\widehat{f}_1(X_{t,1}), \widehat{f}_2(X_{t,2}), \ldots, \widehat{g}_1(X_{t,1}), \widehat{g}_2(X_{t,2}), \ldots\right)$ as covariates to estimate the quantile function $\widehat{q}_\tau$. To do so, we substitute in (8.5) the vector of covariates $\boldsymbol{X}_t$ with the vector of fitted effect $\boldsymbol{Z}_t$ and solve the minimization problem.

We call this procedure quantGAM.

8.3. Probabilistic electric load forecasting by quantGAM

We consider the data available for the Probabilistic Load Forecasting Track of the GEFCom2014 competition. It includes hourly observations of load consumption (from January 1, 2006 to December 31, 2011) and of temperatures from twenty-five weather stations (from January 1, 2001 to December 31, 2011). Our goal is to forecast at the end of each month the τ-quantiles of the load consumption of the next month for $\tau \in \{0.01, \ldots, 0.99\}$. At time step t, the performance of a prediction $\left(\widehat{q}_{0.01,t}, \ldots, \widehat{q}_{0.99,t}\right) \in \mathbb{R}^{99}$ is measured by the average pinball loss over the quantiles defined as:

$$\frac{1}{99}\sum_{\tau=1}^{99} \rho_\tau(Y_t - \widehat{q}_{\tau,t})\,. \tag{8.6}$$

The data set is parsed into two pieces: a training set from 2001 to 2010 and a testing set in 2011. The testing set is predicted online (month by month) by fitting the methods on all the past observations (including the beginning of the testing set). The electric demand and the temperature heavily depend on the hour of the day. Therefore the models described throughout this section are performed per hour (unless stated otherwise). That is, the data is partitioned into 24 independent time series (one for each hour of the day) and 24 separate models are fitted.

The importance of the past consumptions and temperatures is clearly decreasing over time. This has driven us to use two approaches depending on the forecasting horizon: one taking into account these dependencies for "short-term" load forecasting and a second approach for "mid-term" probabilistic load forecasting.

Section 8.3.4 reports the performance of the method obtained for each month of the testing set.

8.3.1. Working on the data

These are the variables that have been defined in order to build the forecasting models and methods:

- Y_t is the electric load at time $t \geqslant 1$[‡].
- T_t is a uniform weighted average of the temperatures of the weather stations 6, 10, 22, and 25. We choose these stations with a relatively simple method. Considering the simplified hourly GAM of equation (8.10), we test successively the impact of each temperature station. We choose these four stations using generalized cross validation scores. We represent on Figure 8.2

[‡]Throughout the paper, $t \geqslant 1$ is a linear chronological index (in hours) for the whole dataset.

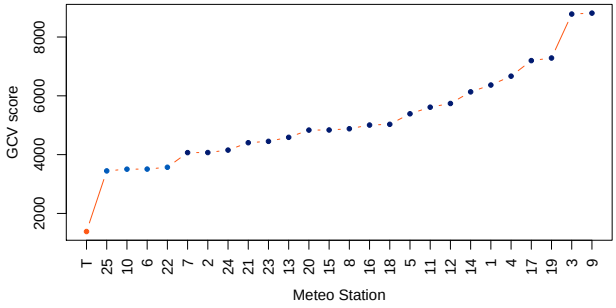


Figure 8.2.: GCV score obtained for each temperature station compared to the one obtained by the average temperature T.

the GCV score obtained for each temperature station compared to the one obtained by T_t. We clearly see that the 4 selected stations have similar GCV scores and that averaging them brings a significant improvement.

- $T_t^{(\gamma)}$ is a smoothed temperature of T_t with exponential smoothing parameter $\gamma \in [0,1]$. It is defined at time t by induction as:

$$T_t^{(\gamma)} \triangleq \gamma\, T_{t-1}^{(\gamma)} + (1-\gamma)\, T_t\,. \tag{8.7}$$

- $Toy_t \in [0,1]$ (Time of year) is a cyclic variable that indicates the annual position and repeats each year. It is each year linearly increasing over time going from 0 on January 1 at 00:00 to 1 on December 31 at 23:30.
- $DayType_t$ is a factorial variable with 7 levels corresponding to different types of day. The levels are: Monday, Tuesday-Wednesday-Thursday, Friday, Saturday, Sunday, bank holidays, and a last category corresponding to the days before and after bank holidays. This choice was driven by our expertise on electricity load data (see e.g., 81).

8.3.2. Medium-term probabilistic forecasting of the load

To forecast the electric demand at more than two days horizons, we use a medium term probabilistic model that does not use recent lags of the load. In fact, we validated (by performing cross validation) the correct horizon where recent observations of the temperature (by using scenarios) were still informative. It appears that the right horizon was around 48 hours.

Our method divides into separate layers the uncertainty of the model and the uncertainty due to the temperature. First we build a medium-term probabilistic forecasting model of the temperature only based on the impact of the annual position Toy_t. Then we fit a model that forecast the distribution of the load conditionally to the temperature (assuming that true temperature is known in advance). Both models are performed by using quantGAM described in Section 8.2.3. We form our final prediction of the load distribution by averaging the forecasted conditional distribution of the load over the forecasted law of the temperature.

Probabilistic forecasting of the temperature We perform quantGAM, by following the steps of the generic procedure of Section 8.2.3, as follows:

1.a. We estimate the non-linear effect of the annual position Toy_t on the expected temperature by fitting GAM (i.e., by minimizing Criterion (8.2) substituting the response variable Y_t with the

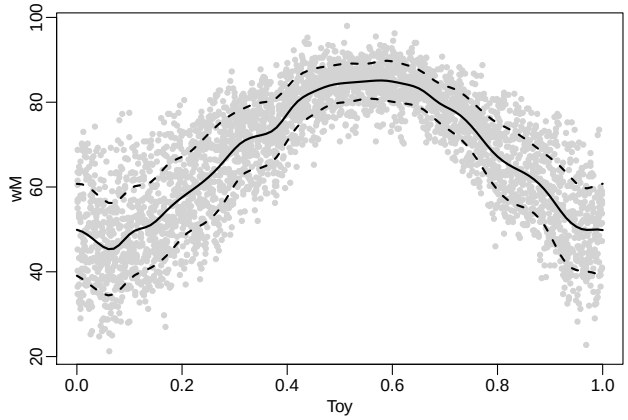


Figure 8.3.: Observed values of T_t together with the smooth functions $\widehat{f}_1$ and $\widehat{f}_1 \pm \widehat{g}_1$ fitted by Models (8.8) and (8.9).

temperature T_t) with the following model:

$$T_t = f_1(Toy_t) + \varepsilon_t \,. \tag{8.8}$$

Here f_1 is estimated by cubic cyclic regression splines to assure continuity of estimated effects at midnight the 1st of January (see Wood 147). We denote by $\widehat{f}_1$ the obtained estimate of f_1 (i.e., the solution of the minimization of (8.2)).

1.b. We estimate the non-linear effect that impacts the forecasting errors by fitting GAM on the residual signal with model:

$$\Big(T_t - \widehat{f}_1\big(Toy_t\big)\Big)^2 = g_1(Toy_t) + \varepsilon_t \,, \tag{8.9}$$

where g_1 is estimated by cubic cyclic regression splines. We call $\widehat{g}_1$ the estimate of g_1.

2. We perform a linear quantile regression (see Section 8.2.2) to forecast the quantiles of the temperature by using

$$\boldsymbol{Z_t} = \Big(\widehat{f}_1(Toy_t),\ \widehat{g}_1(Toy_t)\Big)$$

as vector of covariates. The final estimates of the τ-quantiles, denoted $\widehat{T}_{\tau,t}$, are thus linear combinations of the mean effect $\widehat{f}_1$ and the variance effect $\widehat{g}_1$. That is, they are of the form:

$$\widehat{T}_{\tau,t} = \widehat{a}_{\tau,1}\,\widehat{f}_1(Toy_t) + \widehat{a}_{\tau,2}\,\widehat{g}_1(Toy_t)\,,$$

where $\widehat{a}_{\tau,1}, \widehat{a}_{\tau,2} \in \mathbb{R}$ are the linear coefficients estimated by the quantile regression.

Figure 8.3 plots the estimates $\widehat{f}_1$ and $\widehat{f}_1 \pm \widehat{g}_1$ together with the observed values of the temperatures T_t.

Probabilistic forecasting of the load (knowing the temperature in advance) We fit quantGAM as follows:

1.a. We estimate the non-linear effects that impact the expected load by fitting GAM with model:

$$Y_t = f_1(Toy_t) + f_2(t) + f_3(T_t) + h(DayType_t) + \varepsilon_t \,, \tag{8.10}$$

where f_1, f_2, and f_3 are cubic regression splines and h is a function that takes a different value for each type of day. We recall that the estimation of GAM is performed by minimizing (8.2) over all cubic splines $f_i \in S(K_i)$ and over all $h \in \{\text{Monday}, \ldots\}^{\mathbb{R}}$. Note that h actually corresponds to seven additional coefficients that do not appear in the regularization term of (8.2). Note that f_2 captures the trend.

1.b. We estimate the non-linear effects that impact the forecasting errors by fitting GAM on the residual signal with model:

$$\left(Y_t - \widehat{Y}_t\right)^2 = g_1(Toy_t) + g_2(T_t) + \varepsilon_t \,, \tag{8.11}$$

where $\widehat{Y}_t$ is the load fitted by Model (8.10).

2. We perform a linear quantile regression to forecast the quantiles of the load by using the covariate vector

$$\boldsymbol{Z}_t = \left(\widehat{f}_1(Toy_t),\ \widehat{f}_2(t),\ \ldots\ ,\ \widehat{g}_1(Toy_t),\ \widehat{g}_2(T_t)\right).$$

Figure 8.4 plots the forecasted distribution of the load for three consecutive days by using the observed values of T_t. The forecasted distribution does not take into account the uncertainty due to the temperature and the obtained confidence intervals are thus extremely small.

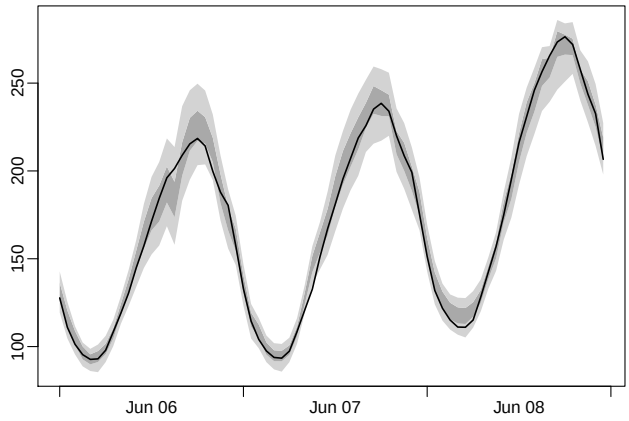


Figure 8.4.: Medium-term forecasted distribution of the load from June 6, 2011 to June 8, 2011 by using the real values of the temperature T_t. As in all the following plots of probabilistic distributions, we only plot the 50% confidence interval in dark gray and the 90% confidence interval in light gray.

Probabilistic forecasting of the load (operational forecast) To provide forecasts of the load distribution, we do not have at our disposal the future true values of the temperature T_t. To deal with it, we average the forecasted distributions of the load over the forecasted distribution of temperature. In other words, to generate quantile forecasts of the load at each time step t:

1. We forecast all the quantiles $\tau \in \{0.01, \ldots, 0.99\}$ of the temperature at time step t by quantGAM described earlier (see Equations (8.8) and (8.9)).
2. For each predicted quantile $\widehat{T}_{\tau,t}$ of the temperature, we perform quantGAM (described in Models (8.10) and (8.11)) by substituting the true value of T_t with the predicted quantile $\widehat{T}_{\tau,t}$ and we obtain a distribution $\widehat{F}_{\tau,t}$ of the load.
3. We form the final prediction of the load distribution by averaging over all temperature per-

centiles τ the forecasted distribution $\widehat{F}_t = (1/99)\sum_{\tau=1}^{99} \widehat{F}_{\tau,t}$. In the end for each level $\tau' \in \{0.01, \ldots, 0.99\}$ we predict the percentile of the load $\widehat{q}_{\tau',t} = \widehat{F}_t^{-1}(\tau')$ by inverting the forecasted distribution $\widehat{F}_t$.

Figure 8.5 plots the forecasted distribution for the same three consecutive days as for Figure 8.4. We remark that the obtained confidence intervals are much wider and less accurate than in Figure 8.4 which knew the true values of T_t.

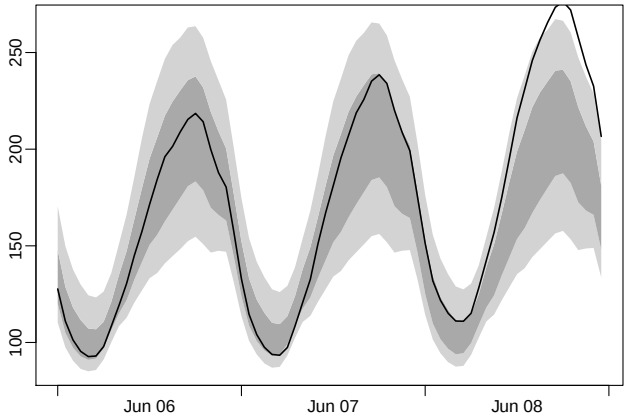


Figure 8.5.: Medium-term forecasted distribution of the load from June 6, 2011 to June 8, 2011 obtained by averaging the forecasted distributions of the load over the forecasted distribution of the temperature.

8.3.3. Forecasting up to two days' horizon.

In order to forecast at a "short-term" horizon (up to two days ahead), we gain in accuracy by considering recent lag of the temperature. We thus built another method for this purpose based on Monte Carlo methods.

Basically, similarly to the medium-term model this method partitions the analysis into two different layers. Both of them use the quantGAM method described in Section 8.2.3. First, we generate 800 temperature scenarios by sampling step by step (one hour ahead) the next value of the temperature. Second, we plug the temperature scenarios into a probabilistic forecasting model of the load that was fitted with the true values of the temperature as exogenous variable. The final prediction of the load distribution is obtained by averaging the forecasted distributions over the 800 simulated scenarios.

We explain below how the temperature scenarios are generated.

Generating randomly the temperature scenarios We generate the temperature scenarios (for T_t and for the smoothed temperatures $T_t^{(0.8)}$ and $T_t^{(0.95)}$) as follows. First, we remove the annual seasonality by estimating the medium-term Model (8.8). Second, we consider the residual signal $e_t \triangleq T_t - \widehat{f}_1(Toy_t)$. Finally, we fit quantGAM so as to predict the distribution of the next residual temperature (one hour ahead):

1.a. We fit the expected residuals e_t according to Model (8.12) to take into account autocorrelation within the data:

$$e_t = \alpha_1 e_{t-1} + \alpha_2 e_{t-2} + \ldots + \alpha_{48} e_{t-48} + \varepsilon_t \,. \tag{8.12}$$

Here, α_i are coefficients to be estimated. Contrary to the other models, this residual analysis is not performed per hour.

1.b. Then, we estimate the non-linear seasonality of the square error of the model. Indeed, we observed that the variance was subject to the annual position (Toy_t). Thus, we consider the fitted errors $(e_t - \widehat{e}_t)^2$ and we fit GAM with model:

$$\left(e_t - \widehat{e}_t\right)^2 = g_1(Toy_t) + \varepsilon_t\,, \tag{8.13}$$

where g_1 is estimated by cubic cyclic regression splines.

2. We perform a linear quantile regression (presented in Section 8.2.2) for all percentiles $\tau \in \{0.01, \ldots, 0.99\}$ with covariates

$$\boldsymbol{Z}_t = \left(\widehat{\alpha}_1 e_{t-1},\ \widehat{\alpha}_2 e_{t-2},\ \ldots,\ \widehat{\alpha}_{48} e_{t-48},\ \widehat{g}_1(Toy_t)\right) \in \mathbb{R}^{49}\,.$$

Here $\widehat{\alpha}_i$ are the coefficients estimated in Step (1.a) and $\widehat{g}_1$ is the annual effect estimated by Model (8.13). We denote by $\widehat{e}_{\tau,t}$ the one hour ahead predicted τ-quantiles of the residual e_t.

To generate a temperature scenario for the next 48 hours, we perform sequentially (step by step) for time $\in \{t+1, \ldots, t+48\}$ the following procedure: we forecast the next τ-quantiles $\widehat{e}_{\tau,s}$ for all $\tau \in \{0.01, \ldots, 0.99\}$; we sample the next residual e_s uniformly over the set of percentiles $\widehat{e}_{\tau,s}$; we compute the next temperature $T_s = \widehat{f}_1(Toy_s) + e_s$; we move to step $s+1$. The values of $T_s^{(0.8)}$ and $T_s^{(0.95)}$ are computed from the scenario of T_s by using Definition (8.7).

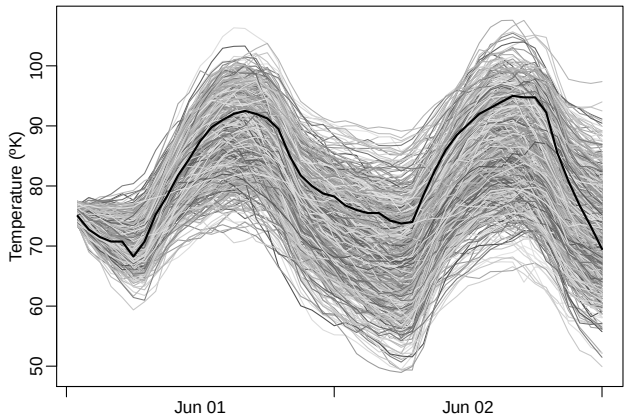


Figure 8.6.: 800 temperature scenarios (T_s) generated for June 1, 2011 to June 2, 2011. The line in black depicts the observed temperature.

We chose to generate 800 scenarios of the temperatures. This has proven to be fast enough and to provide a good overview of what is possible. The generation of the temperature scenarios includes randomness into our method. This partly explains the slight differences into the performance reported in Table 8.1. Figure 8.6 plots the 800 scenarios simulated for June 1, 2011 to June 2, 2011 together with the observed temperature.

Probabilistic forecasting of the electric load (knowing the temperature in advance) Now we model the distribution of the load as if we had access to the true values of the temperature in advance. Once again, it is performed by fitting quantGAM described in Section 8.2.3. We train a

separate model for each hour of the day $h = 1, \ldots, 24$ (i.e, the times series is partitioned into 24 times series that we investigate independently). The model (Step 1.a. in Section 8.2.3) is defined as:

$$Y_t = f_1(T_t, t) + f_2(T_t^{(0.8)}) + f_3(T_t^{(0.95)}) + f_4(Toy_t) + f_5(t) + h\big(DayType_t\big) + \varepsilon_t\,. \tag{8.14}$$

We recall that $f_1, f_2, \ldots$ are smooth cubic splines to be estimated by GAM and that h a real function. Then, we move to Step 2 of quantGAM by performing linear quantile regressions for all quantiles $\tau \in \{0.01, \ldots, 0.99\}$ (exceptionally, we skip Step 1.b.). Thus, we fit 2 376 (= 24 hours $\times$ 99 quantiles) linear quantile regressions.

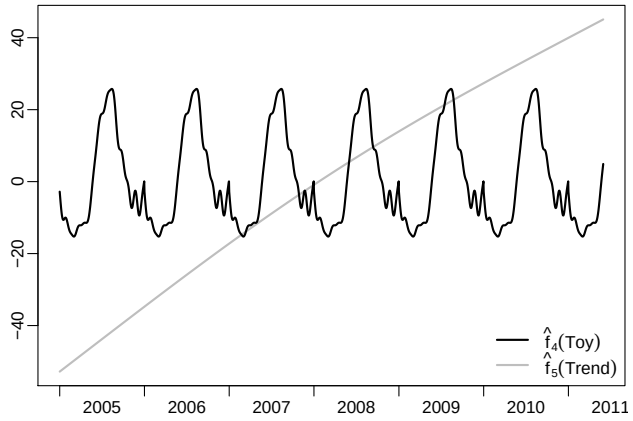


Figure 8.7.: Non-linear effects of the annual seasonality $\widehat{f_4}(Toy_t)$ and of the trend $\widehat{f_5}(t)$ on the load estimated by Model (8.14)

Probabilistic forecasting of the electric load (operational forecast) However, to produce forecasts, once again we do not have access to the real values of the temperatures (T_t, $T_t^{(0.8)}$, and $T_t^{(0.95)}$). We need to substitute them with the 800 sampled scenarios of the temperatures. We obtain a probabilistic forecast of the load for each scenario and we form our prediction by averaging the forecasted distributions over all scenarios.

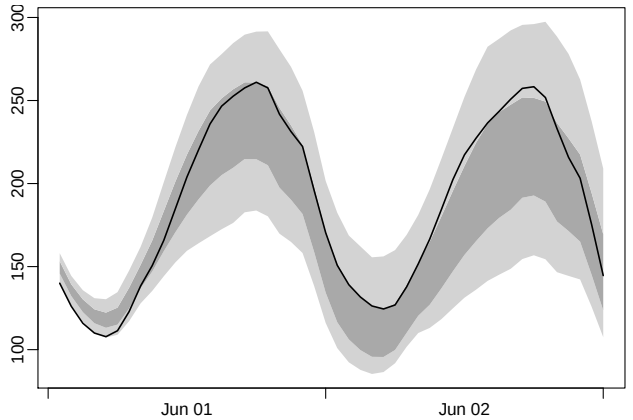


Figure 8.8.: Forecasted distribution of the load from June 1, 2011 to June 2, 2011 by averaging the distributions obtained for each individual scenario of the temperature (T_t, $T_t^{(0.8)}$, $T_t^{(0.95)}$).

This forecasted distribution takes into account both the uncertainty with regard to the weather

(captured by the temperature scenarios) and the uncertainty about the model (captured by quantGAM). Figure 8.8 displays the forecasted distribution for June 1, and June 2, 2011. We see that the predicted confidence intervals are tight a few hours ahead (which was not the case for the medium term forecasts displayed in Figure 8.5) and is widening with the horizon of prediction.

8.3.4. Final forecasts and results

In the end we form our forecasts for the next month by concatenating the short term forecasts (from 1 hour to 48 hours ahead) with the medium term forecasts (from 49 hours ahead).

Figure 8.9 shows the probabilistic forecasts obtained for the month of June 2011. Figure 8.10 plots the percentage of time the observed electricity consumption Y_t is smaller than the predicted quantiles $\widehat{q}_{\tau,t}$ according to $\tau \in [0,1]$ during the year 2011. The closer the curve is from the identity function the better. We remark that the method overestimated the consumption for most quantiles. This can partly be explained by an unexpected drop of consumptions (e.g., end of August or mid-September).

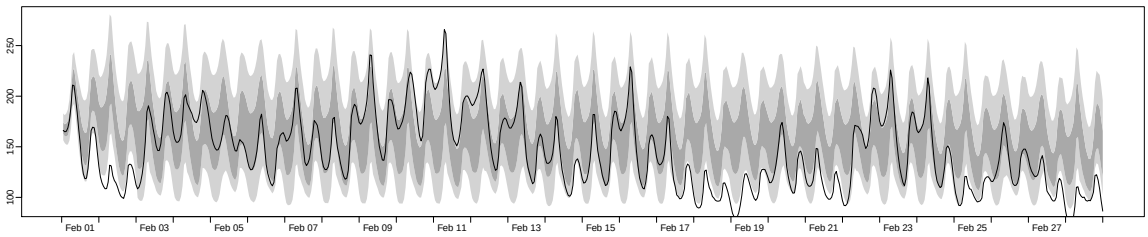
Table 8.1 reports for each month of the year 2011 the performance obtained by the benchmark, the team Tololo, and the described methodology (used from March to December 2011 during the competition).

Month	Benchmark	Tololo	quantGAM
Jan.	18.74	10.44	10.62
Feb.	22.76	12.52	9.83
Mar.	13.22	8.27	7.90
Apr.	8.36	4.42	4.19
May.	10.92	5.90	5.87
Jun.	16.99	6.19	5.80
Jul.	13.40	7.32	8.13
Aug.	17.32	10.80	10.73
Sep.	13.84	5.45	5.46
Oct.	6.42	3.96	3.98
Nov.	10.94	6.32	6.33
Dec.	34.07	8.48	8.51

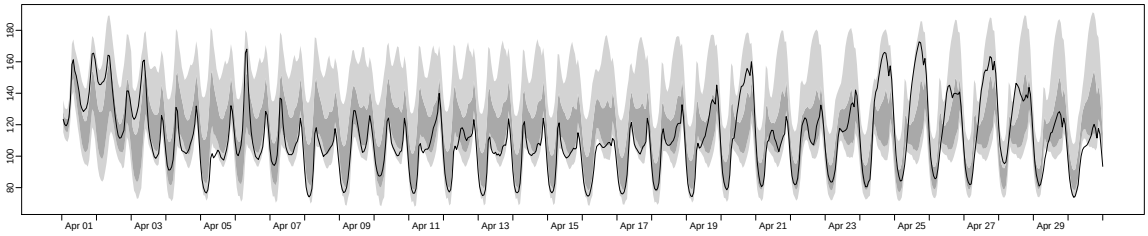
Table 8.1.: Performance of the benchmark, the team Tololo, and the method described in this paper (quantGAM).

8.4. Probabilistic electricity price forecasting by quantGAM

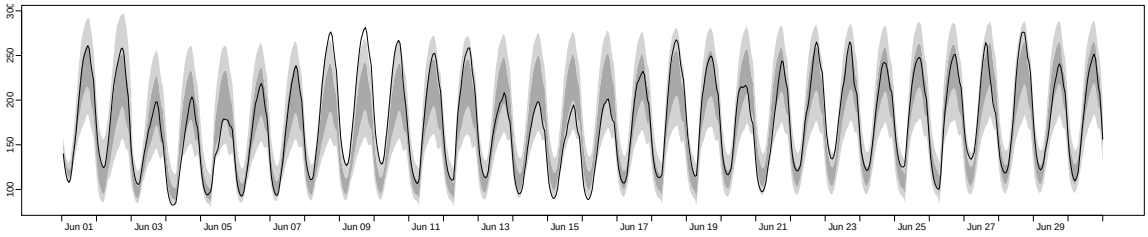
We turn to the problem of electricity price forecasting. We consider the data available for the Probabilistic Electricity Price Forecasting Track of the GEFCom2014 competition. It contains hourly observations of the historical prices from January 1, 2011 to December 17, 2013, together with hourly historical zonal and system load forecasts. Our goal is to forecast one day ahead (i.e., the next 24 hours) the τ-quantiles of the electricity prices. Our performance is measured by the average pinball scored defined in (8.6). We describe in this section a methodology based on quantGAM. Similarly to the electricity consumption, the electricity price heavily depends on the hour of the day and the models are hourly performed by partitioning the data into 24 independent time series.



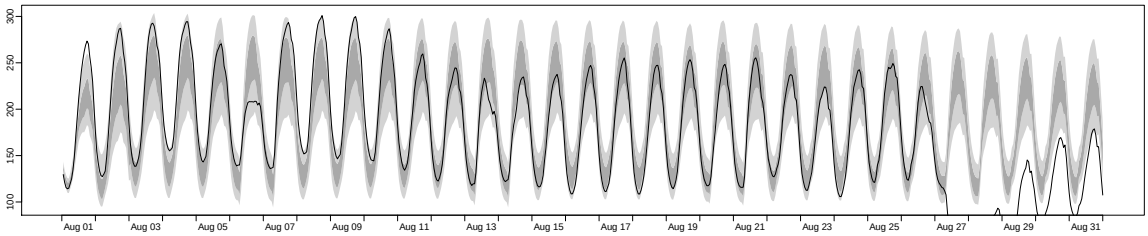
(a) February



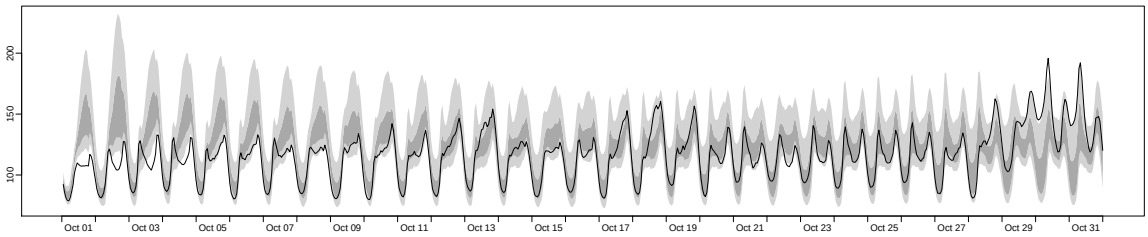
(b) April



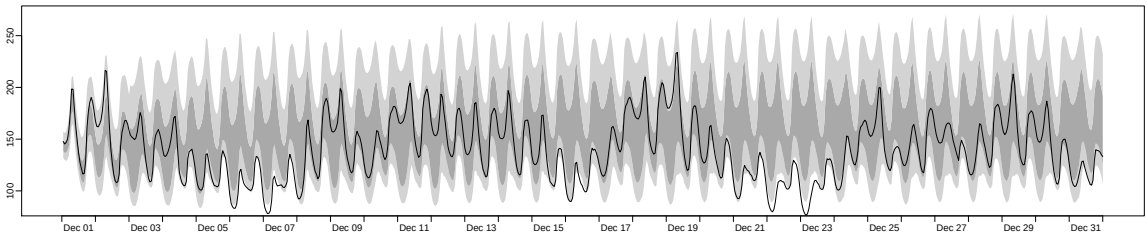
(c) June



(d) August



(e) October



(f) December

Figure 8.9.: Forecasted distribution of the electric load for several months of 2011.

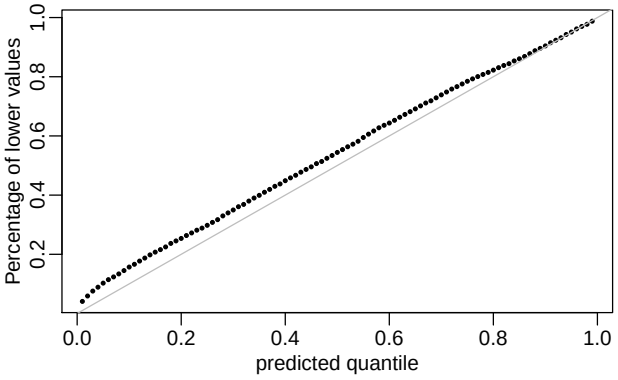


Figure 8.10.: Percentage of observed electric load values Y_t under the predicted quantiles $\widehat{q}_{\tau,t}$ during the year 2011 which has been forecasted month by month.

8.4.1. Working on the data

In order to build our forecasting method, we defined below several covariates:

- P_t is the price at time t.
- $P_t^{(\text{last})}$ is the most recent lagged price available for the forecast at time t.
- FZL_t and FTL_t are respectively the forecasted zonal and total load.
- $FZL_t^{(\gamma)}$ and $FTL_t^{(\gamma)}$ are exponential smoothing of the forecasted zonal (resp. total) load with parameter $\gamma \in [0,1]$ (see Equation (8.7) for a definition).
- $X_t^{(\max)}$ (respectively $X_t^{(\min)}$ and $X_t^{(\text{mean})}$) is the maximum (respectively minimum and mean) of the variable X_t (such as the price P_t or the forecasted zonal load FZL_t) during the day corresponding to observation t.

In the sequel, we will mostly use the logarithms of the above variables.

8.4.2. Probabilistic forecasting of the maximal price

In order to predict the electricity price of the following day, we aim at using the maximal price $P_t^{(\max)}$ of the day to be predicted as an exogenous variable. To do so we should first provide a forecast of $P_t^{(\max)}$. We describe below how to do so by using quantGAM:

1.a. We estimate the linear and non-linear effects that impact the expected maximal price by fitting the generalized additive model:

$$\begin{aligned}\log(P_t^{(\max)}) = {} & \alpha_1 \log(P_{t-24}^{(\max)}) + \alpha_2 \log(P_{t-48}^{(\max)}) + \alpha_3 \log(P_{t-24}^{(\text{mean})}) + \alpha_4 \log(P_{t-48}^{(\text{mean})}) \\ & + f_1\big(\log(FTL_{t-24}^{(\text{mean})})\big) + f_2\big(\log(FTL_t^{(\text{mean})})\big) + f_3\big(\log(FZL_t^{(\max)})\big) + f_4\big(FZL_{t-24}^{(\max)}\big) + \varepsilon_t\,,\end{aligned} \tag{8.15}$$

where α_i are linear coefficients and f_j are smooth cubic splines to be estimated by $\widehat{\alpha}_i$ and $\widehat{f}_j$.

2. We perform a linear quantile regression to forecast the quantiles of the maximal price by using the vector of covariates

$$\boldsymbol{Z}_t = \Big(\widehat{\alpha}_1 \log(P_{t-24}^{(\max)}),\ \widehat{\alpha}_2 \log(P_{t-48}^{(\max)}), \widehat{\alpha}_3 \log(P_{t-24}^{(\text{mean})}),\ \widehat{\alpha}_4 \log(P_{t-48}^{(\text{mean})}),$$

$$\widehat{f}_1\big(\log(FTL_{t-24}^{(\text{mean})})\big),\ \widehat{f}_2\big(\log(FTL_t^{(\text{mean})})\big), \widehat{f}_3\big(\log(FZL_t^{(\text{max})})\big),\ \widehat{f}_4\big(FZL_{t-24}^{(\text{max})}\big)\Big).$$

Figure 8.11 plots the forecasted distribution of the maximal price from June 16, 2013 to June 25, 2013.

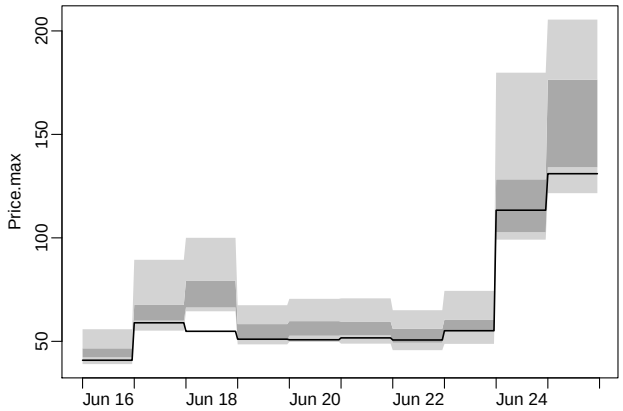


Figure 8.11.: One day ahead forecasted distribution of the maximal price from June 16, 2011 to June 25, 2013.

8.4.3. Probabilistic forecasting of the electricity price

We are now ready to forecast the electricity price distribution. We remarked that it was not necessary to consider non-linear effects here. Therefore, we only estimated $\log(P_t)$ by fitting linear quantile regression (see Section 8.2.2) without performing a first step to estimate non-linear effects. We used the following vector of covariates:

$$\boldsymbol{Z}_t = \Big(\log(P_t^{(\text{last})}),\ \log(P_t^{(\text{max})}),\ \log(P_{t-24}),\ \log(P_{t-48}), \log(P_{t-168}),\ \log(P_{t-24}^{(\text{min})}),\ DayType_t,$$
$$FZL_t^{(0.95)},\ FTL_t^{(0.95)},\ FZL_t^{(0.8)},\ FTL_t^{(0.8)}\Big).$$

All the covariates in $\boldsymbol{Z}_t$ are available 24 hours in advance except $\log(P_t^{(\text{max})})$. Thus, to address this problem, we average the forecasted distribution of the price over the forecasted distribution of the maximal price performed in Section 8.4.2. The method is similar to the one used for the temperature forecasts at the end of Section 8.3.2. Figure 8.12 displays the forecasted distribution and the observed prices for several days.

8.4.4. Results

We present in Table 8.2 and in Figure 8.13 the practical performance obtained by the method described in this section, denoted quantGAM, on the days evaluated by the competition. Table 8.2 also summarizes the results obtained by the team Tololo during the competition and by the two other methodologies quantMixt and quantGLM that will be detailed in the next sections. We note that quantGAM is especially robust to price spikes (which occur on July 18 and July 19, 2013) and exhibits good performance over all tested days.

During the competition, we started by using from Task 2 to Task 8 several versions of quantMixt, then we used from Task 9 to Task 12 versions of quantGAM. For the last three tasks, that correspond

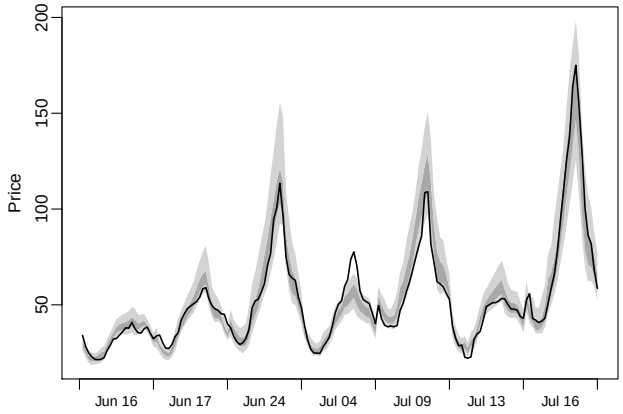


Figure 8.12.: One day ahead forecasted distribution of the electricity price for days corresponding to tasks 1 to 7.

to days in winter, we adopted quantGLM that is especially designed for winter. The results obtained by Tololo and by the three methods are different. This is mostly due to the fact that during the competition we did not use the same versions of the methods described in this paper. Indeed we constantly changed our methodology during the competition. Due to space constraint we cannot get into the details in this paper.

Task	Date	Benchmark	Tololo	quantGAM	quantMixt	quantGLM
1	Jun. 06	3.13	XX	**0.72**	0.85	1.87
2	Jun. 17	0.68	1.06	1.15	1.37	**0.71**
3	Jun. 24	8.13	1.91	**1.31**	1.58	3.05
4	Jul. 04	4.03	1.71	2.06	**1.27**	1.59
5	Jul. 09	7.97	1.45	2.67	3.31	**1.57**
6	Jul. 13	5.70	1.10	**0.99**	1.20	1.18
7	Jul. 16	12.15	2.01	**2.23**	2.28	5.02
8	Jul. 18	38.35	9.15	**5.13**	7.90	11.72
9	Jul. 19	44.23	4.68	**4.80**	6.45	13.27
10	Jul. 20	18.22	1.59	**1.90**	2.35	2.80
11	Jul. 24	31.57	0.75	**0.75**	1.78	1.42
12	Jul. 25	42.95	2.46	2.30	**0.84**	2.12
13	Dec. 06	2.86	2.96	**0.82**	1.03	0.86
14	Dec. 07	3.20	1.35	3.63	3.23	**3.22**
15	Dec. 17	22.38	3.56	3.83	4.26	**2.87**
Global		16.36	2.55	**2.40**	2.78	3.67

Table 8.2.: Performance of quantGAM (Section 8.4), quantMixt (Section 8.5), and quantGLM (Section 8.6) together with the performance obtained by the benchmark proposed by the competition organizers and by the team Tololo on the tested days in 2013 of the electricity price competition. For each task the best of the results is highlighted in bold. The average performance over all days (other than June 06) is also reported.

In the two remaining sections, we present the other methodologies that we considered for probabilistic electricity price forecasting. Although their results seem to be worse than quantGAM in Table 8.2, we think that they can be largely improved in the future.

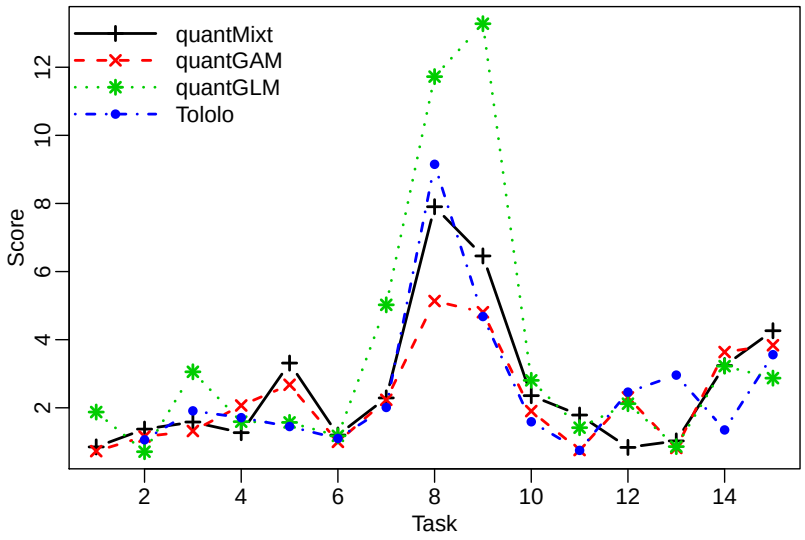


Figure 8.13.: Performance of the different methods described in this article on the Tasks of the electricity price competition.

8.5. Probabilistic electricity price forecasting by combining individual predictors

We present in this section a second approach that we used during the competition to form probabilistic forecast of the electricity price. We consider a methodology inspired from Nowotarski and Weron [118]. The idea is a two steps approach: first, we build a set of individual predictors that aims at predicting the mean of the electricity price; second, we combine these individual predictors so as to obtain a forecast of the quantiles by combining the individual predictors.

The training set is partitioned into two pieces. A *mixing set* that consists of the last 30 days of the training set (i.e., the last $30 \times 24 = 720$ hours) is used to learn the best combination of individual forecasters. The rest of the training set is the *fitting set* (data older than one month) which serves to fit the individual models.

8.5.1. Individual Predictors

We consider 13 individual predictors that were chosen because they exhibit various behaviors with the idea that the combining algorithm will be able to catch the best of each. We use the notations defined in Section 8.4.1.

A first class of individual forecasters are well-known predictors that have been proven to perform well in electricity price forecasting. They include several autoregressive models, spike preprocessed autoregressive models, and threshold autoregressive models described more in details in Nowotarski and Weron [118]:

1. An autoregressive model (`AR`) defined as:

$$\log(P_t) = \alpha_1 \log(P_{t-24}) + \alpha_2 \log(P_{t-48}) + \alpha_3 \log(P_{t-168}) + \alpha_4 \log(P_{t-24}^{(\min)}) + h(DayType_t) + \varepsilon_t \,,$$

where ε_t (as in the other models) are i.i.d. centered Gaussian noise, α_i are linear coefficient to be estimated, and h is a real function to be estimated as in Equation (8.10).

2. An autoregressive model with forecasted electric loads as additional covariates (`ARX`). It is defined as:

$$\log(P_t) = \alpha_1 \log(P_{t-24}) + \alpha_2 \log(P_{t-48}) + \alpha_3 \log(P_{t-168}) + \alpha_4 \log(P_{t-24}^{(\min)}) + \alpha_5 \log(FTL_t) + \alpha_6 \log(FZL_t) + h(DayType_t) + \varepsilon_t \,.$$

3. A threshold autoregressive model `TAR` defined as an extension of `AR` to two regimes depending on the variation of the mean price between a day and eight days ago.

4. `TARX` the extension of `ARX` to the two regimes model.

5. Spike pre-processed autoregressive model (`PAR`): the idea is to pre-process the price data by removing large spike (see the "damping scheme" presented in Weron and Misiorek 145): for all $P_t > M$ set $\widetilde{P}_t = M + M \log_{10}(P_t/M)$ where M is a fixed parameter set to the mean price plus three standard deviations. Then `AR` is fitted by substituting the prices $P_t, P_{t-24}, \ldots$ with the pre-processed prices $\widetilde{P}_t, \widetilde{P}_{t-24}, \ldots$.

6. `PARX` similar to `PAR`, but `ARX` is fitted with pre-processed prices.

The seven remaining individual forecasters are designed by considering several regression methods, several subsets of covariates, and different sets of fitting data. More precisely we list them hereafter:

7. A linear regression (function `lm` in `R`) fitted with model:

$$\log(P_t) = \alpha_1 \log(P_{t-24}) + \alpha_2 \log(P_{t-48}) + \alpha_3 \log(P_{t-168}) + \alpha_4 \log(P_t^{(\max)}) + \alpha_5 FZL_t^{(0.95)} + \alpha_6 FTL_t^{(0.95)} + \alpha_7 FZL_t^{(0.8)} + \alpha_8 FTL_t^{(0.8)} + h(DayType_t) + \varepsilon_t$$

   In order to produce a forecast, $P_t^{(\max)}$ is substituted with its forecast performed by a generalized additive model with Equation (8.15). This substitution is also performed for the next individual predictors.

8. A linear regression (`lm`) fitted as follows:

$$\log(P_t) = \alpha_1 \log(P_t^{(\text{last})}) + \alpha_2 \log(P_{t-24}) + \alpha_3 \log(P_t^{(\max)}) + \alpha_4 FTL_t + \alpha_5 FZL_t + h(DayType_t) + \varepsilon_t \,.$$

9. A generalized additive model (see Section 8.2.1 for details) fitted with:

$$\log(P_t) = f_1(ToY_t) + f_2\big(\log(P_{t-24})\big) + f_3\big(\log(P_t^{(\max)})\big) + f_4\big(\log(P_{t-24}^{(\text{mean})})\big) + h(DayType_t) + \varepsilon_t \,.$$

   We recall that f_1 is a cubic cyclic regression spline, f_2, f_3, and f_4 are cubic regression splines, and h is some real function as in Equation (8.10).

10. A generalized additive model fitted with:

$$\log(P_t) = f_1\big(\log(P_{t-24})\big) + f_2\big(\log(P_t^{(\max)})\big) + f_3\big(\log(P_{t-24}^{(\mathrm{mean})})\big) + f_4(FTL_t) + f_5(FZL_t) + f_6(ToY_t) + h(DayType_t) + \varepsilon_t\,.$$

    This model was only trained on Summer data (i.e., from June to August) if the day to be predicted is in Summer itself. The idea of creating specialized forecasters in order to obtain more diversity was investigated in Gaillard and Goude [75] in the context of electricity consumption forecasting.

Then we considered two forecasters using random forests regression. Random forests were introduced by Breiman [35] and are available in the `R`-package `randomforest`[§]. They are a powerful ensemble method that builds a large number of random regression binary trees before aggregating them, so as to improve their prediction accuracy. The process of fitting a random forests model and using it to produce a prediction performs three steps. First it performs bagging: it creates multiple bootstrap samples of the training set by sampling from the training set n observations uniformly and with replacement, where n is the size of the training set. Second it builds random prediction trees: for each bootstrap sample, it builds a corresponding binary decision tree by partitioning the covariate space recursively in a dyadic fashion. The trees are extended very deeply in practice and have high variance and low bias. Third it forms a prediction, by averaging the predictions of the individual regression trees. The two random forests individual predictors are:

11. Random forests regression fitted with all covariates described in the previous models.

12. Random forests regression fitted with the same covariates used by the individual predictor 8.

Our last predictor is trained by using gradient boosted methods. Gradient boosted methods were introduced by Friedman [72] and are implemented in the `R`-package `gbm`. It is another ensemble method. By contrast to random forests the basis forecasters are a class of weak regression methods (e.g., decision trees with very limited maximal depth) and are built sequentially by trying to reduce the bias of the combined predictor. We ran the experiments by using the default parameters of the package `gbm` and by optimizing the number of trees (basis forecasters) on the out-of-bag error using the function `gbm.perf`. The last predictor is:

13. Gradient boosting machine fitted with the same covariates used by the individual predictor 8.

In the end we have 13 individual forecasters of the price. The exact form and number of individual forecasters changed over time and over our submissions. For the sake of simplicity, we only presented in this paper few predictors among those we tested. This partly explains why the results obtained by the methods presented in this paper are different from those obtained during the competition.

8.5.2. Combining forecasts

Once the forecasters have been designed and fitted on the *fitting set*, we learn on the *mixing set* (the last thirty days of available data, that we denote by E) how to combine them in order to provide probabilistic forecasts. To do so, we consider the setting of online robust aggregation of predictors (see the monograph of Cesa-Bianchi and Lugosi [43] for a nice overview) which was proven to perform well on electricity load forecasting, see Devaine et al. [60].

[§] we run all experiments with default parameters of the package

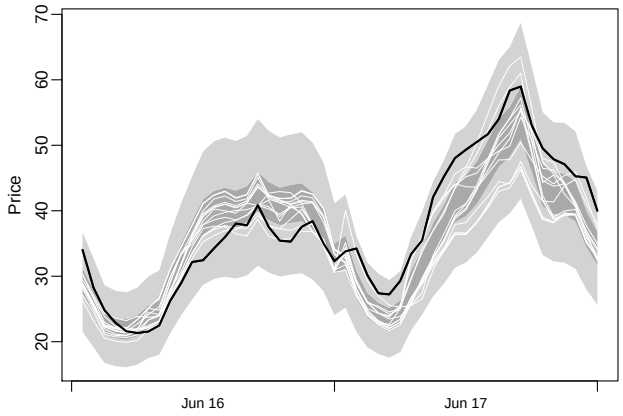


Figure 8.14.: One day ahead forecasted distribution of the electricity price for days corresponding to tasks 1 and 2. The black line depicts the observed electricity price and the white lines plots the individual forecasts.

We consider a version of the ML-Poly forecaster introduced in Gaillard et al. [77] because it is fully adaptive and was proven to exhibit good performance on the electric load signal (see Gaillard and Goude 75). We detail it now. Let us denote for each time t by $\boldsymbol{x}_t = (x_{t,1}, \dots, x_{t,K}) \in \mathbb{R}_+^K$ the forecasts of P_t produced by the $K = 13$ individual forecasters described in Section 8.5.1. Our algorithm is described as Algorithm 14. For each quantile $\tau \in (0,1)$, and for $\beta > 0$, it consists in performing a kind of stochastic gradient descent so as to minimize in $\boldsymbol{\theta} \in \mathcal{B}_1(\beta) \triangleq \{\boldsymbol{\theta} \in \mathbb{R}^K, \|\boldsymbol{\theta}\|_1 \leqslant \beta\}$ a regularized version of the average pinball loss over the mixing set: $1/(\operatorname{card} E) \sum_{t \in E} \rho_\tau(P_t - \boldsymbol{\theta}^\top \boldsymbol{x}_t)$. Here, $\operatorname{card} E$ denotes the number of observations in the *mixing set* (i.e., $24 \times 30 = 720$ hours), and ρ_τ is the pinball loss defined in Section (8.2.2).

More precisely, we set $m \in \mathbb{N}^*$ a number of optimization steps. At each instance $i \in \{1, \dots, m\}$, Algorithm 14 samples an observation t_i uniformly in the *mixing set* E (Step 1). Then it updates (Step 3) a weight-vector $\widehat{\boldsymbol{p}}_i$ to $\widehat{\boldsymbol{p}}_{i+1}$ in the simplex $\Delta_{2K} \triangleq \{x \in \mathbb{R}_+^{2K} \ : \ \sum_k x_k = 1\}$ in order to improve the τ-quantile prediction of P_{t_i} by the weighted average $\widehat{P}_i = \sum_{k=1}^{2K} \widehat{p}_{i,k} \widetilde{x}_{t_i,k}$, where for all times $t \geqslant 1$,

$$\widetilde{\boldsymbol{x}}_t \triangleq \beta \left(- x_{t,1}, \, x_{t,1}, \dots, -x_{t,K}, \, x_{t,K} \right).$$

Doing so, it can be guaranteed (see Gaillard et al. 77, Cesa-Bianchi and Lugosi 43) under i.i.d. assumption of the data that the risk of the average weight vector $\bar{\boldsymbol{p}}_\tau = (1/m) \sum_{i=1}^m \widehat{\boldsymbol{p}}_i$ is close to the optimal risk $\min_{\boldsymbol{p} \in \Delta_{2K}} \mathbb{E}\big[\rho_\tau\big(P_t - \boldsymbol{p}^\top \widetilde{x}_t\big) | x_t\big]$. The returned combination vector is finally $\widehat{\boldsymbol{\theta}}_\tau$ defined in (8.16) because $\widehat{\boldsymbol{\theta}}_\tau^\top \boldsymbol{x}_t = \bar{\boldsymbol{p}}_\tau^\top \widetilde{\boldsymbol{x}}_t$. The idea of considering convex combinations of $\widetilde{\boldsymbol{x}}_t$ in Δ_{2K} in order to perform combinations of $\boldsymbol{x}_t$ in the linear ball $\mathcal{B}_1(\beta)$ dates back to Kivinen and Warmuth [97].

In order to produce a forecast of the τ-quantile of P_t, we run over the *mixing set* E Algorithm 14 with parameters $m = 5000$ and $\beta = 2$ and we predict $\widehat{q}_{t,\tau} = \sum_{k=1}^K \widehat{\theta}_{\tau,k} x_{t,k}$, where $\boldsymbol{x}_t$ are the forecasts of the individual forecasters defined in Section 8.5.1. We call quantMixt this method.

8.5.3. Results

Figure 8.14 plots the forecasted price distribution together with the observed price values and the individual forecasts for June 16 and June 17, 2013. The performance of this method is reported

Algorithm 14: The averaged linear ML-Poly weighted average forecaster for quantile prediction

Input: $\tau \in (0,1)$, $\beta > 0$, $m \in \mathbb{N}^*$
Initialize: $\widehat{\boldsymbol{p}}_1 = \big(1/(2K), \ldots, 1/(2K)\big) \in \Delta_{2K}$

for each step $i = 1, 2 \ldots, m$ **do**
1. sample t_i uniformly in E
2. define for all $k = \{1, \ldots, 2K\}$

$$\widetilde{x}_{t_i,k} = \Big(\mathbb{1}_{\{k \text{ is even}\}} - \mathbb{1}_{\{k \text{ is odd}\}}\Big)\, \beta\, x_{t_i,\lceil k/2 \rceil}$$

and the learning rate

$$\eta_{i,k} = \left(1 + \sum_{s=1}^{i} \Big(\ell_s(\widehat{P}_s) - \ell_s(\widetilde{x}_{t_s,k})\Big)^2\right)^{-1}$$

where $\ell_s : x \mapsto \Big(\mathbb{1}_{\{P_{t_s} < \widehat{P}_s\}} - \tau\Big)x$ and $\widehat{P}_s = \widehat{\boldsymbol{p}}_s^{\top} \widetilde{\boldsymbol{x}}_{t_s}$.
3. form the mixture $\widehat{\boldsymbol{p}}_{i+1} \in \Delta_{2K}$ component-wise by

$$\widehat{p}_{i+1,k} = \frac{\eta_{s,k} \Big(\sum_{s=1}^{i} \ell_s(\widehat{P}_s) - \ell_s(\widetilde{x}_{t_s,k})\Big)_+}{\sum_{j=1}^{2K} \eta_{s,j} \Big(\sum_{s=1}^{i} \ell_s(\widehat{P}_s) - \ell_s(\widetilde{x}_{t_s,j})\Big)_+} \in [0,1]$$

where $x_+ \triangleq \max\{x, 0\}$.
end for

Return the weight vector $\widehat{\boldsymbol{\theta}}_\tau \in [-\beta, \beta]^K$ component-wise defined as

$$\widehat{\theta}_{\tau,k} = \frac{1}{m} \sum_{i=1}^{m} \beta\ (p_{i,2k} - p_{i,2k+1}) \tag{8.16}$$

in Table 8.2. The performance is good except on July 18 and July 19, 2013 that correspond to spikes in the electricity price. We remarked that this was partly due to bad predictive performance of individual forecasts which did not detect spikes and partly due to the relative small size of the *mixing set* where few electricity spikes are observed. This suggests two possible directions for future research to improve the method. The first avenue would be to improve the set of individual predictors. The second one would be to consider sequential versions of the individual predictors in order to learn the combinations over the whole training set.

8.6. Kernel based quantile regression with Lasso penalty

Here we present the last method that we implemented in this competition for electricity price forecasting. It was motivated by the fact that after twelve weeks of competition we have generated a lot of covariates from the three original ones: total and zonal loads and prices, and we want to take advantage of that in an automatic fashion. Here is the list of those transformations, where X_t could be either the price P_t, the zonal load FZL_t or the total load FTL_t at time t:

- lagging: we consider the following lagged variables X_{t-24}, X_{t-48}, X_{t-168}, X_{t-336}, and $X_t^{(\text{last})}$ the last available observation at the time of the forecast;
- log and log log transforms: $\log(x_t)$, $\log\big(\log(x_t)\big)$;

- spike preprocessing of the price (see models TARX and ARX of Section 8.5.1).
- mean, max and min of the day corresponding to observation t: $X_t^{(\min)}$, $X_t^{(\max)}$, and $X_t^{(\text{mean})}$ (see Section 8.4.1);
- exponential smoothing with parameters $\gamma \in \{0.8, 0.95\}$ defined by induction as in Equation (8.7).

All these single transformations could be coupled: for example we could take the log of a lagged covariates or the maximum of a smoothed covariate etc. In the end by adding also the calendar variables $DayType_t$ we obtain $d \triangleq 192$ covariates that are candidates to enter into a linear quantile regression. These covariates are potentially highly correlated. In the following we denote by $\boldsymbol{X}_t \in \mathbb{R}^d$ the vector that contains all transformations of covariates. Our idea is to select among those 192 covariates the ones which produce good forecasting performance for each quantile. To this end we propose to use a ℓ_1 selection procedure as presented in Tibshirani [136] together with a kernel regularized regression. This approach has already been studied theoretically in Belloni and Chernozhukov [23] but we did not find any existing `R` package. To benefit from the nice performance of the `R` package `glmnet` (see Friedman et al. 74) we propose a two steps approach detailed hereafter. First we fit a single ℓ_1-regression model on the mean by using `glmnet` with the Lasso penalty (cf. Step 1). Then we fit a quantile regression model on the residuals of this model by using a weighted version of the previous algorithm with weights corresponding to a Gaussian kernel centered around each quantile (cf. Step 2).

- Step 1: we estimate the mean price by a sparse linear combination of the covariates. To do so, we solve the optimization problem:

$$\widehat{\boldsymbol{\beta}} \in \underset{\boldsymbol{\beta}\in\mathbb{R}^d}{\arg\min} \left\{ \sum_{t=1}^{n} \left(P_t - \boldsymbol{X}_t^\top \boldsymbol{\beta}\right)^2 + \lambda \parallel \boldsymbol{\beta} \parallel_1 \right\}$$

  where $d = 192$ denotes the number of covariates, n is the number of observations in the training set, $\boldsymbol{X}_t \in \mathbb{R}^d$ is the vector of covariates, and $\lambda > 0$ is a parameter that penalizes large models. It has to be optimized. We obtain the estimate $\widehat{\boldsymbol{\beta}} \in \mathbb{R}^d$ and the residual signal $\widehat{\varepsilon}_t \triangleq P_t - \boldsymbol{X}_t^\top \widehat{\boldsymbol{\beta}}$.
- Step 2: for each quantile $\tau \in (0,1)$, we estimate a correction to add to the mean estimates $\boldsymbol{X}_t^\top \widehat{\boldsymbol{\beta}}$ in order to estimate the quantile of level τ. To achieve this, we first compute e_τ the empirical quantile of the sequence $(\widehat{\varepsilon}_t)_{t=1,\dots,n}$ and the weights for all times $t = 1, \dots, n$:

$$w_{\tau,t} = \frac{\exp\left(-(\widehat{\varepsilon}_t - e_\tau)^2/h\right)}{\sum_{t=1}^{n} \exp\left(-(\widehat{\varepsilon}_t - e_\tau)^2/h\right)}$$

  where $h > 0$ is a window parameter to be optimized. Then we proceed to the optimization problem:

$$\widehat{\boldsymbol{\beta}}_\tau \in \underset{\boldsymbol{\beta}_\tau\in\mathbb{R}^d}{\arg\min} \left\{ \sum_{t=1}^{n} w_{\tau,t} \left(\widehat{\varepsilon}_t - \boldsymbol{X}_t^\top \boldsymbol{\beta}_\tau\right)^2 + \lambda \left\| \boldsymbol{\beta}_\tau \right\|_1 \right\}.$$

- Step 3: the final quantile forecast $\widehat{q}_{\tau,t}$ of the price at time t is then obtained by

$$\widehat{q}_{\tau,t} \triangleq \underbrace{\boldsymbol{X}_t^\top \widehat{\boldsymbol{\beta}}}_{\text{mean estimate}} + \underbrace{\boldsymbol{X}_t^\top \widehat{\boldsymbol{\beta}}_\tau}_{\tau\text{-quantile correction}} .$$

In Steps 1 and 2, the optimal penalization parameter λ is found by using 10-fold cross-validation implemented in the `glmnet` package. The window parameter h has been optimized on a grid over the last winter period (because we used this method at the end of the competition in order to predict days in winter). As the process is time consuming, Step 2 is actually only performed for quantiles $\tau \in \{0.01, 0.99\} \cup \{0.1, 0.2, \ldots, 0.9\}$. The other quantiles are formed by linear interpolation between those quantiles.

8.6.1. Results

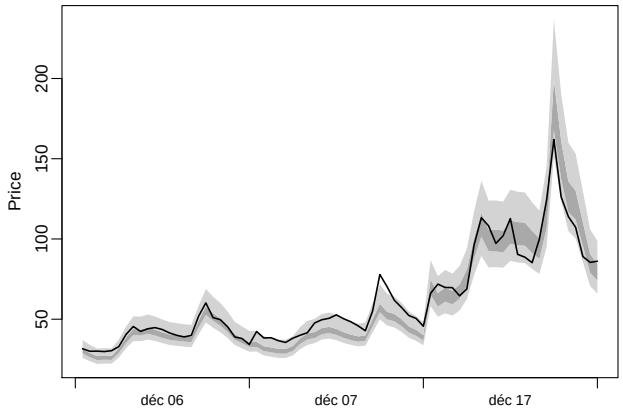


Figure 8.15.: One day ahead forecasted distribution by quantGLM of the electricity price for days corresponding to tasks 13, 14, and 15.

The performance of this method (denoted quantGLM) is reported in Table 8.2. Note that the parameter of the method were optimized on winter which may explains its bad performance of several days in summer. Another critical point is that the covariate selection is mostly done at Step 1 inducing potential bias for some quantile that could not be corrected at Step 2. We leave for future research a unified algorithm for covariate selection for each quantile avoiding these kind of bias as well as the optimization of the method in summer.

8.7. Conclusion

In the end, we proposed for the probabilistic price forecasting task three methods (quantGAM, quantMixt, and quantGLM) that achieve good performance (cf. Table 8.2). We think that they can be largely improved in the future. Figure 8.16 plots the percentage of time the observed electricity price P_t is smaller than the quantiles $\widehat{q}_{\tau,t}$ predicted by these methods according to $\tau \in (0, 1)$. The closer the curve is from the identity function the better. While quantGAM and quantMixt do not seem to be biased, quantGLM used to overestimate low quantiles and underestimate high quantiles. Understanding this behavior may yield a possible improvement of quantGLM in the future.

For the probabilistic load forecasting task we converged to a single method (quantGAM) based on

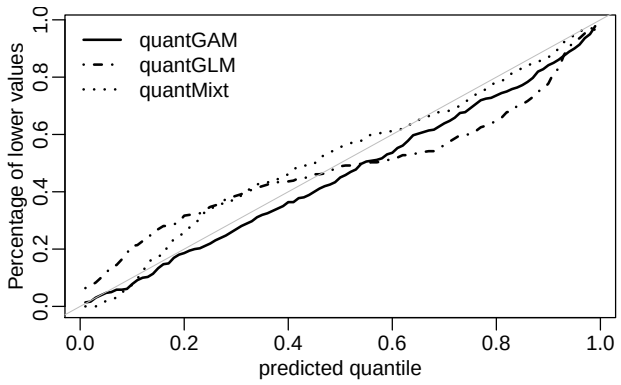


Figure 8.16.: Percentage of observed electric load values P_t under the predicted quantiles $\widehat{q}_{\tau,t}$.

generalized additive models. In addition to perform well on the load data set (see Table 8.1), it is easy to use and offers a good interpretation of the effects that impact the electricity demand.

IV

Performance of sequential robust aggregation on real-world data

Introduction

The key message of this last part is that the setting of online robust aggregation is universal. Prediction with expert advice does not perform well on specific data set only. It can be applied to multiple kinds of data with great forecasting performance.

In Chapter 4 and 8 we already considered two energy data sets. Chapter 4 shows that combining forecasts largely improves the prediction accuracy of electric load upon the base forecasting methods. Chapter 8 gets promising results for one day ahead probabilistic forecasting of the electric price through aggregation of expert advice.

This part is composed of two other empirical studies carried out for EDF. Chapter 9 addresses the task of forecasting at different horizons of time (ranging from one half hour ahead to 72 hours ahead). It builds a methodology which is successfully applied to two real world data sets: a heat load data set and an electric load data set. In the end, Chapter 10 investigates a setting (which grows in interest with the emergence of smart grids) where the global electric load (y_t) to be predicted can be decomposed into low-level signals (each corresponding to some sub-group of consumers). The experts do not output forecasts of the global consumption y_t but form forecasts of low-level consumptions. We show that linear online aggregation of expert advice is still a good option for our data in order to improve the prediction accuracy of the global load.

9

Heat load and electricity load multi-horizon forecasting by online robust aggregation

In this empirical study, we address the task of forecasting the future heat demand on an urban area of a city at different horizons of time going from one hour ahead to 72 hours ahead. We develop a forecasting strategy based on aggregation of expert forecasts. In a first step, we design various forecasting methods by using several machine learning methods (gradient boosting machine and random forests) and nonparametric statistical models (generalized additive models and curve linear regression). We show how, on this data set, combining these heterogeneous methods online improves significantly the forecasting accuracy. We also consider an electric load data set and show that the methodology used for heat load forecasting can easily be extended to electricity demand.

Contents

NOTA: This chapter is a joint work with Yannig Goude and Côme Bissuel. It is based on the technical report [11].

9.1. Introduction

9.1.1. Context

We consider a cogeneration plant which aims at producing electricity and heat at the same time. Every day, the operators have to know how much heat will be consumed by the nearby city so as to plan the functioning of the plant and thus know how much electricity they will be able to sell at the best cost. For big combined heat and power plants, the forecast in electricity production is also required by the local network manager. The forecast of the heat demand is therefore needed at different time horizons, from the next few hours for the daily steering of the plant up to a week for the forecast in electricity production.

We develop a forecasting strategy based on experts aggregation. The term experts refers to the different forecasts obtained from various forecasting models based on machine learning algorithms such as random forests (RF), see Breiman [35], or gradient boosting machine (GBM), see Ridgeway [127], non-linear additive models such as Generalized Additive Models (GAMs), see Hastie and Tibshirani [88], Wood [147], and curve linear regression models (CLR) presented in Cho et al. [50]. We consider these experts as they previously obtained nice performance on electricity load data sets and we found a lot of similarity between these case studies and the present application. GAMs were extensively and successively tested for electricity forecasting in e.g., Pierrot and Goude [121], Fan and Hyndman [66], Ba et al. [22]; and Wood et al. [148]. GAMs were already applied on heat load forecasting in Bissuel et al. [26]. Random forests were coupled to a GAM in Nedellec et al. [117] to derive short term forecasts. To our knowledge, gradient boosting machines have never been applied for load forecasting but we chose it for the nice performance obtained in many machine learning contests (see e.g., Conort [54]). Finally, curve linear regression models proposed a different -and complementary- view on load data and at the same time obtained nice forecasting performance as demonstrated in the empirical studies of Cho et al. [50, 51].

Aggregation of experts is a dynamic field of research in the machine learning community and the empirical literature is large and diverse. Due to the massive development of new forecasting methods and their implementation in open source softwares, practitioners have more and more access to a large variety of forecasts and aggregating them is a natural ambition. In many recent forecasting challenges (see e.g., the energy forecasting competition GEFCOM12 presented in Hong et al. [95], or the well known Netflix competition studied in Paterek [119]), combining forecasts is thus a key point that often makes the difference between competitive teams. Aggregation rules are developed by a lot of different researchers with heterogeneous backgrounds and making an exhaustive review is a hard task (see e.g., the one of Clemen [53]). We will focus on the framework of individual sequences as described in Cesa-Bianchi and Lugosi [43] to design our aggregation algorithms. Due to its very general theoretical hypothesis, this framework could easily embed our heterogeneous experts. However, few real-world empirical studies consider this framework. One can cite for instance climate prediction in Monteleoni et al. [115], air-quality prediction in Mallet [110], Mallet et al. [111], quantile prediction of daily call volumes entering call centers in Biau and Patra [24], or electricity consumption in Devaine et al. [60] and Chapter 4. To our knowledge, no publication exists dealing with random forests and gradient boosting machines neither than aggregation of experts for heat load forecasting. Furthermore, we extend the aggregation procedures proposed in Devaine et al. [60] and Chapter 4 to the context of multi-horizon (from 1 hour to 72 hours ahead)

forecasting.

In Section 9.2 we detail the our methodology and the generic statistical methods that we will use for each data set to perform predictions. Our forecasting strategy is based on a two-step procedure: first, we built our experts using machine learning methods (RF and GBM) and statistical methods (GAM and CLR); then, we combine them online by using combining algorithms from the setting of individual sequences (see Section 9.2.2). We study two data sets. The first one deals with heat load and is analyzed in Section 9.3. The second data set aims at predicting the electricity demand in France. It is analyzed in Section 9.4.

9.2. Methodology

In this section, we present the methodology that we used on both data sets to forecast the heat load and the electric demand at different horizon of time. It follows a two-step approach. First, we use four generic regression methods (that we describe in Section 9.2.1) to design five individual forecasters for each data set. Second, we combine online throughout a testing set these five forecasters by using combining algorithms detailed in Section 9.2.2.

9.2.1. Classification and regression methods considered

We implemented several forecasting methods in `R`. They all rely in combinations of the following four regression (or classification) methods: random forests, gradient boosting machine, generalized additive models, and curve linear regression.

Random forests are a powerful ensemble method introduced by Breiman [35] that builds a large number of random regression (or classification) binary trees before aggregating them, so as to improve their prediction accuracy. The process of fitting a random forests model and using it to produce a prediction performs three steps.

1. It performs bagging: it creates multiple bootstrap samples of the training set by randomly sampling n instants in the training set with replacement, where n is the size of the training set. About a third of the initial training set is left out in each bootstrap sample.

2. It builds random prediction trees: for each bootstrap sample, it builds a corresponding binary decision tree. To do so, it partitions the covariate space recursively by choosing a covariate to split and a corresponding threshold until a stopping condition is reached. The trees are extended very deeply in practice and have high variance and low bias. For a new data point (X_t, Y_t), where X_t denotes the vector of covariates (like $T_t^p, Toy_t, \dots$), each tree forms a prediction for Y_t by averaging the outputs Y_s of the past data of its bootstrap sample whose covariates X_s belongs to the same subspace as X_t. For classification, averaging is replaced by majority voting.

3. It aggregates the trees: to form a prediction, it averages the predictions of the trees. This allows to reduce drastically the variance of the predictor.

In our experiment, we used the `R` package `randomForests` of Liaw and Wiener [106] that implements Breiman's random forests algorithm [35] for classification and regression. We used the default parameters of the method. The parameter `ntree` which controls the number of trees of the forest

is set to its default value 500 and the parameter `mtry`, which controls the number of variables randomly sampled as candidates at each split is set to $\sqrt{p}$ for classification and $p/3$ for regression, where p denotes the total number of covariates.

Gradient boosting machine (`gbm`) is another ensemble method introduced by Friedman [72]. But, by contrast to random forests it builds the basis forecasters sequentially by trying to reduce the bias of the combined predictor. Furthermore, the basis forecasters are a class of weak regression methods (e.g., decision trees with very limited maximal depth). They exhibit large bias but low variance. Basically, at each optimization step $i \geqslant 1$, the boosting algorithm fits a new basis forecaster $\widehat{g}_i$ that aims at predicting the residuals of the preceding forecaster $\widehat{f}_{i-1}$. Then, it updates the combined forecaster by adding the correction $\widehat{f}_i = \widehat{f}_{i-1} + \eta \widehat{g}_i$, where $\eta > 0$ is a tuning parameter. The bias of the combined predictor will thus decrease step by step.

The learning rate $\eta > 0$ and the number of optimization steps are two parameters that have to be tuned carefully. As their optimal values are closely related, usually one of them is fixed in advance, and the other one optimized by performing cross-validation or by using the out-of-bag error.

In our experiments, we used the `R` package `gbm` of Ridgeway [127], which implements extensions to Adaboost algorithm of Freund and Schapire [70] and Friedman's gradient boosting machine, see Friedman [72], Friedman et al. [74], Friedman [73], Hastie et al. [86]. It can be used either for classification (by choosing a Bernoulli distribution) or for regression (by choosing a Gaussian distribution). In all our experiments, we used the following parameters: the basis forecasters were binary decision trees with a depth of variable interaction `interaction.depth` $= 2$ and at least `n.minobsinnode` $= 5$ observations per node. First, we performed a large number of optimization steps controlled by `n.trees` $= 7000$ with learning rate `shrinkage` $= 0.005$ (i.e., $\eta = 0.005$); then, the actual number of trees used for prediction was optimized by minimizing out-of-bag error thanks to the function `gbm.perf`.

Generalized additive models (`gam`) are a nonparametric regression method introduced by Hastie and Tibshirani [87]. It consists in the following statistical model:

$$Y_i = f_1(X_{1,i}) + f_2(X_{2,i}) + ... + f_p(X_{p,i}) + \varepsilon_i$$

where Y_i is a univariate response variable and $X_{q,i}$ are the covariates that drive Y_i. In the following application, Y_i will be the heat load demand, $X_{q,i}$ the meteorological and calendar predictors. ε_i denotes the model error at time i. The non-linear functions f_q are supposed to be smooth. They are estimated by penalized regression in a spline basis: $f_q(X) = \sum_{j=1}^{k_q} \beta_{q,j} b_j^q(X)$, where k_q is the dimension of the spline basis –potentially different for each effect– and $b_j^q(X)$ the corresponding spline functions. Denoting by B the matrix formed by concatenation of the b_j^q, S_q a smoothing matrix depending on the spline basis, the regression parameters β are obtained by minimizing $\min_{\beta,\lambda} \|Y - B\beta\|^2 + \sum_{q=1}^{p} \lambda_q \ \beta^T S_q \beta$ which is the ridge regression problem corresponding to the optimization problem

$$\sum_{i=1}^{n} (Y_i - \sum_{q=1}^{p} f_q(X_i))^2 + \sum_{q=1}^{p} \lambda_q \int \|f_q''(x)\|^2 \, \mathrm{d}x \,,$$

where the penalty parameter $\Lambda = (\lambda_1, ..., \lambda_p)$ controls the degree of smoothness of each effect -the

higher λ_q the smoother f_q is- and has to be optimized, minimizing the GCV –Generalized Cross Validation– criterion. We used the `R` package `mgcv`, see Wood [147], Wood et al. [148].

Curve linear regression (`clr`) is a nonparametric regression method described in Cho et al. [50, 51]. It consists in performing a data driven dimension reduction together with a data transformation in order to reduce the complex functional regression problem to a multiple linear regression. Schematically, we consider that observations are couple of curves $(X_i(.), Y_i(.))$ corresponding to covariates (e.g., temperature at the plant) and the heat load on the i^{th} day. Y_i is defined on the index set $\mathcal{I}_1$ and $X_i(.)$ on the index set $\mathcal{I}_2$. We model the dependency between X and Y via the curve linear regression:

$$Y_i(u) = \int_{\mathcal{I}_2} X_i(v)\beta(u,v)\,\mathrm{d}v + \varepsilon_i(u)\ u \in \mathcal{I}_1$$

where β is a regression coefficient function defined on $\mathcal{I}_1 \times \mathcal{I}_2$ and $\varepsilon_i(u)$ is a mean 0 noise. As explained in detail in Cho et al. [50], we use a dimension reduction based on SVD decomposition of the covariance matrix between X and Y, to single out the direction upon which this 2 processes are most correlated.

It is not implemented in an `R` package yet. We will describe in details in Section 9.3.3 our implementation of the method.

9.2.2. Combining algorithms

We consider the following sequential setting of robust online aggregation of predictors, see the monograph of Cesa-Bianchi and Lugosi [43] for details. We suppose that the sequence of observations of the heat load $y_1, \ldots, y_n$ of the testing set is observed step by step. At each time step $t = 1, \ldots, n$ of the testing set, a finite number $K \geqslant 1$ of experts (designed in Section 9.1) propose predictions $x_{k,t+h}$ of the future heat load y_{t+h}. The combining algorithm is then asked to form a weight vector $\widehat{\boldsymbol{p}}_{t+h} \in \mathbb{R}^K$, with knowledge of the past observations of the heat load $y_1, \ldots, y_t$ and the experts predictions until time $t + h$. The combining algorithms predicts the weighted linear combination of the expert forecasts $\widehat{y}_{t+h} = \widehat{\boldsymbol{p}}_{t+h} \cdot \boldsymbol{x}_{t+h}$, where $\cdot$ denotes the scalar product. The goal of the combining algorithm is to minimize its final RMSE over the testing set.

The literature of prediction with expert advice is rich and many combining algorithm (or aggregation rules) exist. We performed our experiments by considering two different aggregation rules: the ridge regression forecaster (Ridge), and the polynomially weighted average forecaster with multiple learning rates (ML-Poly). Both of them are described in the context of electricity load forecasting in Chapter 4. We choose these algorithms among others because they achieve nice forecasting performance and produce quite different evolutions of weights. Furthermore, ML-Poly can be calibrated totally online based on theoretical oracle bounds.

Usually, combining algorithms provide one step ahead predictions, i.e., at horizon $t+1$. We detailed below the extension of ML-Poly and Ridge to horizon $t + h$ for any fixed horizon $h \geqslant 1$.

The polynomially weighted average forecaster with multiple learning rates (ML-Poly) is introduced and detailed in Section 2.3.2. Its implementation is extended in Algorithm 15 to an arbitrary horizon of prediction $h \geqslant 1$. Algorithm 15 basically consists in shifting by h time steps the weight vectors formed by Algorithm 3.

Algorithm 15: The polynomially weighted average forecaster with multiple learning rates for arbitrary horizon of prediction (ML-Poly)

Input: $h \geqslant 1$, horizon of prediction
Initialize: for $t \leqslant h$, $\boldsymbol{p}_t = (1/K, \ldots, 1/K)$ and $\boldsymbol{R}_0 = (0, \ldots, 0)$

for each instance $t = 1, 2, \ldots, n-h$ **do**
1. predict $\widehat{y}_t = \widehat{\boldsymbol{p}}_t \cdot \boldsymbol{x}_t$ and observe y_t
2. define the pseudo prediction $\widetilde{y}_t = \widehat{\boldsymbol{p}}_{t+h-1} \cdot \boldsymbol{x}_t$
3. pick the learning rates

$$\eta_{k,t} = \left(1 + \sum_{s=1}^{t} \big(\ell_s(\widetilde{y}_s) - \ell_s(x_{k,s})\big)^2\right)^{-1}$$

where $\ell_s : x \mapsto x(y_s - \widetilde{y}_s)$.
4. for each expert k update the regret

$$R_{k,t} = R_{k,t-1} + \ell_t(\widetilde{y}_t) - \ell_t(x_{k,t})$$

5. form the mixture $\widehat{\boldsymbol{p}}_{t+h}$ defined component-wise by

$$\widehat{p}_{k,t+h} = \eta_{k,t} \left(R_{k,t}\right)_+ \Big/ \left[\textstyle\sum_{j=1}^{K} \eta_{j,t} \left(R_{j,t}\right)_+\right]$$

where $\boldsymbol{x}_+$ denotes the vector of nonnegative parts of the components of $\boldsymbol{x}$
end for

Remark 9.1. Another natural solution to produce forecasts for an horizon $h > 1$ consists in replacing the pseudo predictions $\widetilde{y}_t$ with the true predictions $\widehat{y}_t$ in Algorithm 15 (i.e., the definition of $\widetilde{y}_t$ in Step 2 is substituted with $\widetilde{y}_t = \widehat{y}_t$). Unfortunately we observed poor practical performance for this version. This can be intuitively explained as follows. Suppose that a weight vector $\widehat{\boldsymbol{p}}_{t+h-1}$ was badly updated at time step $t-1$. In Algorithm 15 the error of $\widehat{\boldsymbol{p}}_{t+h-1}$ is immediately taken into account at time step t and corrected in the following weight vector $\widehat{\boldsymbol{p}}_{t+h}$ (because $\widetilde{y}_t$ depends on $\widehat{\boldsymbol{p}}_{t+h-1}$). This is not the case if we substitute the $\widetilde{y}_s$ with the $\widehat{y}_s$. The weight vector $\boldsymbol{p}_{t+h-1}$ is used for the first time in $\widehat{y}_{t+h-1}$ and its correction is then only included for the definition $\widehat{\boldsymbol{p}}_{t+2h-1}$ (Step 5 of Algorithm 15) at time step $t+h-1$. Thus the algorithm suffers a delay in its optimization procedure which increases the variance of the errors.

Remark 9.2. Remark that there is a downside in using pseudo predictions $\widetilde{y}_t$ instead of $\widehat{y}_t$ in Algorithm 15. The algorithm overestimates its performance and behaves as if it was performing one step ahead prediction. The learning rates $\eta_{k,t}$ are not reduced enough. A solution (left for future research) could be to replace the $\widetilde{y}_t$ with the $\widehat{y}_t$ only in the learning-rate update (Step 3).

In Chapter 2 we proved that the performance of ML-Poly converged to the one of the best fixed weight vector in the simplex at the speed $\mathcal{O}(n^{-1/2})$. ML-Poly is particularly interesting since its fully adaptive and theoretical tuning of the learning parameters $\boldsymbol{\eta}_{k,t}$ achieves great performance.

Remark 9.3. We leave for future research the proofs of theoretical bounds of the regret for the

Algorithm 16: The ridge regression forecaster for arbitrary horizon of prediction (Ridge)

Input: $\lambda > 0$, learning rate; $h \geqslant 1$, horizon
Initialize: for $t \leqslant 0$, $\quad \widehat{\boldsymbol{p}}_t = (1/K, \ldots, 1/K)$
for each instance $t = 1, 2, \ldots, n-h$ **do**
1. form the mixture $\widehat{\boldsymbol{p}}_{t+h}$ defined by

$$\widehat{\boldsymbol{p}}_{t+h} = \operatorname*{arg\,min}_{\boldsymbol{u} \in \mathbb{R}^K} \left\{ \sum_{s=1}^{t} \left(y_s - \boldsymbol{u} \cdot \boldsymbol{x}_s \right)^2 + \lambda \left\| \boldsymbol{u} - \widehat{\boldsymbol{p}}_0 \right\|_2^2 \right\}$$

2. output prediction $\widehat{y}_{t+h} = \widehat{\boldsymbol{p}}_{t+h} \cdot \boldsymbol{x}_{t+h}$
end for

extension of ML-Poly to h steps ahead prediction. We are convinced that the theoretical guarantees are maintained since the weights output by Algorithm 15 are not to far from the ones formed by the original version of ML-Poly (Algorithm 3). Indeed the weights are computed on almost the same sets which differ by at most h instantaneous bounded losses. Defining generic extension of combining algorithms to multi horizon forecasting with robust guarantees on the regret should be an interesting direction of future work.

The ridge regression forecaster (Ridge) is presented in Algorithm 16. It was introduced in a stochastic setting by Hoerl and Kennard [94]. It forms at each instance the linear combination of experts minimizing a L_2-regularized least-square criterion on past data. It was first studied in the context of prediction with expert advice by Azoury and Warmuth [21] and Vovk [142] and was proved to enjoy nice theoretical properties, namely a convergence $o(n^{-1})$ to the performance of the best fixed linear oracle.

Once again, the learning parameter λ of the ridge regression aggregation rule has to be calibrated online. This tuning can be done using the methodology detailed in Section 2.4 of Devaine et al. [60], which yields good practical performance but has no theoretical guarantees.

Ridge produces linear mixtures: the weights may be negative and their sum does not necessarily equal one. ML-Poly, on the contrary, restricts itself to convex combination of experts. In other words, it only forms weight vectors $\widehat{\boldsymbol{p}}_t$ that belong to the simplex $\Delta_K = \{\boldsymbol{x} \in \mathbb{R}_+^K \quad \sum_i x_i = 1\}$. While linear aggregation rules might have more flexibility to detect correlation between experts and therefore often reach better performance, convex aggregation offers easy interpretation and safe predictions. Indeed convex weight vectors only assign nonnegative weights to experts and their predictions always lie in the convex hull of experts predictions. Thus if all the experts are known to perform well, the aggregation rule will do so as well.

9.3. A first data set: forecasting heat load

9.3.1. Presentation of the data set

We consider a data set that consists of hourly measurements of the load Y_t of a cogeneration plant between October 1, 2009 and October 15, 2013. Several covariates, that are identified by operational

teams to have an impact on the load, are also provided:

- Temperatures: T^p, the outside temperature at the plant; T^a, the temperature at the nearby airport; T^w, the temperature of the water leaving the plant; T^s, the setpoint temperature of the water asked by the network operator a few hours in advance (not known at the time of prediction);
- Calendar variables: $Toy \in [0, 1]$ a variable describing the position in the year ranging from 0 at the beginning of each year to 1 at the end; $D \in \{\text{Monday}, \ldots, \text{Sunday}\}$ represents the weekdays.

In the following, all variables are indexed by the time instant t. Our goal is to provide forecasts of the load for different time horizons going from 1 hour to 72 hours in advance.

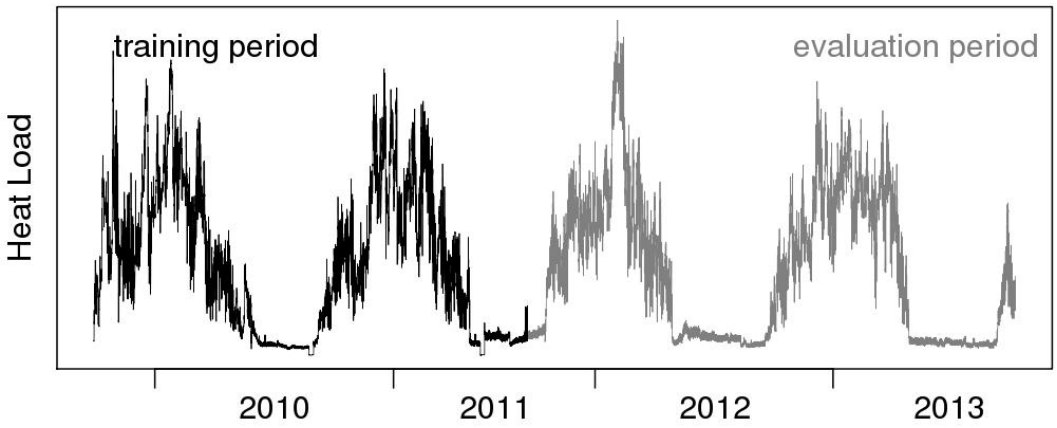


Figure 9.1.: The observed hourly load between October 1, 2009 to October 15, 2013.

The data set is partitioned into two pieces of approximatively two years (see Figure 9.1). The first one, the training set, consists of 656 days from October 1, 2009 to August 31, 2011. It will be used to fit the forecasting methods. The second one, the testing set, has 734 days from September 1, 2011 to October 15, 2013. It will be used to evaluate the performance of the forecasting methods. Prediction accuracy is measured by the root mean square error (RMSE). We clearly see on Figure 9.2a, representing the heat load according to the position in the year (Toy), a marked yearly cycle mostly due to the outside temperature variation along the year.

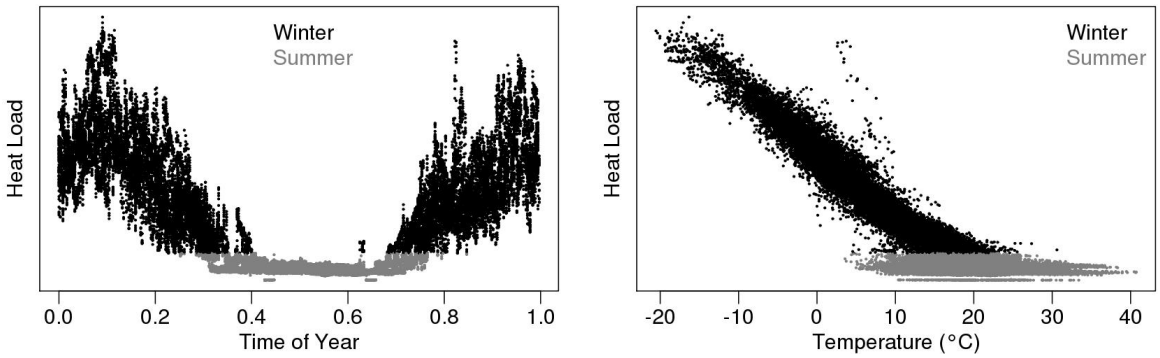
Winter and summer modes. Figure 9.2b shows the heat load according to the outside temperature (T^p). It confirms the impact of this covariate. The main property of these data stands into a two regimes behavior: winter and summer modes. We can observe that the relationship between temperature and heat load is not really a function but that, for temperature ranging from 5 to 10 Celsius degrees, the same temperature value corresponds to two heat load responses depending on the season. More precisely, two behaviors can be observed:

- winter: heavy correlation between the outside temperature and the load. The primary goal of the plant is to product heat and electricity comes as an extra.
- summer: low correlation between the outside temperature and the load. There is little need for heating and the cogeneration plant is idling.

Models that consider few interactions between covariates (such as generalized additive models with univariate effects) cannot capture these two regimes in a single model, because they try to explain

the impact of each covariate on the load separately.

Thus, to cope with these two regimes, we introduce S_t an exogenous discrete variable defined by $S_t = \mathbb{1}_{Y_t>100}$. Thus, $S_t = 1$ (resp. $S_t = 0$) indicates high (resp. low) value of the load, which corresponds to the winter (resp. summer) regime. These two different modes are depicted in Figure 9.2. We observe in black a linearly decreasing dependency of the heat load in winter: the colder it is outside, the more heat needs to be produced. Gray points plot the summer mode when low correlation between T^p and Y is observed.



(a) Heat load according to the position in the year.

(b) Heat load according to the outside temperature T^p.

Figure 9.2.: Graphical representation of the two heat load regimes

Horizon of prediction. We consider several horizons of prediction going from 1 hour ahead to 72 hours ahead. In the following, the horizon of prediction is denoted by h. That is, to predict Y_t, the forecaster has access to the past observations $Y_1, \ldots, Y_{t-h}$ of the load and past observations of the meteorological covariates. However we assume that the forecaster has access to the calendar variable at time t and the airport temperature T^a until time instant t. In case of operational forecasting, the future values of T^a should be replaced by their forecasts. Temperature at the plant T^p is measured but no forecasts were provided at this point.

The evaluation criterion. To assert the predictive accuracy of our forecasting methods, we consider the root mean square error (RMSE) over the testing set. It is defined for a sequence of predictions $\widehat{Y}_{1:n} = (\widehat{Y}_1, \ldots, \widehat{Y}_n)$ of the heat loads $Y_{1:n} = (Y_1, \ldots, Y_n)$ as follows:

$$\text{RMSE}\big(\widehat{Y}_{1:n}\big) = \sqrt{(1/n)\textstyle\sum_{t=1}^{n}\big(Y_t - \widehat{Y}_t\big)^2},$$

where $t = 1, \ldots, n$ are the time instants of the testing set. The regression methods considered in this empirical study are thus optimized to minimized the square loss.

Due to confidentiality constraints, the RMSEs reported in Section 9.3.4 are all divided by the average heat load over the testing set $(1/n)\sum_{t=1}^{n} Y_t$.

The RMSE was preferred to the mean absolute percentage of error (MAPE) because some of the heat load values are close to zero in summer (see Figure 9.2a and 9.2b). MAPEs are extremely sensitive to those values and thus quite unstable.

We propose a two-step forecasting strategy to cope with the two regimes structure of the data. First, we form $\widehat{S}_t$ a forecast of the cluster S_t in which the observation t to be predicted belongs to.

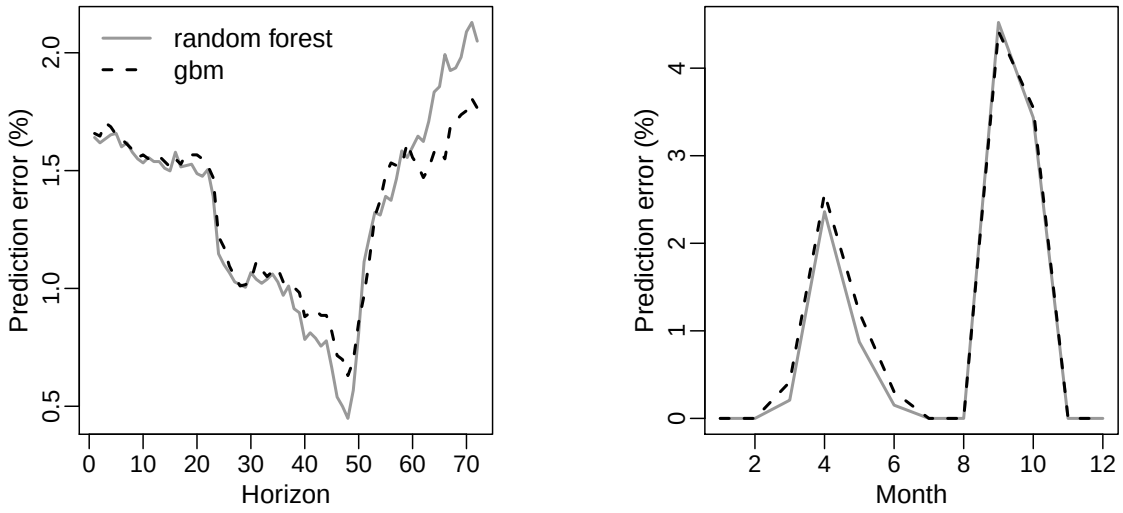
This is done by using random forests or gradient boosting machine classifiers. Then, conditionally to $\widehat{S}_t$, we propose different non-linear regression methods based on GAM, random forest, gradient boosting machines or curve linear regression to forecast Y_t.

9.3.2. Forecasting the mode: winter or summer?

Let us fix a time horizon $h \geqslant 1$ (in hours) of prediction. To predict the cluster S_t at time horizon h, we consider the following exogenous variables defined in section 9.3.1: D_t, Toy_t, T_t^a, T_{t-h}^p the last observation of the temperature at the plant; Y_{t-h}, the last observation of the load; and Y_{t-H}, the observation of the load H hours before, where H equals $\min_{x\in\{24,48,72,96\}}\{h < x\}$.

Then, to perform the classification, we fit either a random forests classifier or a gradient boosting machine classifier (`gbm`) by plugging these covariates in the methods described in Section 9.2.1.

Figure 9.3 reports the prediction performance of both classifiers according to the horizon of prediction and to the month. Random forests and `gbm` achieve similar performance which is about 0.5% or errors one hour ahead and 2% of errors 72 hours ahead. Besides, we see that the errors are largely suffered during the mid-season.



(a) Performance according to the horizon of prediction (in hours)

(b) Performance according to the month

Figure 9.3.: Percentage of prediction errors of the mode (winter vs summer).

9.3.3. Five experts to forecast the heat load

We build five forecasters to produce h hours in advance the predictions of the load Y_t. The predictions formed by these procedures will then be plugged in the combining algorithms described in Section 9.2.2. Section 9.3.4 will compare the performance obtained by the different forecasters and combining algorithms.

In the rest of this section, except when stated otherwise (CLR and residual analysis), we use 24 separate models for each hourly period. This choice is driven by our previous experiences in load forecasting, as detailed in Pierrot and Goude [121] or Nedellec et al. [117].

The Gam forecaster (GAM). As previously explained we propose two Generalized Additive Models corresponding to a winter and a summer regime:

- Winter model (sensibility to the temperature). For observations such that $S_t = 1$. The forecaster fits the following `gam` (see Section 9.2.1):

$$Y_t = \sum_{k=1}^{7} \alpha_k \mathbb{1}_{\{D_t=k\}} + f_1(Toy_t) + f_2(T_t^a) + f_3(Y_{t-H}, T_{t-H}^p) + f_4(Y_{t-h}, T_{t-h}^p) + \varepsilon_t$$

  Here, the α_k are seven real coefficients and the f_i are cubic regression splines to be estimated.
- Summer model (no sensibility to the temperature). For observations such that $S_t = 0$. The forecaster fits the following `gam` model:

$$Y_t = \sum_{k=1}^{7} \alpha_k \mathbb{1}_{\{D_t=k\}} + f_1(Toy_t) + f_2(T_t^a) + f_3(Y_{t-H}) + f_4(Y_{t-h}) + \varepsilon_t$$

where ε_t is a i.i.d. random noise and f_i are smooth functions, estimated by penalized regression on B-spline basis as proposed in Wood [147].

The GBM forecaster (GBM). Similarly to GAM, the GBM forecaster fits two different `gbm` models for summer and winter regimes as described in Section 9.2.1. Both models use the same exogenous variables as in GAM: D_t, Toy_t, Y_{t-h}, Y_{t-H}, T_{t-h}^p, T_{t-H}^p and T_t^a. The only difference stands in the data considered for the estimation corresponding to $S_t = 0$ or $S_t = 1$. To produce a forecast, the model is chosen according to the prediction of S_t by the classification `gbm` described in Section 9.3.2.

The Random forests forecaster (RF). Unlike previous forecasters, the Random forests forecaster fits a single `randomForest` model for all data as described in Section 9.2.1. In other words, RF does not partition the data according to their mode (summer, winter). This clustering step is indeed not needed by RF because it considers high interactions between covariates and can thus directly capture the two regimes of the heat load in a single model.

The covariates considered are the same as for GBM, that is D_t, Toy_t, Y_{t-h}, Y_{t-H}, T_{t-h}^p, T_{t-H}^p, and T_t^a.

Gam mid-term followed by short term residual analysis (GamMTCT). This forecaster, like GAM and GBM, fits two different models according to the cluster S_t:

- Winter model ($S_t = 1$). First, the forecaster fits a medium term generalized additive model following the equation $Y_t = f_1(Toy_t) + f_2(T_t^a) + \varepsilon_t$. Then, a short-term correction is performed assuming a short term `gam` structure of the medium term residuals: $\varepsilon_t = \sum_{k=1}^{7} \beta_k \mathbb{1}_{\{D_t=k\}} + f_1(\varepsilon_{t-H}, \varepsilon_{t-h}) + u_t$ where u_t is a i.i.d. random noise. In practice, we consider the estimated residuals $\widehat{\varepsilon_t}$ obtained after the medium term estimation process and fit the short term `gam` on the $30 \times 24 = 720$ last observations such that $S_t = 1$. This residual analysis, contrary to the medium-term model, is not performed separately for each hourly period.
- Summer model ($S_t = 0$). We proceed similarly to the winter model except that the medium term `gam` model is fitted using the equation $Y_t = f(D_t) + \varepsilon_t$.

Curve Linear Regression (CLR). Contrary to previous methods, CLR does not produce a single punctual forecast $\widehat{Y}_{t+h}$ for an horizon h. With knowledge of the past consumption, of the future temperature T^a, and of the past temperature T^p, it simultaneously forms forecasts of the electricity consumption of the next day (i.e., the next 24 points $Y_{t+1}, \ldots, Y_{t+24}$) assuming it as a curve. Thus, data are not considered point by point but day by day and the models are not hourly performed which is a quite different assumption on the heat load process than the other models proposed before. For any variable X^*, we denote by $\underline{X}_i$ the vector of 24 points $(X_{24\,i+1}, \ldots, X_{24\,i+24})$ that correspond to day number $i \geqslant 0$. CLR aims at predicting sequentially the multidimensional time series $(\underline{Y}_i)$.

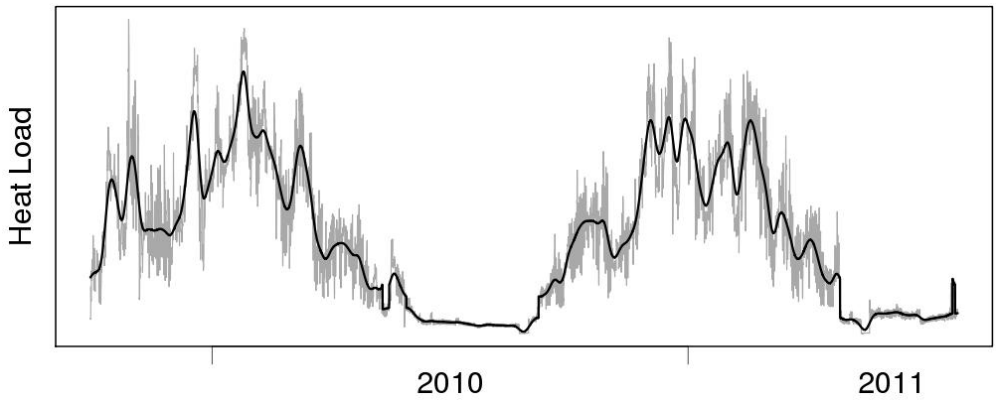


Figure 9.4.: The observed hourly heat loads [gray] together with the smoothed medium term model [black] fitted by clr, see step (b).

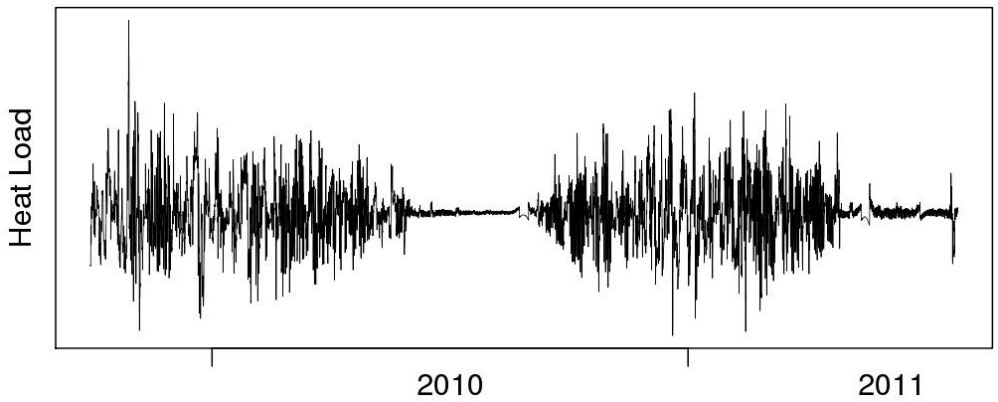


Figure 9.5.: The residual signal (once the medium term model has been removed from the heat loads) on which curve linear regression is performed, see step (c) of clr.

Data are partitioned into two pieces: summer days (such that $\text{mean}(\underline{S}_i) < 0.2$), and winter days ($\text{mean}(\underline{S}_i) \geqslant 0.2$). Forecasting the next day (that is, the next 24 instances $Y_{t+1}, \ldots, Y_{t+24}$) can be done by performing the following steps:

1. we determine in which cluster (summer or winter) the next day will be belonging to by performing random forests classification and we consider only these days in the past training set.

*for instance X will be replaced by Y, T^p, T^a, S, ...

2. as in Cho et al. [50], we stationnarize the signal by fitting a medium term model on the load, see Figure 9.4. To do so, we smooth the load $Y_t \to sY_t$ and the temperature $T_t^a \to sT_t^a$ by applying a Gaussian kernel, with a bandwidth of approximately one week (`ksmooth` function in `R`). Then, we fit on the training data the following `gam` model: $sY_t = f_1(sT_t^a) + \varepsilon_t$. We finally remove the medium-term model from the load by considering the estimated residuals $\widehat{\varepsilon}_t = Y_t - sY_t$, see Figure 9.5.

3. the curve linear regression is performed with daily curves $\underline{\widehat{\varepsilon}}_i$, $\underline{T}_i^p$ and $\underline{T}_i^a$, considering their de-meaned and standardized counterparts $\underline{\widetilde{\varepsilon}}_t$, $\underline{\widetilde{T}}_i^p$ and $\underline{\widetilde{T}}_i^a \in \mathbb{R}^{24}$. We apply the method of Cho et al. [50], where the response is $\underline{\widetilde{\varepsilon}}_i$ and the explanatory variables are $\underline{\widetilde{\varepsilon}}_{i-1}$ the residuals of the last day, $\underline{\widetilde{T}}_{i-1}^p$ the outside temperature of the previous day, and $\underline{\widetilde{T}}_i^a$ the airport temperature of the day. We set the parameters to $r = 10$ and $M = 20$.

   Horizon of prediction. Initially, we divide the data into consecutive blocks of 24 hours: from 00:00 to 23:00 each day so that CLR provides forecasts at horizon $t+1$ at 00:00, $t+2$ at 01am, ..., and $t+24$ at 11pm. In order to provide forecasts at horizons $t+h$ for $h = 1, \ldots, 24$ and this for all hours in the day, we run 24 translated models: the first one with days defined from 00am to 11pm, the second one with days from 01am to 00am, ... Unlike previous methods CLR does not provide forecasts for more than a daily horizon. The extension to horizon $t+h$, for $h \geqslant 25$ is left for future research.

In this section, we have thus designed five forecasters to provide predictions of the heat load. Each of them is trained by using the training set (from October 1, 2009 to August 31, 2011) and used to produce predictions throughout the testing set (from September 1, 2011 to October 15, 2013). By applying aggregation rules (ML-Poly and Ridge) described in Section 9.2.2, their forecasts can be aggregated online the mixture along the testing set. The goal is to improve further the prediction accuracy by adapting online to the recent observations of the testing set. We report in the next section the results obtained by the five experts (CLR, GAM, GamMTCT, GBM, and RF) and by the two combining algorithms (ML-Poly and Ridge) on the testing set.

9.3.4. Results

Figure 9.6 plots the RMSEs suffered by the five experts together with the RMSEs of the combining algorithms according to the horizon of prediction[†]. As it can be expected, the RMSEs are increasing with the horizon of prediction. We observe that both combining algorithms (Ridge and ML-Poly) achieve for all horizons much better performance than individual experts. The RMSE of Ridge is about 8% smaller than the one of GAM (the best expert), while ML-Poly yields an average improvement of 13.2%.

Figure 9.7 depicts the results of the methods according to the month. Once again, we observe that combining experts improves the quality of the predictions for all months. The heat load is especially better predicted during winter and during the mid-season, seasons of interest for operators because they are harder to predict.

Figure 9.8 and 9.9 plot the time evolution of the weights vector respectively formed by ML-Poly and Ridge for the horizons of prediction 1 hour and 48 hours. We recall that ML-Poly only produces weight vectors that have nonnegative weights and that sum to one, while Ridge can design negative

[†]We recall that due to confidentiality constraints the RMSEs are divided by the averaged heat load.

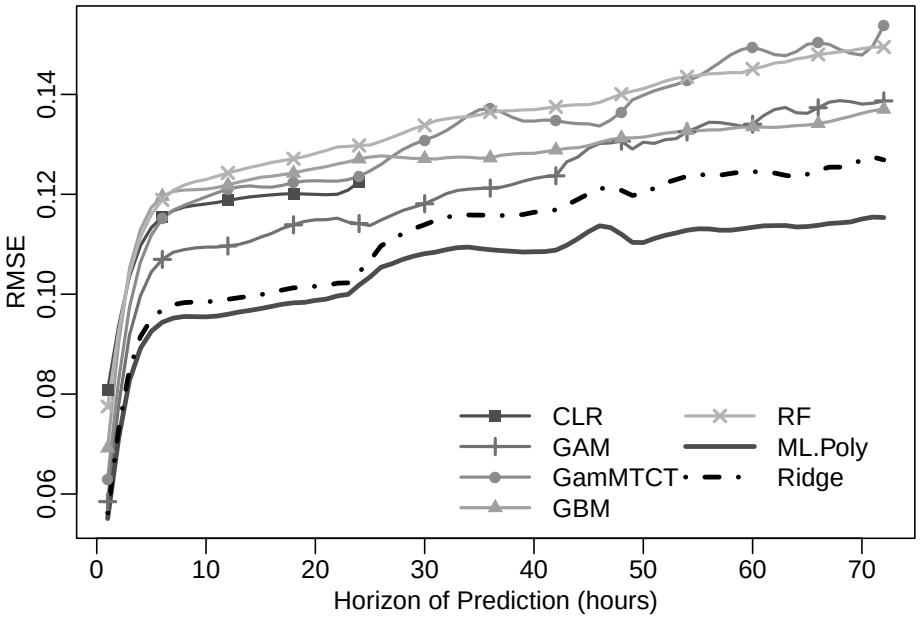


Figure 9.6.: RMSEs according to the horizon of prediction.

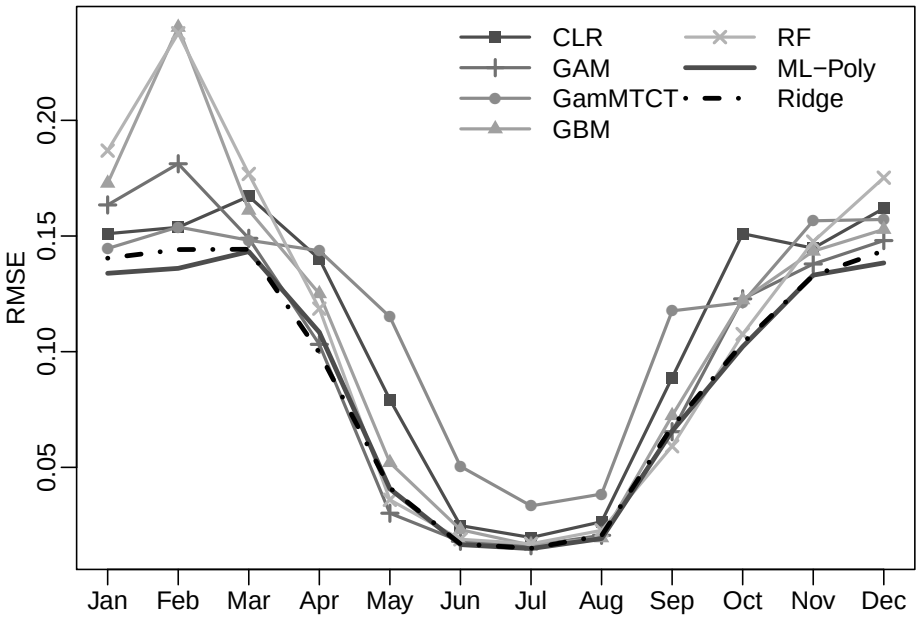
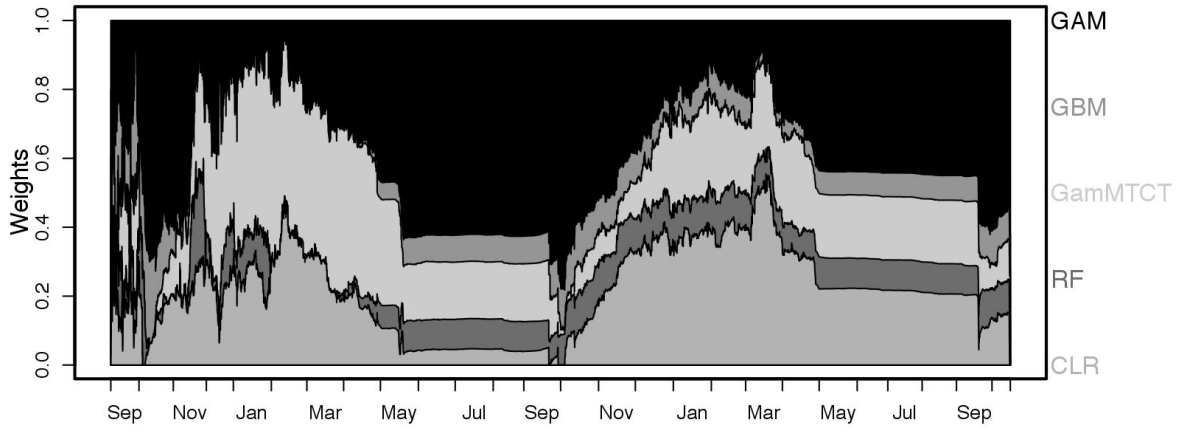
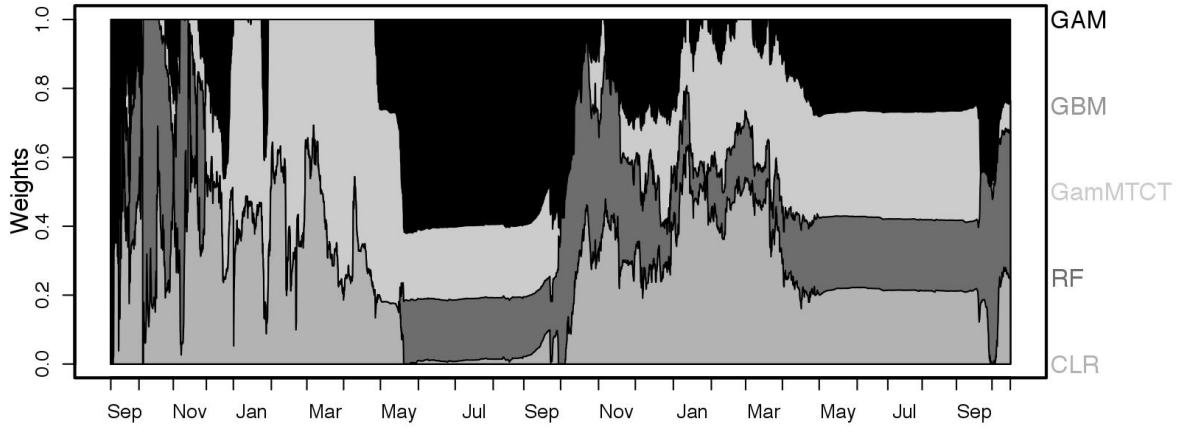


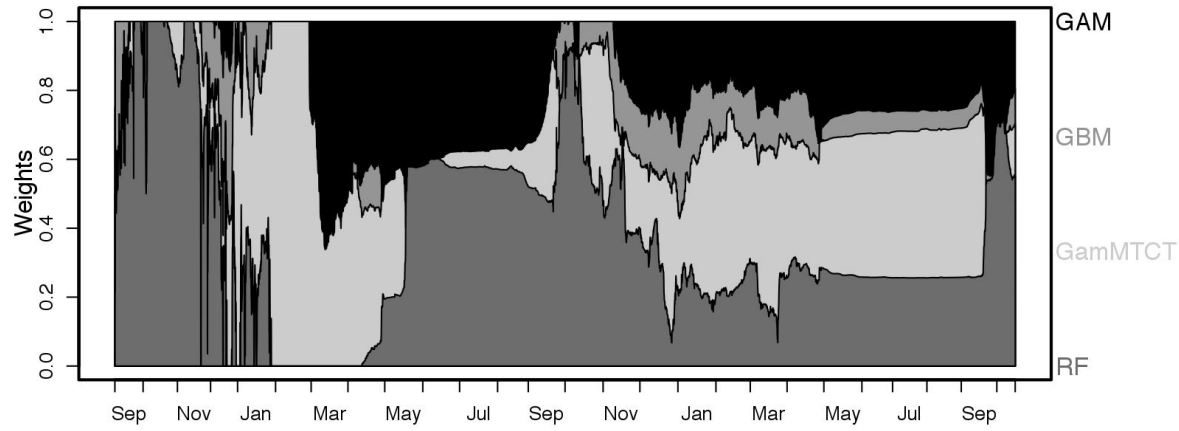
Figure 9.7.: RMSEs according to the month 24 hours ahead.



(a) 1 hour ahead

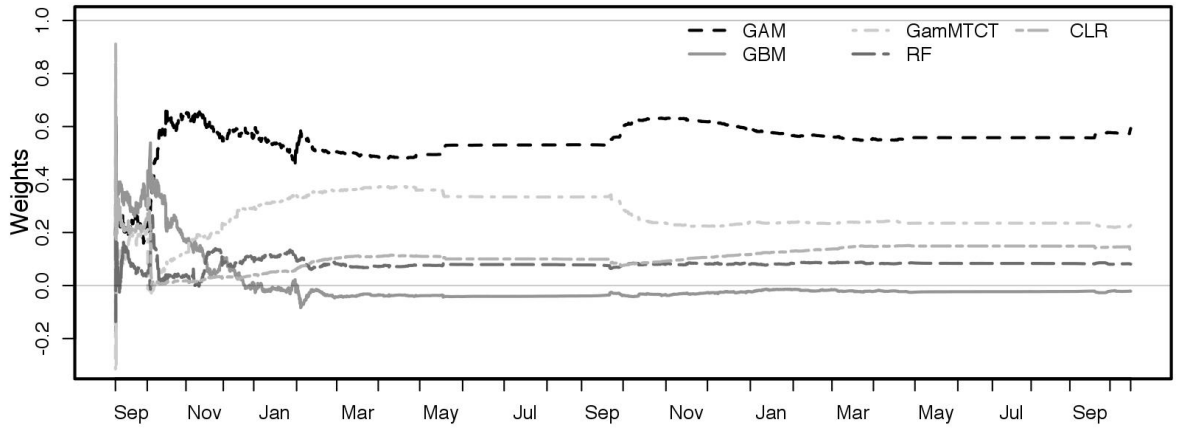


(b) 24 hours ahead

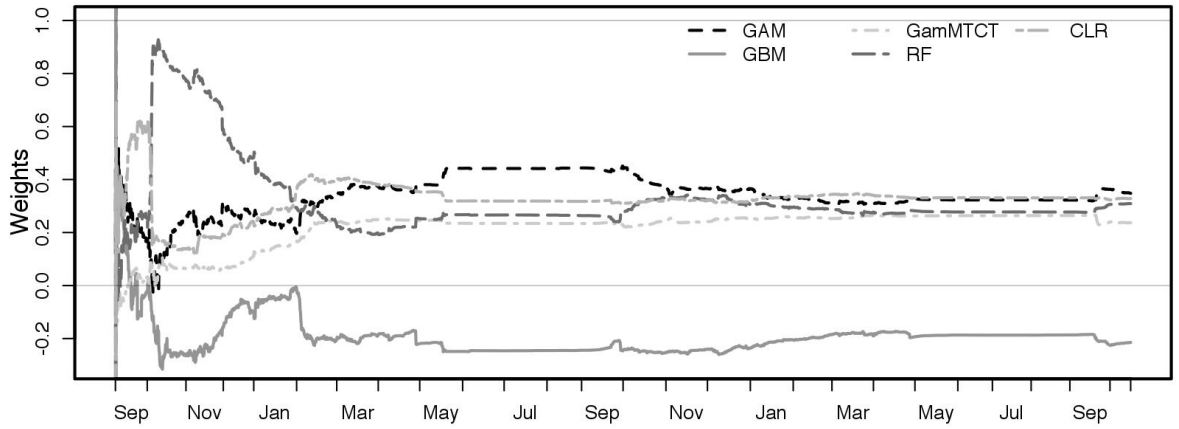


(c) 48 hours ahead

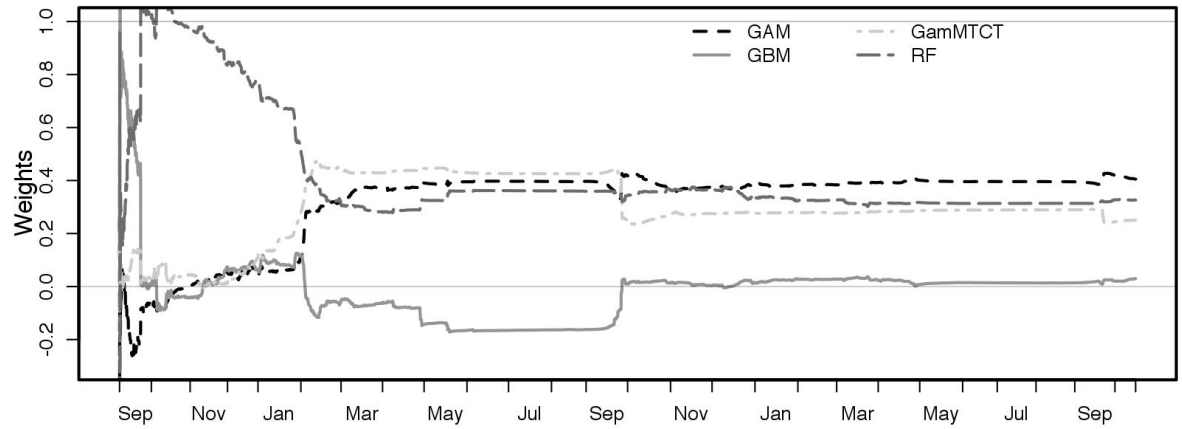
Figure 9.8.: Graphical representation of the weights formed by ML-Poly along the testing set (going from September 1, 2011 to October 15, 2012). At each time instant, the weights of the five experts sum to 1 and are positive. The weight of each expert are assigned from bottom to top to CLR (gray), RF (dark gray), GamMTCM (light gray), GBM (gray), and GAM (black). Remark that CLR does not provide predictions 48 hours ahead.



(a) 1 hour ahead



(b) 24 hours ahead



(c) 48 hours ahead

Figure 9.9.: Graphical representation of the weights formed by Ridge along the testing set (going from September 1, 2011 to October 15, 2012.)

weights. Thus we chose two different graphical representations of the weights. For 1 hour ahead prediction, we have five experts, while only four experts provide predictions for horizon 48 hours since CLR is then inactive. Ridge exhibits weights that are more stable over time which is important for operators who are in favor of stability and easy interpretation. However, we see on Figure 9.8 that the relative volatility of the weights produced by ML-Poly allows the algorithm to retrieve seasonality pattern. Some experts, like CLR, are better during winter, while others are better during summer.

Furthermore, it is interesting to point out that the weights assigned to the experts are quite different one hour ahead in comparison to 48 hours ahead. The strength of the sequential aggregation rules is to retrieve in a fully automatic online fashion the best possible weight combination.

9.4. A second data set: forecasting electricity consumption

To optimize the production, the operational team of EDF needs forecasts at different horizons of time. The goal of this section is to provide such forecasts of the electricity demand from a half-hour ahead to 48 hours ahead. To do so, we reproduce the methodology used to forecast the heat load and we adapt it to a second data set which corresponds to an updated version of the one analyzed in Chapter 4.

First, we design five forecasting methods by adapting the methods used in Section 9.3.3 for heat load forecasting. In the same way, we use gradient boosting machine, random forest, general additive models, and curve linear regression. Then, we produce our final forecasts by combining online these methods.

9.4.1. Presentation of the data set

The data set consists of half-hourly measurements of the total electricity consumption of the EDF market in France from September 6, 2007 to July 27, 2013. We denote by Y_t the electric demand at time $t \geqslant 1$ (all variables are indexed by t the time instant of the observation). It also contains several covariates that were shown to impact the electricity demand. We list below several covariates (or transformations of covariates) that are used in our model. The choices of these covariates and transformations were performed by using a cross validated approach, we refer to Pichavant [120] for details.

Meteorological covariates. T_t is an average of the temperatures in France; $T_t^{(\alpha)}$ is an exponential smoothing of the temperature defined for $\alpha \in [0,1]$ by induction as:

$$T_t^{(\alpha)} = (1-\alpha)T_t + \alpha T_{t-1}^{(\alpha)};$$

$T_t^{(\max)}$ and $T_t^{(\min)}$ are respectively the maximum and the minimum daily smoothed temperature $T_t^{(\alpha)}$, with smoothing parameter $\alpha = 0.996$; we also define $\Delta_t = T_t^{(0.8)} - T_{t-24}^{(0.8)}$ the difference between the smoothed temperature at time t and the one at time $t-24$ (i.e., 12 hours before); W_t is an average of the wind velocity in France; and C_t denotes the cloud cover.

Calendar covariates. $Toy_t \in [0,1]$ (Time of year) is a cyclic variable that indicates the annual position and repeats each year. It is each year linearly increasing over time going from 0 on January

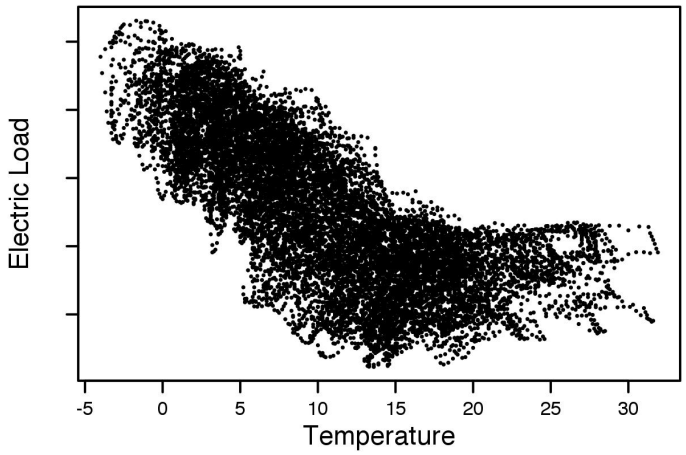


Figure 9.10.: Electric load according to the temperature.

1 at 00am to 1 on December 31 at 12pm, each half-hour having a different value. D_t is a factorial variable with 23 levels corresponding to different types of day (such as Monday, Tuesday,...) and to unusual days (holidays, daylight saving time). This choice was driven by our expertise on electricity load data (see e.g., Goude et al. [81]). We also denote by M_t the month of observation t. It is a categorical variable that can take 12 values (January, February,...).

We excluded from the data unusual days such as public holidays (the day itself, as well as the days before and after it), daylight saving days, winter holidays, and summer holidays. The data then contains 1 623 days (each of them consists of 48 half-hourly observations).

We divide the remaining data set into two sets. The first set covers the period from September 6, 2007 to August 31, 2012. We call it the training set and we use it to design and to fit the experts. Each expert provides then forecasts throughout the second set that we call the testing set. This second set ranges from September 1, 2012 to July 27, 2013. We use it to evaluate the performance of the experts and of the considered combining methods.

There is a great difference with the heat load data set: we do not observe a two regimes behavior and the relationship between temperature and electric demand can here be modeled by a single function (as shown in Figure 9.10). Therefore, we will consider here a single regime and the forecasting methods will not need to determine the regime first.

9.4.2. Five experts to forecast the electric load

We detail now the five forecasting methods that we implemented to forecast the electric demand. We used the same statistical methods as the ones presented in Section 9.3.3. We adapted them to our new data set. Once again, because the electric consumption heavily depends on the hour of the day, the data is parsed into 48 independent time series (one for each half hour of the day) and we train 48 separate half-hourly models unless stated otherwise (CLR and residual analysis). Let $h \in \{1, \dots, 48\}$ be the horizon of prediction in half-hours. We define $H \stackrel{\text{def}}{=} \min\left\{x \in \{48, 96, 144\} : x > h\right\}$. We list below the five experts that we designed to predict h half-hours ahead.

The Gam forecaster (GAM). We consider the following Generalized Additive Model that is inspired from the medium term forecaster designed by Pichavant [120] and analyzed in Thouvenot

et al. [135] :

$$Y_t = g_1(D_t) + g_2(M_t)\Delta T_t + f_1(Toy_t) + f_2(t) + f_3(N_t) + f_4(W_t) + f_5(T_t) + f_6(t\mathbb{1}_{\{T_t \leqslant 15\}}) + f_7(T_t^{(0.8)}) + f_8\big(T_t^{(\max)}\big) + f_9\big(T_t^{(\min)}\big) + f_{10}(Y_{t-H}) + f_{11}(Y_{t-h}) + \varepsilon_t \quad (9.1)$$

where ε_t is a i.i.d. Gaussian noise with zero mean, f_is are smooth functions that are estimated by penalized regression on B-spline basis as proposed in Wood [147] (see Section 8.2.1 for slightly more details), and g_1 and g_2 are two real functions that respectively take 23 and 12 values (i.e., one coefficient for each category of D_t for g_1 and one coefficient for each category of M_t for g_2). Each of these terms aims at modeling an effect. For instance, $f_1(Toy_t)$ estimates the annual seasonality, $f_2(t)$ estimates the trend, and $f_6(t\mathbb{1}_{\{T_t \leqslant 15\}})$ estimates the heating trend.

The gbm forecaster (GBM). It fits a `gbm` model (see Section 9.2.1) by using the same exogenous variables as in the Gam forecaster above (see Equation (9.1)). We recall that the model is fitted by using the default parameters of the `R`-package `gbm` and by optimizing the number of trees on the out-of-bag error by using the function `gbm.perf`.

The Random forest forecaster (RF). We fit a random forest regression (see Section 9.2.1) with the same covariates (see Equation (9.1)) and by using the default parameters of the `R`-package `randomForest`.

Gam mid-term followed by short term residual analysis (GamMTCT). First, the forecaster fits a medium term generalized additive model by removing the recent lagged electricity consumptions from Model (9.1). It is defined as:

$$Y_t = g_1(D_t) + g_2(M_t)\Delta T_t + f_1(Toy_t) + f_2(t) + f_3(N_t) + f_4(W_t) + f_5(T_t) + f_6(t\mathbb{1}_{\{T_t \leqslant 15\}}) + f_7(T_t^{(0.8)}) + f_8\big(T_t^{(\max)}\big) + f_9\big(T_t^{(\min)}\big) + \varepsilon_t$$

This generalized additive model corresponds to the one analyzed in Pichavant [120], Thouvenot et al. [135]. We denote by $\widehat{Y}_t$ the electric load fitted by the medium term model and by $\widehat{\varepsilon}_t = Y_t - \widehat{Y}_t$ the residual signal. Then, a short-term correction is performed assuming a short term `gam` structure of the medium term residuals:

$$\widehat{\varepsilon}_t = g_3(D_t) + f(\widehat{\varepsilon}_{t-H}, \widehat{\varepsilon}_{t-h}) + u_t$$

where u_t is a i.i.d. random noise, f is estimated by cubic regression splines, and g_3 are 23 daily coefficients to be estimated. In practice, we consider the estimated residuals $\widehat{\varepsilon}_t$ obtained after the medium term estimation process and fit the short term `gam` on the $30 \times 24 = 720$ last observations. This residual analysis, contrary to the medium-term model, is not performed separately for each half-hourly period.

Curve Linear Regression (CLR). We fit CLR as explained in Cho et al. [50]. In order to provide forecasts for horizons going from 1 hour ahead to 24 hours ahead, we use the method describe in the end of Section 9.3.3. For horizons of prediction ranging from 25 hours ahead to 48 hours ahead, we replace, in the process to fit the model, the residuals of the day before (i.e., the lagged consumptions from 1 to 24 hours) with the residuals of two days before (i.e., the lagged from 25 to 48 hours).

9.4.3. Results

We recall that the five experts designed in the previous section are fitted on the training set ranging from September 6, 2007 to August 31, 2012. Then, each of these five experts provides forecasts throughout the testing set, which goes from September 1, 2012 to July 27, 2013. The expert forecasts along the testing set are used to evaluate the respective performance of each expert and to learn online how to combine them.

Similarly to the heat load data set, we use the combining algorithms presented in Section 9.2.2: ML-Poly displayed in Algorithm 15, and Ridge recalled in Algorithm 16.

We display in Figure 9.11 the RMSEs suffered by each expert and by the combining algorithms along the testing set according to the horizon of prediction. We remark that both combining algorithms (ML-Poly and Ridge) perform better than the best expert (which is GamMTCT) for all horizons of predictions. Figure 9.12 plots the improvement on the accuracy of the best expert GamMTCT obtained by ML-Poly and Ridge. We see that the improvement decreases from more than 15% for very short horizons to approximatively 1% for a horizon of 48 hours. Though the improvement is not extremely important for large horizons the combining algorithms are fully automatic online methods to estimate online the best expert. Besides, the best expert seems to be (especially for large horizons) much better than the four other experts. Yet, both combining algorithm succeed to improve further its prediction accuracy.

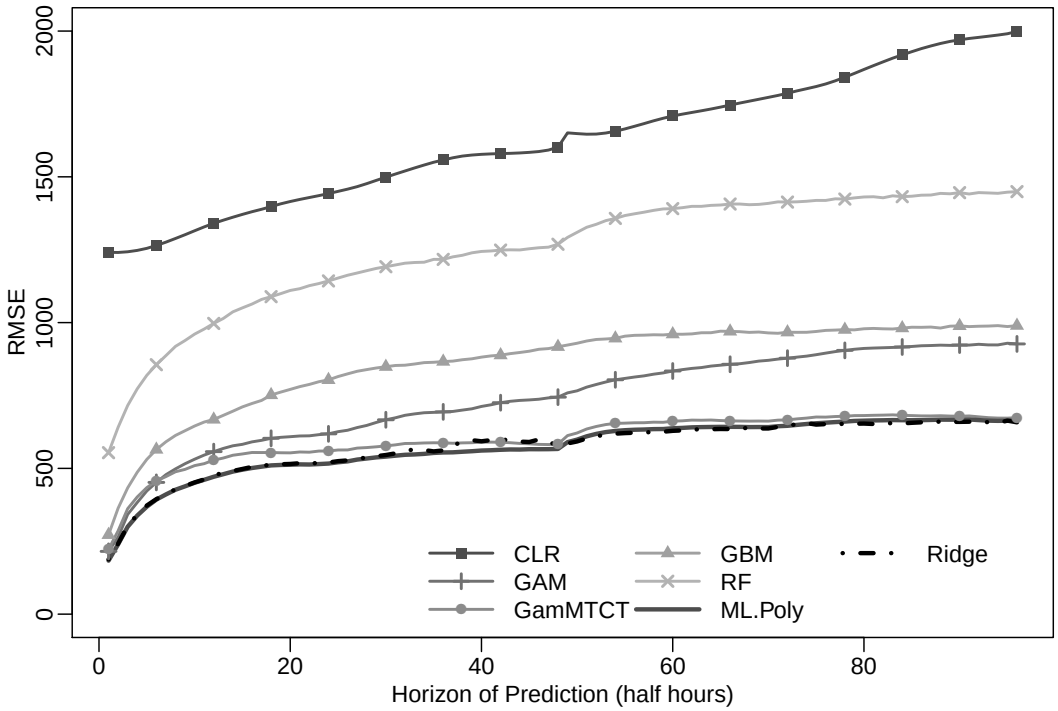


Figure 9.11.: Performance (RMSEs) obtained by the five experts on the testing set together with the performance obtained by the combining methods (ML-Poly and Ridge) according to the horizon of prediction.

Figure 9.13 plots the RMSEs suffered by the methods (experts and aggregation rules) according to the month. Once again both combining algorithms improve the performance for almost all months.

Since an expert (GamMTCM) seems to outperform all other experts a natural question is whether

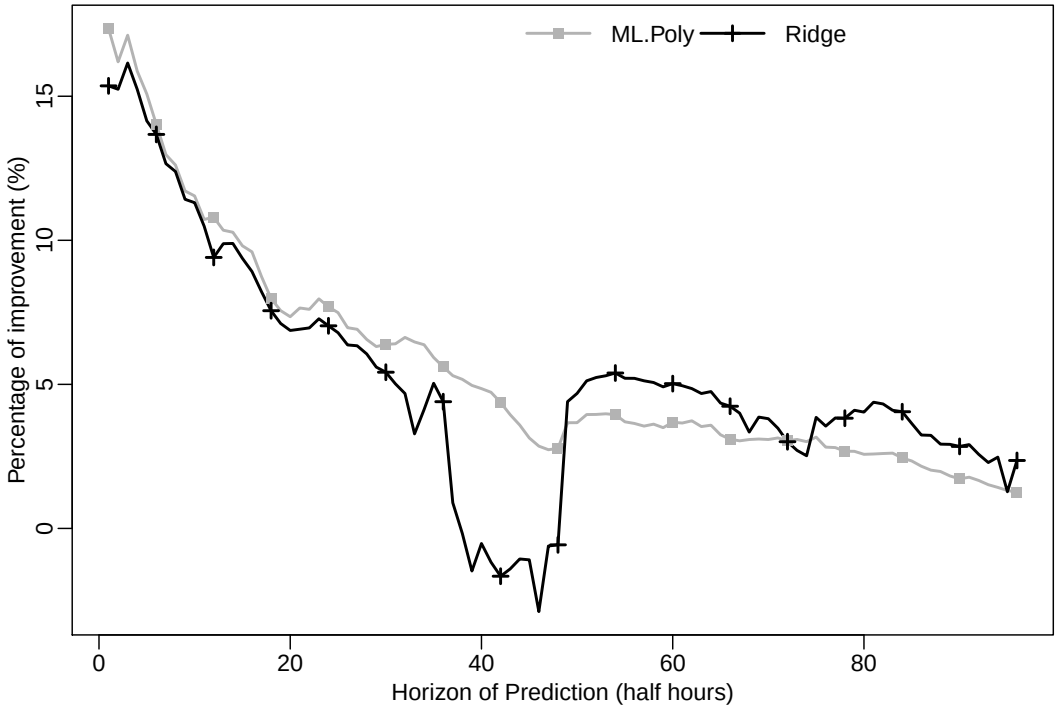


Figure 9.12.: Percentage of improvement of the RMSEs obtained by ML-Poly and Ridge compared to the best expert GamMTCT according to the horizon of prediction.

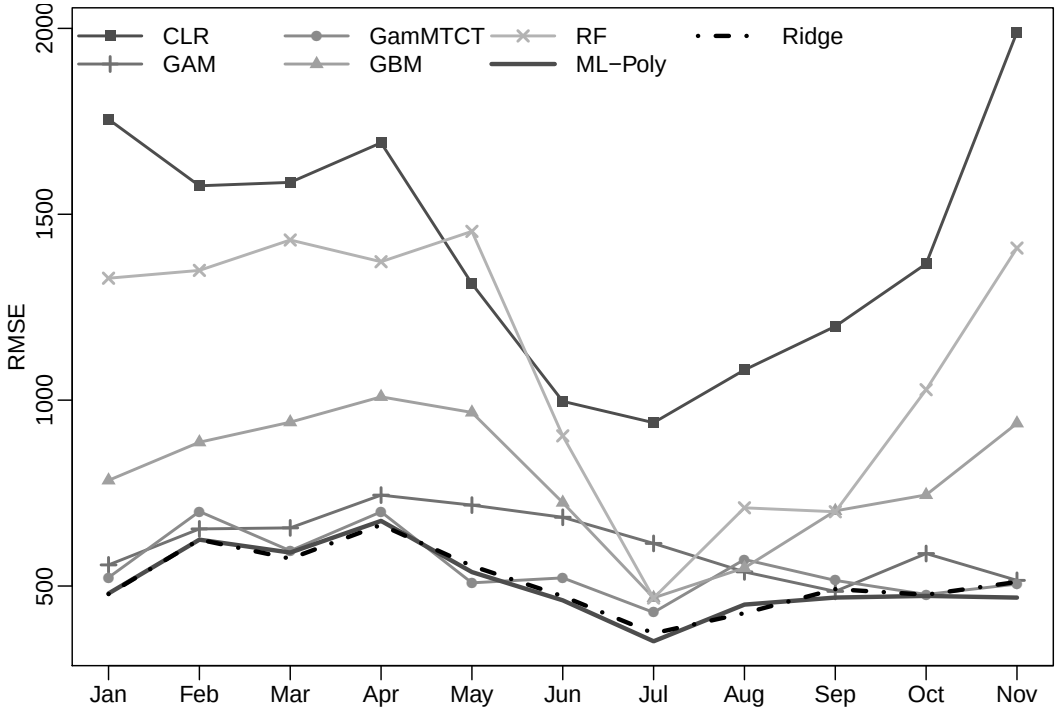
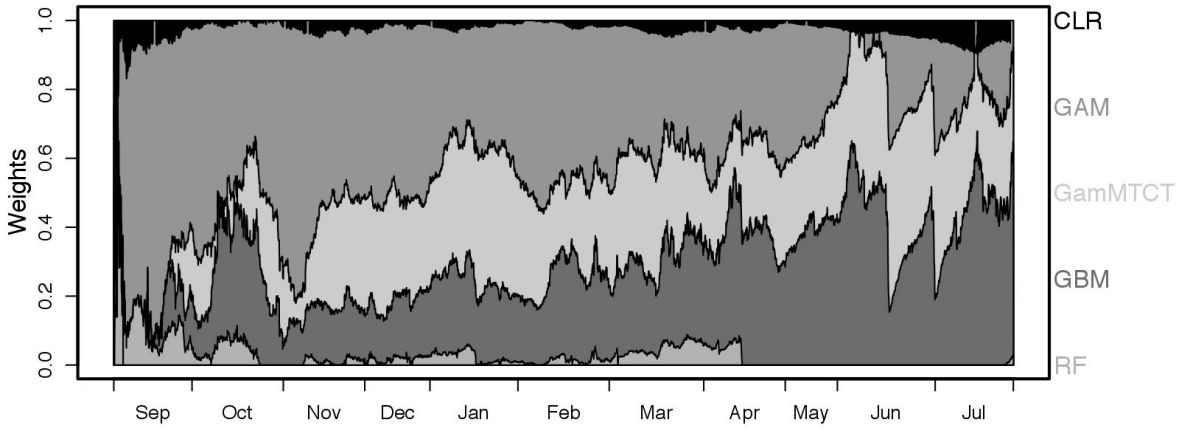
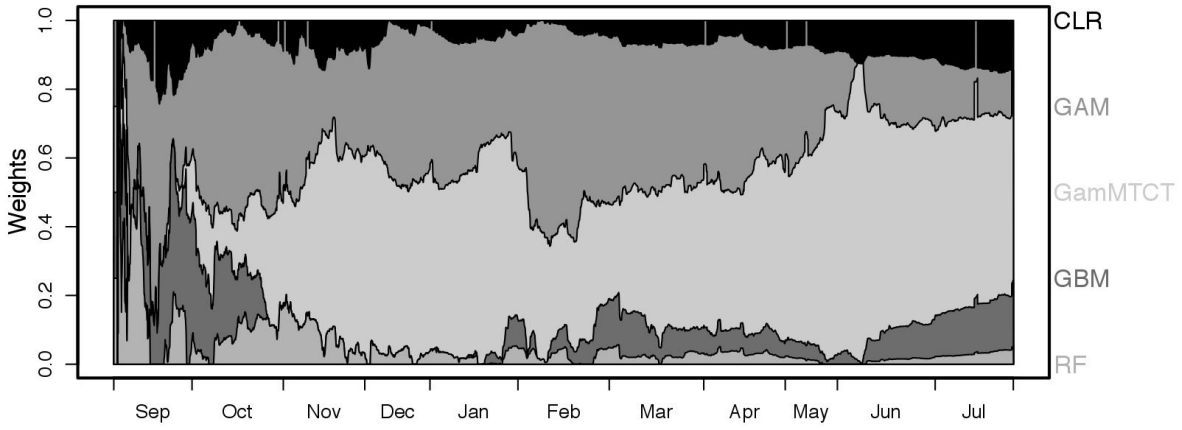


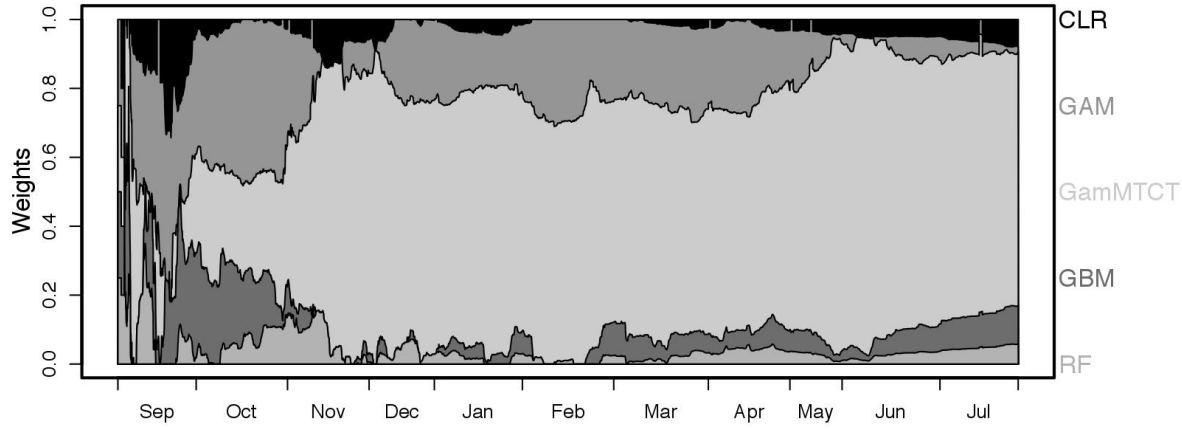
Figure 9.13.: RMSEs obtained by the experts and by ML-Poly and Ridge according to the month.



(a) 1 half-hour ahead

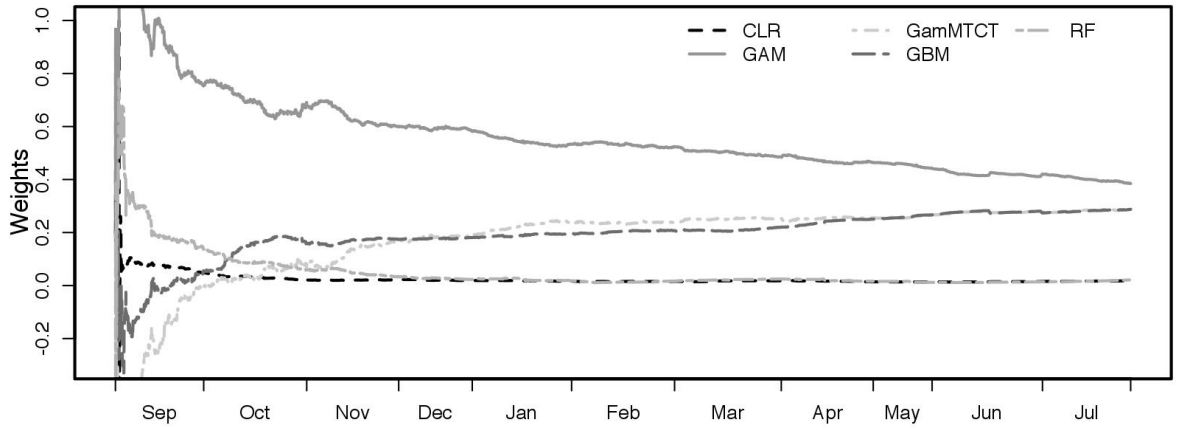


(b) 12 hours ahead

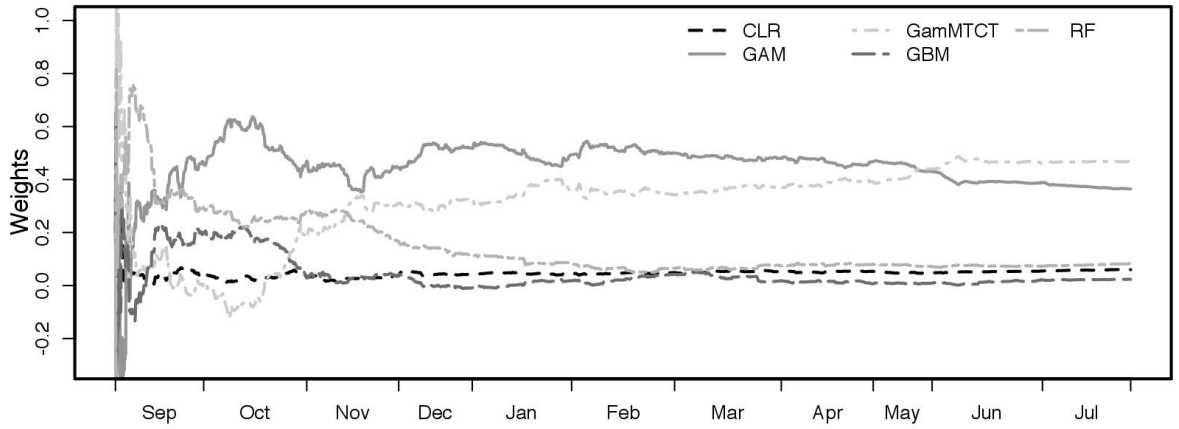


(c) 24 hours ahead

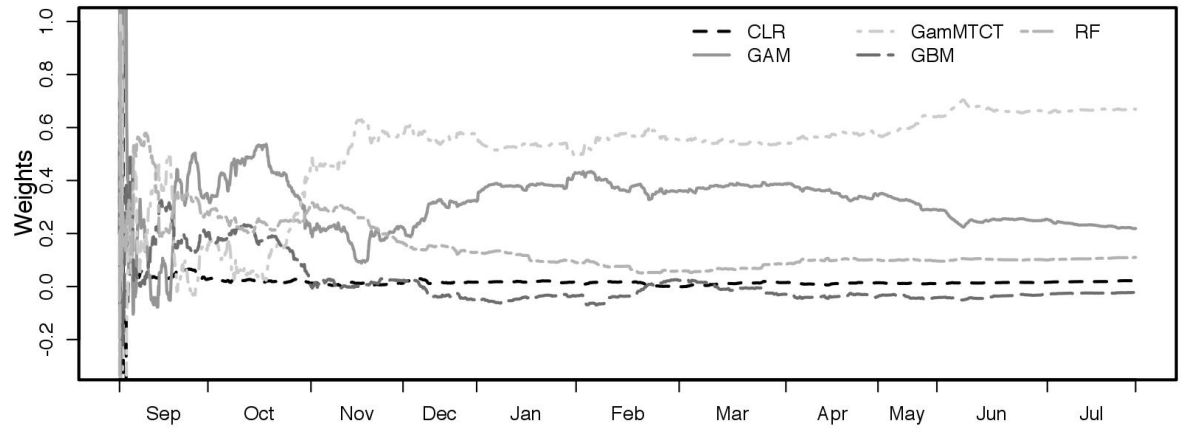
Figure 9.14.: Graphical representation of the weights formed by ML-Poly along the testing set (going from September 1, 2012 to July 27, 2013). The weight of each expert are assigned from bottom to top to RF (gray), GBM (dark gray), GamMTCM (light gray), GAM (gray), and CLR (black).



(a) 1 half-hour ahead



(b) 12 hours ahead



(c) 24 hours ahead

Figure 9.15.: Graphical representation of the weights formed by Ridge along the testing set (going from September 1, 2012 to July 27, 2013).

the combining algorithms concentrate their weights on this single expert GamMTCT or not. We plot in Figure 9.14 the time evolution of the weights formed by ML-Poly for three horizons of prediction (1 hour ahead, 24 hours ahead, and 48 hours ahead). Note that for all time instants, the weights of the five experts sum to 1 and are all positive. This is not the case with Ridge, whose weights are displayed in Figure 9.14.

The larger the horizon of prediction is, the larger is the difference in performance between GamMTCT and the other experts (see Figure 9.11), and the more weight is assigned to GamMTCM by ML-Poly. We observe the same behaviors as for the heat load data set: Ridge produces weights that are much more stable over time. However, Ridge may thus be less adaptive to a changing environment. Furthermore, we see that both algorithms fully exploit the full set of experts and do not concentrate only on the best expert GamMTCT.

9.5. Conclusion

All in all, the combining algorithms ML-Poly and Ridge perform well on both data sets. They can compete online with the best convex weight combination of experts and this for all horizons of prediction in a fully automatic fashion. They improve the accuracy (in RMSEs) of the best expert by about 5% for the electric load data set to about 10% for the heat load data set. It seems that the smaller the horizon of prediction is, the more benefit we gain in combining the methods online.

10

Aggregate sub-model predictions

In this chapter, we study an electricity load data set that consists of a global electricity consumption signal together with consumption signals of smaller groups of customers (henceforth referred to as low-level signals). The goal is to compare the performance attained by forecasting the global consumption directly using only the global signal with the performance obtained by predicting all low-level signals separately before aggregating them online with a combining algorithm. We show a performance gain of about 7% by combining the sub-models online.

Contents

NOTA: This chapter is a based on the internal study [9] carried out for EDF.

10.1. Introduction

In this thesis we mostly use combining algorithms to perform convex aggregation between a set of individual forecasters, where each forecaster was previously designed to predict the global signal Y_t. However, in some cases the individual forecasters may not forecast Y_t so that convex aggregation makes no sense anymore. Here we study such situation and we show that online aggregation can still be a powerful tool.

10.1.1. Setting and motivation

We consider a sequence of nonnegative real observations $(y_t)_{t\geqslant 1}$ that needs to be predicted online. Now, suppose that we also observe K low-level sequences $\left(y_t^{(k)}\right)_{t\geqslant 1}$ for $k = 1, \dots, K$ that satisfy for all times $t \geqslant 1$

$$y_t = y_t^{(1)} + \cdots + y_t^{(k)} \,.$$

In order to predict the observations y_t, we have two alternatives. Either we design a forecasting model that directly estimates the observations y_t or we build K separate forecasting sub-models that aim to estimate the low-level sequences $\left(y_t^{(k)}\right)$ before aggregating them. The natural approach to aggregate the sub-models is to assign a weight 1 to each of them. However, we will show in this chapter that the weights can be learned sequentially. Many real world applications match this setting.

Example 10.1. An example was provided by the the hierarchical load forecasting track of the Global Energy Forecasting Competition 2012 (see Hong et al. [95]). The participants were required to backcast and forecast hourly loads for a US area with 20 geographical zones at both the zonal (20 time series) and global (sum of the 20 zonal level series) levels.

Example 10.2. Within the next few years, the French electric network operator will be able to record the individual electricity consumption of all French households (i.e., 30 million of individual loads) in real time (see Pompey et al. [122]). Exploiting such amount of information in order to improve the forecasting accuracy of the global electric demand in France will be a real challenge. A possible solution may be to cluster similar individual loads in groups of customers large enough to be predictable before aggregating the predictions of all groups.

In this study we consider a less ambitious goal. The total energy consumption of the EDF Market in France is parsed into seven signals corresponding to subgroups of customers. We show that predicting each low-level signal separately before aggregating them improves the forecasting performance of the global consumption by about 7% compared to predicting the global signal directly.

10.1.2. Data set description

The data set includes half hourly observations of the total energy consumption of the EDF Market in France between January 1, 2008 to June 15, 2012. The data set also contains with the same frequency the electric load of seven groups of customers listed below:

- Individuals: most of small EDF consumers (for instance households);
- NOP.SPE, TVsup7, and 32 0000: three groups of big electricity consumers (such as companies), not sensible to temperature, and clustered by pricing policy;

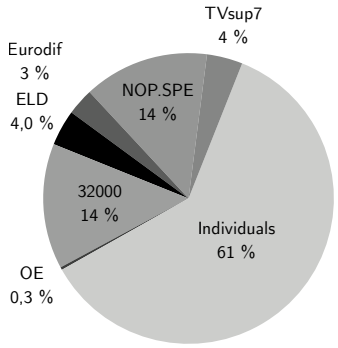
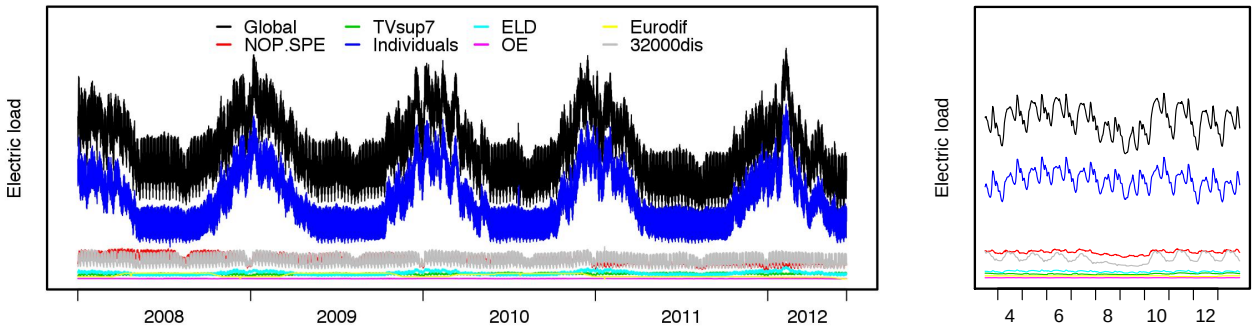


Figure 10.1.: Percentage of consumption of each group of customers.



(a) From January 1, 2008 to June 15, 2012

(b) From March 3, 2008 to March 13, 2008

Figure 10.2.: Half-hourly observations of the seven individual electricity consumptions together with the total electric demand in France in black (sum of the 7 individual series).

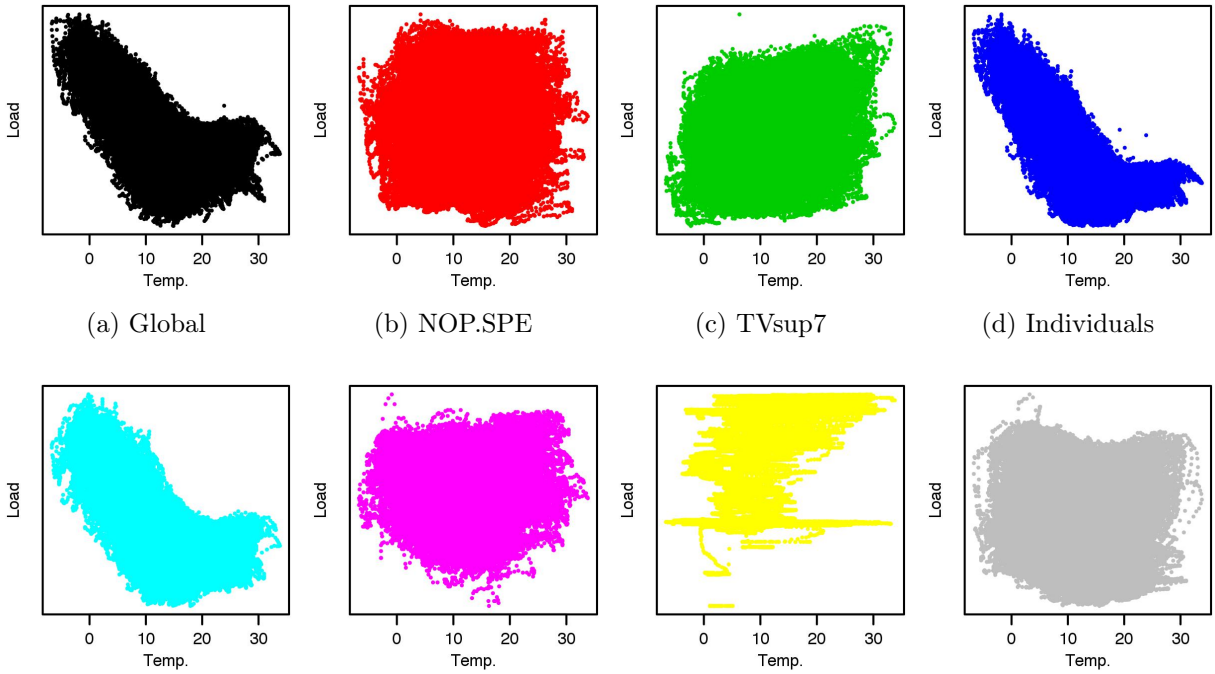
- ELD: some companies that locally provide electricity.
- Eurodif: the biggest EDF consumer.
- OE: foreign operators.

The rate of consumption for each group of customers is displayed in Figure 10.1. Covariates that have been shown to impact the electric demand are also provided. They include meteorological covariates (such as temperature, wind, and cloud cover in France), calendar covariates (such as public holidays), and additional covariates (such as the number of French customers). The goal is to predict a day ahead (48 half hours in advance) the electricity consumption of the global signal.

We exclude from the data set some special days corresponding to public holidays and the days before and after them. The data set is then partitioned into two pieces. The first one, the training set, consists of 1 136 days from January 1, 2008 to August 31, 2011. It will be used to estimate the models. The second set is called the testing set. It contains 243 days from September 1, 2011 to June 15, 2012 and is used to evaluate the performance. Prediction accuracy is measured by the root mean square error (RMSE) and by the mean absolute percentage error (MAPE) (see Equations (4.2) and (4.3)).

Figure 10.2 displays the signal for each group together with the global load (i.e., the sum of the seven series). We observe that the electricity consumption of Individuals is dominant in the total consumption. The six other signals are much smaller and show different behaviors.

Figure 10.3 depicts the impact of temperature on the electricity consumptions. The dependency on temperature is highly different between the groups: some groups are not dependent on temperature (such as NOP.SPE, OE, or TVsup7) while others show a notable linear dependency for low temperatures because of heating (such as Individuals or ELD). We hope that catching these different dependencies into each sub-models may improve the accuracy of the global forecasting procedure. Remark that the electricity consumption of Eurodif only takes a finite number of levels.



(a) Global (b) NOP.SPE (c) TVsup7 (d) Individuals

(e) ELD (f) OE (g) Eurodif (h) 32000

Figure 10.3.: Electric load according to temperature (^{o}C) for each group of consumers.

10.2. Methodology and performance

10.2.1. Methodology

We briefly describe the methodology employed in this empirical study. First we model the electric load of each customer group and the global consumption. The same generalized additive model (10.2) is used for each time series (except for Eurodif). It is described in Section 10.A. The eight models, also referred to as experts, are fitted using the training set. They are used to produce forecasts throughout the testing set. In the end we show that combining the expert forecasts online improves the forecasting performance. The experts are aggregated by using the ridge regression forecaster detailed in Section 9.2.2.

10.2.2. Expert performance

Table 10.1 reports the performance obtained for all low-level signals by the sub-models described in Section 10.A. We remark that the different groups of customers present contrasting levels of

difficulty with MAPEs ranging from 1.4% for the global load to 11.9% for the 32 000 group. The RMSE of the global series is smaller than the ones of the Individuals and 32 000 loads. The forecasting errors of all groups will hopefully balance out.

Signal	RMSE (MW)	MAPE (%)
Global	864	1.41
NOP.SPE	495	7.95
TVsup7	113	4.94
Individuals	1 243	3.02
ELD	133	4.71
OE	17.2	9.46
Eurodif	11.8	2.89
32 000	989	11.9

Table 10.1.: Performance obtained by the generalized additive models (10.2) on the testing set for the different signals.

10.2.3. Uniform strategies

Since the global electric load is the sum of the other seven time series, we may want to predict it by summing the predictions of the seven individual models. We obtain a deceiving RMSE of 1 438 MW while its MAPE approximatively equals 2.4%. We call this forecaster Uniform7.

Uniform7 does not use the Global forecaster which reaches an excellent performance. We thus define Uniform8 a second uniform forecaster which is the uniform average of Global and Uniform7. Put differently, Uniform8 is a combination of the eight experts of Table 10.1 with weight vector $\boldsymbol{u}_0 \overset{\text{def}}{=} (1/2, \ldots, 1/2) \in \mathbb{R}^8$. Uniform8 achieves a RMSE of 1 045 MW and a MAPE of 1.7% (see Table 10.2) which is better than Uniform7 but again not satisfactory.

The performance of Uniform7 and Uniform8 are recalled in Table 10.2 together with the results of the following combining algorithms.

10.2.4. Performance of online robust aggregation

Now, we show that we can still take benefit of the individual forecasts by learning the weighted combination throughout the testing set sequentially. Since the weighted combination is not convex (i.e., the sum of the weights does not sum to 1) we use the ridge regression forecaster (see Algorithm 16 and Section 9.2.2 for details). Other aggregation rules are considered later on in Remark 10.4 but they obtain poor performance.

We initialize the weight vector with $\boldsymbol{u}_0$ (we include the global forecaster in the aggregation procedure). The RMSE obtained along the testing set is about 805 MW which improves the performance by 6.8% when compared to the generalized additive model applied to the global signal directly (see Performance of Global in Table 10.1). Its MAPE equals 1.35%. Performance is summarized in Table 10.2.

Ridge is compared to the best linear combination of experts constant over time computed in hind-

Forecaster	RMSE (MW)	MAPE (%)
Uniform7	1 438	2.41
Uniform8	1 045	1.74
Best linear	776	1.32
Ridge	805	1.35

Table 10.2.: Performance obtained by the uniform forecaster (uniform sum of the low level forecasts), the ridge regression forecaster, and the best constant linear combination of the experts.

sight. The best linear combination equals

$$\boldsymbol{u}^{(\text{best})} = (0.92, 0.07, -0.16, 0.10, 0.14, 0.36, 0.17, 0.03) \in \mathbb{R}^8 .$$

It performs slightly better than Ridge (see in Table 10.2). Global (the first expert) receives the preponderant weight 0.92, the seven sub-models are used to correct some errors.

The weights assigned by Ridge to the experts are depicted in Figure 10.4. They are highly variable at the beginning of the testing set and then slowly converge to $\boldsymbol{u}^{(\text{best})}$.

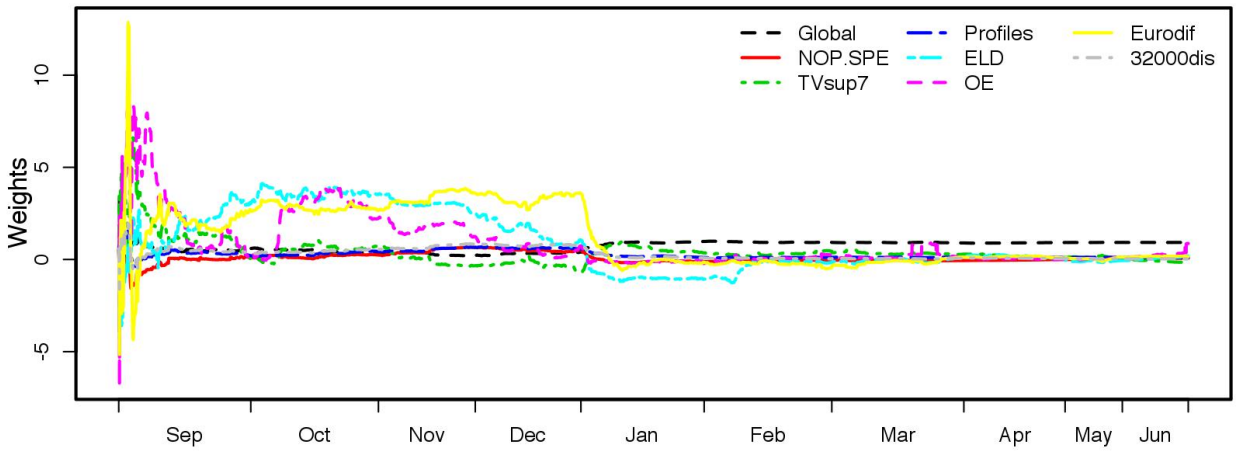


Figure 10.4.: Graphical representation of the weights formed by Ridge along the testing set.

Remark 10.3. We also tried to aggregate different sets of experts with Ridge:

- a set of seven experts: combining the seven low-level experts (i.e., we excluded the global model from the set of experts) yields a RMSE of 1 124 and a MAPE of 1.6%. It performs better than Uniform7 but it is less efficient than the Global forecaster.
- a set of two experts: combining the Global model with Uniform7 suffers a RMSE of 820 MW and a MAPE of 1.38%.

Remark 10.4. Convex aggregation rules such as the exponentially weighted average forecaster, the fixed-share forecaster, or ML-Poly (see Section 4.2.2 for definitions) can be considered instead of Ridge. However to do so we need to scale the expert predictions so that they are of the same order of magnitude as the global consumptions y_t.

More formally fix $\boldsymbol{\beta} \in \mathbb{R}_+^8$ a scaling vector. At each time step $t \geqslant 1$ we denote by $\boldsymbol{x}_t^{(\boldsymbol{\beta})} \overset{\text{def}}{=} (\beta_1 x_{t,1}, \dots, \beta_8 x_{t,8})$ the vector of scaled predictions where $x_{t,1}$ is the prediction of the global model and $x_{t,2}, \dots, x_{t,8}$ are the low-level forecasts. The idea is to substitute in the convex aggregation rules the expert predictions $\boldsymbol{x}_t$ with the scaled ones $\boldsymbol{x}_t^{(\boldsymbol{\beta})}$. If $\boldsymbol{\beta}$ is well-chosen, the approximation error

$$\min_{\boldsymbol{q} \in \Delta_8} \left\{ \sum_{t=1}^{T} \left(y_t - \boldsymbol{q}^\top \boldsymbol{x}_t^{(\boldsymbol{\beta})} \right)^2 \right\} \tag{10.1}$$

will be small and the convex algorithm may perform well. We recall that y_t denotes the global electric load observed at time t and $\Delta_8 \overset{\text{def}}{=} \left\{ \boldsymbol{p} \in \mathbb{R}_+^8 : \sum_{i=1}^8 p_i = 1 \right\}$ is the simplex. Unfortunately for most natural choices of $\boldsymbol{\beta} \in \mathbb{R}^8$ non significant improvements were observed for all convex aggregation rules.

The best results were obtained with $\boldsymbol{\beta} = (1, 7, \dots, 7)$. This choice is based on the fact that the predictions of Uniform7, Uniform8, and the Global forecaster match a convex combination $\boldsymbol{q} \in \mathbb{R}^8$ of $\boldsymbol{x}_t^{(\boldsymbol{\beta})}$ fixed over time. Indeed, the forecast output at time t by Uniform7 (resp. Uniform8 and Global) equals $\boldsymbol{q}^\top \boldsymbol{x}_t^{(\boldsymbol{\beta})}$ with $\boldsymbol{q} = (0, 1/7, \dots, 1/7)$ (resp. $(1/2, \dots, 1/2)$ and $(1, 0, \dots, 0)$). Therefore the approximation error (10.1) is ensured to be small for this choice of $\boldsymbol{\beta}$.

The best fixed convex combination (the solution of (10.1) with $\boldsymbol{\beta} = (1, 7, \dots, 7)$) achieves a RMSE of 778 MW and a MAPE of 1.32%. It is similar to the performance obtained by the best fixed linear combination of the experts (see Table 10.2). The combining forecasts obtained by ML-Poly (described in Algorithm 15 in Chapter 9) suffer a RMSE of 835 MW and a MAPE of 1.39%. If we resort to the gradient trick (see 2.C.1) Fixed-Share and EWA perform extremely badly with RMSEs exceeding 2 000 MW. It we do not, their predictions converge to the ones of the global forecaster and incur the same errors.

10.2.5. Conclusion

All in all, we showed that we could improve the prediction of the global signal by aggregating low-level forecasts online with a fully automatic procedure. Our method can surely be highly improved in the future by designing specific forecasting models for each low-level signal.

10.A. Appendix: Underlying statistical models used by the experts

Here we give a brief description of the underlying statistical models used to create the eight experts. With the exception of Eurodif that we will specifically detail later, we model as usual the electric load of each group of customers with a generalized additive model. We refer the curious reader to Section 8.2.1 and to [87, 147] for details. Because the electric load heavily depends on the hour of the day, the generalized additive models presented below are half-hourly performed. The data is parsed into 48 separate time series (one for each half hour of the day) and 48 separate models are fitted. For the sake of simplicity, we do not write hereinafter the dependencies of the parameters on the hour of the day. We estimate the consumption of each signal with the following generalized additive model:

$$Y_t = h(D_t) + \alpha E_t + f_1(Toy_t) + f_2(T_t) + f_3(T_{t-48}) + f_4(C_t) + f_5(W_t) + f_6(F_t) + f_7(t) + f_8(Y_{t-48}) + \varepsilon_t \,, \tag{10.2}$$

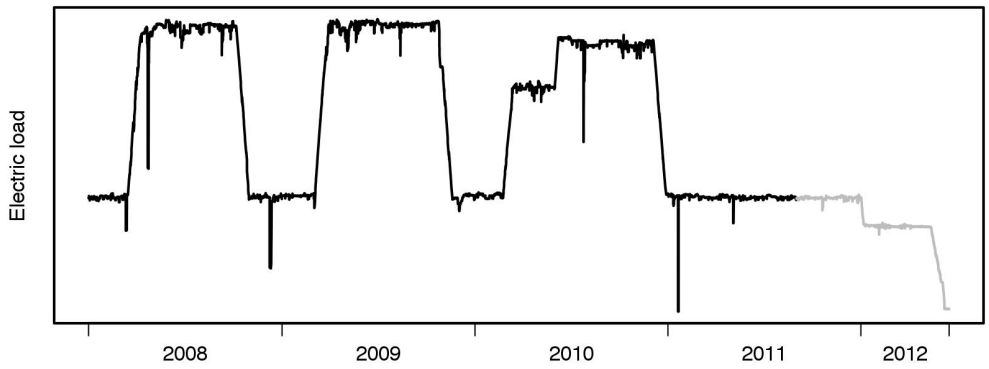


Figure 10.5.: Half-hourly observation of the electricity consumption of Eurodif from January 1, 2008 to June 15, 2012.

where ε_t are Gaussian i.i.d random variables with zero mean and

- Y_t is the electric load of the considered group at time t (all variables are indexed by the time of the observation);
- D_t is a factorial variable with 7 levels corresponding to different types of day. The levels are: Monday, Tuesday-Wednesday-Thursday, Friday, Saturday, Sunday, bank holidays, and a last category corresponding to the days before and after bank holidays. This choice was driven by our expertise on electricity load data (see e.g., Goude et al. [81]).
- E_t is a level provided by EDF that depends on the price policy of the day;
- $Toy_t \in [0, 1]$ (Time of year) is a cyclic variable that indicates the annual position and repeats each year. It is each year linearly increasing over time going from 0 on January 1 at 00am to 1 on December 31 at 12pm, each hour having a different value;
- T_t, C_t, and W_t are respectively a spacial average of the temperatures, the cloud cover, and the wind velocity in France;
- F_t denotes the number of EDF customers.

Besides, the function h corresponds to seven real coefficients (one for each level of day D_t) and the f_i are cubic regression splines that are estimated by the `R`-package `mgcv` (see Wood [147]) by solving a regularized optimization problem. Remark that $f_1(Toy_t)$ aims at estimating the annual seasonality while $f_7(t)$ estimates the trend of the signal.

The Eurodif series is special as depicted in Figure 10.5. The generalized additive model (10.2) performs extremely badly on this signal with a MAPE of 146%. This is partly due to the significant decrease of the consumption in the testing set. As the electric consumption of Eurodif seems to be approximatively piecewise constant over time, we forecast it by using the last available lag of consumption. In other words, the consumption Y_t at time t is predicted 48 half-hours ahead by Y_{t-48}. This autoregressive model suffers a MAPE of 2.9% (see Table 10.1).

Appendix: Package `opera`

The following pages are the documentation files of the package `opera`. The package will continue to be developed to add new aggregation rules and new features.

Type Package
Title Online Prediction by ExpeRts Aggregation
Version 0.01
Date 2015-09-10
Author Pierre Gaillard
Maintainer Pierre Gaillard `<pierre@gaillard.me>`
Copyright EDF R&D 2015
Description This package implements, for regression-oriented time-series, online predictions by combining a finite set of forecasts provided by the user.
License To be decided
Depends R (>= 3.1.0)
Imports quadprog, quantreg
Suggests testthat, caret, mgcv, survival,knitr
LazyData true
VignetteBuilder knitr

Contents

`opera-package` *Online Prediction by ExpeRts Aggregation*

Description

For regression-oriented time-series, the package `opera` performs predictions by combining a finite set of forecasts provided by the user. More formally, it considers a sequence of observations $y_1, \ldots, y_T$ (such as electricity consumption or any bounded time series) to be predicted element by element. At each time instance $t \geqslant 1$, a finite set of experts (basically some based forecasters) provide predictions $x_{k,t}$ of the next observation y_t. This package proposes several adaptive and robust methods to combine the expert forecasts based on their past performance.

References

Cesa-Bianchi and Lugosi [43], Devaine et al. [60], and Gaillard et al. [77]

Examples

```
library('opera')  # load the package
set.seed(1)

# Example: find the best one week ahead forecasting strategy (weekly data)
# packages
library(mgcv)
library(caret)

# import data
data(electric_load)
idx_data_test <- 680:nrow(electric_load)
data_train <- electric_load[-idx_data_test, ]
data_test <- electric_load[idx_data_test, ]

# Medium term model to remove trend and seasonality (using generalized additive
    model)
detrend.fit <- gam(Load ~ s(Time,k=3) + s(NumWeek) + s(Temp) + s(IPI), data =
    data_train)
electric_load$Trend <- c(predict(detrend.fit), predict(detrend.fit ,newdata =
    data_test))
electric_load$Load.detrend <- electric_load$Load - electric_load$Trend

# a few graphs to display the data
attach(data_train)
plot(Load, type = 'l')
plot(Temp, Load, pch = 16, cex = 0.5)
plot(NumWeek, Load, pch = 16, cex = 0.5)
plot(Load, Load1, pch = 16, cex = 0.5)
acf(Load, lag.max = 20)
detach(data_train)

# Build the expert forecasts
# ############################

# A generalized additive model
gam.fit <- gam(Load ~ s(IPI) + s(Temp) + s(Time, k=3) +
```

```
                s(Load1) + as.factor(NumWeek), data = data_train)
gam.forecast <- predict(gam.fit, newdata = data_test)

# An online autoregressive model on the residuals of the medium term model
ar.forecast <- numeric(length(idx_data_test))
for (i in seq(idx_data_test)) {
 ar.fit <- ar(electric_load$Load.detrend[1:(idx_data_test[i] - 1)])
 ar.forecast[i] <- as.numeric(predict(ar.fit)$pred) + electric_load$Trend[idx_
     data_test[i]]
}

# A GBM
gbm0.fit <- train(Load ~ IPI + IPI_CVS + Temp + Temp1 + Time + Load1 + NumWeek,
                  data = data_train, method = 'gbm')
gbm.forecast <- predict(gbm0.fit, newdata = data_test)


# Aggregation of experts
################################

X <- cbind(gam.forecast, ar.forecast, gbm.forecast)
colnames(X) <- c('gam', 'ar', 'gbm')
Y <- data_test$Load

matplot(cbind(Y, X), type = 'l', col = 1:6, ylab = 'Weekly load', xlab = 'Week')


# How good are the expert? Look at the oracles
oracle.convex <- oracle(Y = Y, experts = X, loss.type = 'square', model = '
    convex')
plot(oracle.convex)
oracle.convex

# Is a single expert the best over time ? Are there breaks ?
oracle.shift <- oracle(Y = Y, experts = X, loss.type = 'percentage', model = '
    shifting')
plot(oracle.shift)
oracle.shift

# Online aggregation of the experts with MLpol
#########################################################

# Initialize the aggregation rule
m0.MLpol <- mixture(model = 'MLpol', loss.type = 'square')

# Perform online prediction using MLpol There are 3 equivalent possibilities 1)
# start with an empty model and update the model sequentially
m1.MLpol <- m0.MLpol
for (i in 1:length(Y)) {
 m1.MLpol <- predict(m1.MLpol, newexperts = X[i, ], newY = Y[i])
}

# 2) perform online prediction directly from the empty model
m2.MLpol <- predict(m0.MLpol, newexpert = X, newY = Y, online = TRUE)

# 3) perform the online aggregation directly
```

```
m3.MLpol <- mixture(Y = Y, experts = X, model = 'MLpol', loss.type = 'square')

# These predictions are equivalent:
identical(m1.MLpol, m2.MLpol)  # TRUE
identical(m1.MLpol, m3.MLpol)  # TRUE

# Display the results
summary(m3.MLpol)
plot(m3.MLpol)
```

`electric_load` *Electricity forecasting data set*

Description

Electricity forecasting data set provided by EDF R&D. It contains weekly measurements of the total electricity consumption in France from 1996 to 2009, together with several covariates, including temperature and industrial production indexes.

Usage

```
data(electric_load)
```

Format

```
'data.frame':  731 obs. of  11 variables:
 $ Time   : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Day    : int  1 8 15 22 29 5 12 19 26 4 ...
 $ Month  : int  1 1 1 1 1 2 2 2 2 3 ...
 $ Year   : int  1996 1996 1996 1996 1996 1996 1996 1996 1996 1996 ...
 $ NumWeek: num  0.0189 0.0377 0.0566 0.0755 0.0943 ...
 $ Load   : num  51306 51072 54407 56738 53877 ...
 $ Load1  : num  61395 51306 51072 54407 56738 ...
 $ Temp   : num  5.86 8.49 4.42 3.65 2.83 ...
 $ Temp1  : num  -0.313 5.862 8.489 4.419 3.649 ...
 $ IPI    : num  90.1 90.1 90.1 90.1 90.1 88.4 88.4 88.4 88.4 93 ...
 $ IPI_CVS: num  87.9 87.9 87.9 87.9 87.9 88.1 88.1 88.1 88.1 88.6 ...
```

Examples

```
data(electric_load)
 # a few graphs to display the data
 attach(electric_load)
 plot(Load, type = 'l')
 plot(Temp, Load, pch = 16, cex = 0.5)
 plot(NumWeek, Load, pch = 16, cex = 0.5)
 plot(Load, Load1, pch = 16, cex = 0.5)
 acf(Load, lag.max = 20)
 detach(electric_load)
```

loss *Errors suffered by a sequence of predictions*

Description

The function `loss` computes the sequence of instantaneous losses suffered by the predictions in `x` to predict the observation in `y`.

Usage

```
loss(x, y, loss.type = "square")
```

Arguments

`x`	A vector of length `T` containing the sequence of prediction to be evaluated.
`y`	A vector of length `T` that contains the observations to be predicted.
`loss.type`	A string or a list with a component 'name' specifying the loss function considered to evaluate the performance. It can be 'square', 'absolute', 'percentage', or 'pinball'. In the case of the pinball loss, the quantile can be provided by assigning to loss.type a list of two elements: **name** A string defining the name of the loss function (i.e., 'pinball') **tau** A number in `[0,1]` defining the quantile to be predicted. The default value is 0.5 to predict the median.

Value

A vector of length `T` containing the sequence of instantaneous losses suffered by the prediction `x`.

mixture *Compute an aggregation rule*

Description

The function `mixture` builds an aggregation rule chosen by the user. It can then be used to predict new observations Y sequentially. If observations `Y` and expert advice `experts` are provided, `mixture` is trained by predicting the observations in `Y` sequentially with the help of the expert advice in `experts`. At each time instance `t`, the mixture forms a prediction by assigning a weight to each expert and by combining the expert advice.

Usage

```
mixture(Y = NULL, experts = NULL, model = "MLpol", loss.type = "square",
  loss.gradient = TRUE, coefficients = "Uniform", awake = NULL,
  parameters = list())
```

Arguments

`Y`	A vector of length T (a non negative integer) containing the observations to be predicted sequentially in order to train the aggregation rule.

`experts` A matrix containing the expert forecasts. Each column corresponds to the predictions proposed by an expert to predict `Y`. It has as many columns as there are experts. Its number of row should be `T`.

`model` A character string specifying the aggregation rule to use. Currently available aggregation rules are:

'EWA' Exponentially weighted average aggregation rule. A positive learning rate **eta** can be chosen by the user. The bigger it is the faster the aggregation rule will learn from observations and experts performances. However, too hight values lead to unstable weight vectors and thus unstable predictions. If it is not specified, the learning rate is calibrated online. A finite grid of potential learning rates to be optimized online can be specified with **grid.eta**.

'FS' Fixed-share aggregation rule. As for `ewa`, a learning rate **eta** can be chosen by the user or calibrated online. The main difference with `ewa` aggregation rule rely in the mixing rate **alpha**$\in [0, 1]$ which considers at each instance a small probability `alpha` to have a rupture in the sequence and that the best expert may change. Fixed-share aggregation rule can thus compete with the best sequence of experts that can change a few times (see), while `ewa` can only compete with the best fixed expert. The mixing rate **alpha** is either chosen by the user either calibrated online. Finite grids of learning rates and mixing rates to be optimized can be specified with parameters **grid.eta** and **grid.alpha**.

'Ridge' Ridge regression. It minimizes at each instance a penalized criterion. It forms at each instance linear combination of the experts' forecasts and can assign negative weights that not necessarily sum to one. It is useful if the experts are biased or correlated. It cannot be used with specialized experts. A positive regularization coefficient **lambda** can either be chosen by the user or calibrated online. A finite grid of coefficient to be optimized can be specified with a parameter **grid.lambda**.

'MLpol' Polynomial Potential aggregation rule with different learning rates for each expert. The learning rates are calibrated using theoretical values. There are similar aggregation rules like 'BOA' (Bernstein online Aggregation see [Wintenberger, 2014] 'MLewa', and 'MLprod' (see Gaillard et al. [77])

`loss.type` A string or a list with a component 'name' specifying the loss function considered to evaluate the performance. It can be 'square', 'absolute', 'percentage', or 'pinball'. In the case of the pinball loss, the quantile can be provided by assigning to loss.type a list of two elements:

name A string defining the name of the loss function (i.e., 'pinball')

tau A number in `[0,1]` defining the quantile to be predicted. The default value is 0.5 to predict the median.

Ridge is restricted to square loss.

`loss.gradient`

A boolean. If TRUE (default) the aggregation rule will not be directly applied to the loss function at hand but to a gradient version of it. The aggregation

rule is then similar to gradient descent aggregation rule.

`coefficients` A vector containing the prior weights of the experts (not possible for 'MLpol').

`awake` A matrix specifying the activation coefficients of the experts. Its entries lie in `[0,1]`. Possible if some experts are specialists and do not always form and suggest prediction. If the expert number `k` at instance `t` does not form any prediction of observation `Y_t`, we can put `awake[t,k]=0` so that the mixture does not consider expert `k` in the mixture to predict `Y_t`.

`parameters` A list that contains optional parameters for the aggregation rule. If no parameters are provided, the aggregation rule is fully calibrated online. Possible parameters are:

eta A positive number defining the learning rate. Possible if model is either 'EWA' or 'FS'

grid.eta A vector of positive numbers defining potential learning rates for 'EWA' of 'FS'. The learning rate is then calibrated by sequentially optimizing the parameter in the grid. The grid may be extended online if needed by the aggregation rule.

gamma A positive number defining the exponential step of extension of grid.eta when it is needed. The default value is 2.

alpha A number in [0,1] defining the mixing rate for 'FS'.

grid.alpha A vector of numbers in [0,1] defining potential mixing rates for 'FS' to be optimized online. The grid is fixed over time. The default value is $[0.0001, 0.001, 0.01, 0.1]$.

lambda A positive number defining the smoothing parameter of 'Ridge' aggregation rule.

grid.lambda Similar to `grid.eta` for the parameter `lambda`.

Value

An object of class mixture that can be used to perform new predictions. It contains the parameters `model`, `loss.type`, `loss.gradient`, `experts`, `Y`, `awake`, and the fields

`coefficients` A vector of coefficients assigned to each expert to perform the next prediction.

`weights` A matrix of dimension `c(T,N)`, with `T` the number of instances to be predicted and `N` the number of experts. Each row contains the convex combination to form the predictions

`prediction` A vector of length `T` that contains the predictions outputted by the aggregation rule.

`loss` The average loss (as stated by parameter `loss.type`) suffered by the aggregation rule.

`parameters` The learning parameters chosen by the aggregation rule.

`training` A list that contains useful temporary information of the aggregation rule to be updated and to perform predictions.

See Also

See and opera-vignette for a brief example about how to use the package.

`oracle` *Compute oracle predictions*

Description

The function `oracle` performs a strategie that cannot be defined online (in contrast to). It requires in advance the knowledge of the whole data set `Y` and the expert advice to be well defined. Examples of oracles are the best fixed expert, the best fixed convex combination rule, the best linear combination rule, or the best expert that can shift a few times.

Usage

```
oracle(Y, experts, model = "convex", loss.type = "square", awake = NULL,
  lambda = NULL, niter = NULL, ...)
```

Arguments

`Y`	A vector containing the observations to be predicted.
`experts`	A matrix containing the experts forecasts. Each column corresponds to the predictions proposed by an expert to predict `Y`. It has as many columns as there are experts.
`model`	A character string specifying the oracle to use or a list with a component `name` specifying the oracle and any additional parameter needed. Currently available oracles are: **'expert'** The best fixed (constant over time) expert oracle. **'convex'** The best fixed convex combination (vector of non-negative weights that sum to 1) **'linear'** The best fixed linear combination of expert **'shifting'** It computes for all number m of stwitches the sequence of experts with at most m shifts that would have performed the best to predict the sequence of observations in `Y`.
`loss.type`	A string or a list with a component 'name' specifying the loss function considered to evaluate the performance. It can be 'square', 'absolute', 'percentage', or 'pinball'. In the case of the pinball loss, the quantile can be provided by assigning to loss.type a list of two elements: **name** A string defining the name of the loss function (i.e., 'pinball') **tau** A number in `[0,1]` defining the quantile to be predicted. The default value is 0.5 to predict the median.
`awake`	A matrix specifying the activation coefficients of the experts. Its entries lie in `[0,1]`. Possible if some experts are specialists and do not always form and suggest prediction. If the expert number `k` at instance `t` does not form any prediction of observation `Y_t`, we can put `awake[t,k]=0` so that the mixture does not consider expert `k` in the mixture to predict `Y_t`.
`lambda`	A positive number used by the 'linear' oracle only. A possible L_2 regularization parameter for computing the linear oracle (if the design matrix is not identifiable)

`niter` A positive integer for 'convex' and 'linear' oracles if direct computation of the oracle is not implemented. It defines the number of optimization steps to perform in order to approximate the oracle (default value is 3).

`...` Additional parameters that are passed to function is order to perform convex optimization (see parameter `niter`).

Value

An object of class 'oracle' that contains:

`loss` The average loss suffered by the oracle. For the 'shifting' oracle, it is a vector of length `T` where `T` is the number of instance to be predicted (i.e., the length of the sequence `Y`). The value of $loss(m)$ is the loss (determined by the parameter `loss.type`) suffered by the best sequence of expert with at most $m-1$ shifts.

`coefficients` Not for the 'shifting' oracle. A vector containing the best weight vector corresponding to the oracle.

`prediction` Not for the 'shifting' oracle. A vector containing the predictions of the oracle.

`rmse` If loss.type is the square loss (default) only. The root mean square error (i.e., it is the square root of `loss`.

`predict.mixture` *Predict method for Mixture models*

Description

Performs sequential predictions and updates of a mixture object based on new observations and expert advice.

Usage

```
## S3 method for class 'mixture'
predict(object, newexperts = NULL, newY = NULL,
  awake = NULL, online = TRUE, type = c("model", "response", "weights",
  "all"), ...)
```

Arguments

`object` Object of class inheriting from 'mixture'

`newexperts` An optional matrix in which to look for expert advice with which predict. If omitted, the past predictions of the object are returned and the object is not updated.

`newY` An optional vector of observations to be predicted. If provided, it should have the same length as the number of rows of `newexperts`. If omitted, the object (i.e, the aggregation rule) is not updated.

`awake` An optional matrix specifying the activation coefficients of the experts. Its entries lie in `[0,1]`. Possible if some experts are specialists and do not always form and suggest prediction. If the expert number `k` at instance `t` does not form any prediction of observation `Y_t`, we can put `awake[t,k]=0` so that the mixture does not consider expert `k` in the mixture to predict `Y_t`.

`online` A boolean determining if the observations in newY are predicted sequentially (by updating the object step by step) or not. If FALSE, the observations are predicting using the object (without using any past information in newY). If TRUE, newY and newexperts should not be null.

`type` Type of prediction. It can be

model return the updated version of object (using newY and newexperts).

response return the forecasts. If type is 'model', forecasts can also be obtained from the last values of object$prediction.

weights return the weights assigned to the expert advice to produce the forecasts. If type is 'model', forecasts can also be obtained from the last rows of object$weights.

all return a list containing 'model', 'response', and 'weights'.

`...` further arguments are ignored

Value

`predict.mixture` produces a vector of predictions (type = 'response'), an updated object (type = 'model'), or a matrix of weights (type = 'weights').

References

[13] D. Adamskiy, W.M. Koolen, A. Chernov, and V. Vovk. A closer look at adaptive regret. In *Algorithmic Learning Theory (ALT)*, pages 290–304, 2012.

[14] M. Aiolfi, C. Capistrán, and A. Timmermann. Forecast combinations. Working Papers 2010-04, Banco de México, 2010. URL http://EconPapers.repec.org/RePEc:bdm:wpaper:2010-04.

[15] P. Algoet. The strong law of large numbers for sequential decisions under uncertainty. *IEEE Transactions on Information Theory*, 40(3):609–633, 1994.

[16] A. Antoniadis, E. Paparoditis, and T. Sapatinas. A functional wavelet–kernel approach for time series prediction. *Journal of the Royal Statistical Society: Series B*, 68(5):837–857, 2006.

[17] A. Antoniadis, X. Brossat, J. Cugliari, and J.M. Poggi. Prévision d'un processus à valeurs fonctionnelles en présence de non stationnarités. Application à la consommation d'électricité. *Journal de la Société Française de Statistique*, 153(2):52–78, 2012.

[18] A. Antoniadis, X. Brossat, J. Cugliari, and J.M. Poggi. Clustering functional data using wavelets. *International Journal of Wavelets, Multiresolution and Information Processing*, 11 (01), 2013.

[19] C-P. Astolfi, S. Da Veiga, and G. Stoltz. Forecasting production data of oil reservoirs with experts. Technical report, ENS Paris, 2012.

[20] P. Auer, N. Cesa-Bianchi, and C. Gentile. Adaptive and self-confident on-line learning algorithms. *Journal of Computer and System Sciences*, 64:48–75, 2002.

[21] K.S. Azoury and M.K. Warmuth. Relative loss bounds for on-line density estimation with the exponential family of distributions. *Machine Learning*, 43(3):211–246, 2001.

[22] A. Ba, M. Sinn, Y. Goude, and P. Pompey. Adaptive learning of smoothing functions: Application to electricity load forecasting. In P. Bartlett, F.C.N. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2519–2527. 2012. URL http://books.nips.cc/papers/files/nips25/NIPS2012_1205.pdf.

[23] A. Belloni and V. Chernozhukov. l1-penalized quantile regression in high-dimensional sparse models. *Ann. Statist.*, 39(1):82–130, 02 2011.

[24] G. Biau and B. Patra. Sequential quantile prediction of time series. *IEEE Transactions on Information Theory*, 57(3):1664–1674, 2011.

[25] G. Biau, K. Bleakley, L. Györfi, and G. Ottucsák. Nonparametric sequential prediction of time series. *Journal of Nonparametric Statistics*, 22(3):297–317, 2010.

[26] C. Bissuel, Y. Goude, and B. Péchiné. Heat load forecasting. In J. Taler, editor, *Modern Energy Technologies, Systems and Units*. Wydawnictwo Politechniki Krakowskiej, Kraków, 2013.

[27] D. Blackwell. An analog of the minimax theorem for vector payoffs. *Pacific Journal of Mathematics*, 6:1–8, 1956.

[28] A. Blum. Empirical support for winnow and weighted-majority algorithms: Results on a calendar scheduling domain. *Machine Learning*, 26:5–23, 1997.

[29] A. Blum and Y. Mansour. From external to internal regret. *Journal of Machine Learning Research*, 8:1307–1324, 2007.

[30] D. Bosq. *Nonparametric statistics for stochastic processes : estimation and prediction*. Lecture notes in statistics. Springer, New York, 1996.

[31] S. Boucheron, G. Lugosi, and P. Massart. *Concentration inequalities: a nonasymptotic theory of independence*. Oxford University Press, 2013.

[32] O. Bousquet and M.K. Warmuth. Tracking a small set of experts by mixing past posteriors. *Journal of Machine Learning Research*, 3:363–396, 2002.

[33] L. Breiman. The individual ergodic theorem of information theory. *Annals of Mathematical Statistics*, 31:809–811, 1957.

[34] L. Breiman. Bagging predictor. *Machine Learning*, 24(2):123–140, 1996.

[35] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, October 2001.

[36] L. Breiman, J. Friedman, C.J. Stone, and R.A. Olshen. *Classification and Regression Trees*. Wadsworth International Group, Belmont, CA, 1984.

[37] G.W. Brier. Verification of Forecasts expressed in terms of probability. *Monthly Weather Review*, 78(1):1–3, January 1950.

[38] P.J. Brockwell and R.A. Davis. *Time series : theory and methods*. Springer Series in Statistics. Springer, New York, 1991.

[39] A. Bruhns, G. Deurveilher, and J.-S. Roy. A non-linear regression model for mid-term load forecasting and improvements in seasonnality. In *Proceedings of the Fifteenth Power Systems Computation Conference (PSCC)*, 2005.

[40] O. Cappé, E. Moulines, and T. Rydén. *Inference in hidden Markov models*. Springer Series in Statistics. Springer, New York, 2005. ISBN 978-0387-40264-2; 0-387-40264-0.

[41] N. Cesa-Bianchi. Analysis of two gradient-based algorithms for on-line regression. *J. Comput. System Sci.*, 59(3):392–411, 1999.

[42] N. Cesa-Bianchi and G. Lugosi. Potential-based algorithms in on-line prediction and game theory. *Machine Learning*, 51(3):239–261, 2003.

[43] N. Cesa-Bianchi and G. Lugosi. *Prediction, learning, and games*. Cambridge University Press, 2006.

[44] N. Cesa-Bianchi, G. Lugosi, and G. Stoltz. Minimizing regret with label efficient prediction. *IEEE Trans. Inform. Theory*, 51:77–92, 2005.

[45] N. Cesa-Bianchi, Y. Mansour, and G. Stoltz. Improved second-order bounds for prediction with expert advice. *Machine Learning*, 66(2/3):321–352, 2007.

[46] K. Chaudhuri, Y. Freund, and Daniel J.H. A parameter-free hedging algorithm. In Y. Bengio, D. Schuurmans, J.D. Lafferty, C.K.I. Williams, and A. Culotta, editors, *Advances in Neural Information Processing Systems 22*, pages 297–305. Curran Associates, Inc., 2009. URL http://papers.nips.cc/paper/3883-a-parameter-free-hedging-algorithm.pdf.

[47] A. Chernov and F. Zhdanov. Prediction with expert advice under discounted loss. In *Proceedings of the 21st International Conference on Algorithmic Learning Theory, ALT 2010*, pages 255–269. Springer, 2008.

[48] A.V. Chernov and V. Vovk. Prediction with advice of unknown number of experts. *CoRR*, abs/1006.0475, 2010.

[49] C-K. Chiang, T. Yang, C-J. Lee, M. Mahdavi, C-J. Lu, R. Jin, and S. Zhu. Online optimization with gradual variations. In *Proceedings of the 25th Annual Conference on Learning Theory (COLT)*, pages 6.1–6.20, 2012.

[50] H. Cho, Y. Goude, X. Brossat, and Q. Yao. Modeling and forecasting daily electricity load curves: a hybrid approach. *Journal of the American Statistical Association*, 108:7–21, 2013.

[51] H. Cho, Y. Goude, X. Brossat, and Q. Yao. Modeling and forecasting daily electricity load using curve linear regression. In *Lecture Notes in Statistics: Modeling and Stochastic Learning for Forecasting in High Dimension*, 2014. To appear.

[52] Y.S. Chow. Local convergence of martingales and the law of large numbers. *Annals of Mathematical Statistics*, 36:552–558, 1965.

[53] R.T. Clemen. Combining forecasts: A review and annotated bibliography. *International Journal of Forecasting*, 5(4):559–583, 1989.

[54] X. Conort. 10 r packages to win kaggle competitions. In *Ruser*, 2014.

[55] J. Cugliari. *Prévision non paramétrique de processus à valeurs fonctionnelles : application à la consommation d'électricité*. PhD thesis, Université Paris-Sud 11, 2011.

[56] S. Dasgupta and D. Hsu. On-line estimation with the multivariate gaussian distribution. In *20th Annual Conference on Learning Theory, COLT 2007*, pages 278–292, 2007.

[57] A. Dawid and V. Vovk. Prequential probability: Principles and properties. *Bernoulli*, pages 125–162, 1999.

[58] S. de Rooij, T. van Erven, P.D. Grünwald, and W.M. Koolen. Follow the leader if you can, hedge if you must. *Journal of Machine Learning Research*, 15:1281–1316, 2014.

[59] R. Deswartes. Application des méthodes random forests à la prévision de consommation électrique. Technical report, EDF R&D, 2012.

[60] M. Devaine, P. Gaillard, Y. Goude, and G. Stoltz. Forecasting electricity consumption by aggregating specialized experts – a review of the sequential aggregation of specialized experts, with an application to Slovakian and French country-wide one-day-ahead (half-)hourly predictions. *Machine Learning*, 90(2):231–260, 2013.

[61] V. Dordonnat. *State-space modelling for high frequency data: Three applications to French national electricity load*. PhD thesis, Université d'Amsterdam, 2009.

[62] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra. Efficient projections onto the ℓ_1–ball for learning in high dimensions. In *Proceedings of the 25th International Conference on Machine Learning, ICML 2008*, 2008.

[63] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(Jul):2121–2159, 2011.

[64] R.M. Dudley. The sizes of compact subsets of hilbert space and continuity of gaussian processes. *Journal of Functional Analysis*, 1(3):290 – 330, 1967.

[65] E. Eban, A. Birnbaum, S. Shalev-Shwartz, and A. Globerson. Learning the experts for online sequence prediction. In *Proceedings of ICML*, 2012.

[66] S. Fan and R. J. Hyndman. Short-term load forecasting based on a semi-parametric additive model. *IEEE Transactions on Power Systems*, 27(1):134–141, 2012.

[67] D.P. Foster and R.V. Vohra. Asymptotic calibration. *Biometrika*, 85(2):379–390, 1998.

[68] D. Freedman. On tail probabilities for martingales. *Annals of Probability*, 3:100–118, 1975.

[69] Y. Freund. Predicting a binary sequence almost as well as the optimal biased coin. *Information and Computation*, 182(2):73 – 94, 2003.

[70] Y. Freund and R.E. Schapire. A decision-theoretic generalization of on-line learning and an applicationto boosting. *Journal of Computer and System Sciences*, 55:119–139, 1997.

[71] Y. Freund, R.E. Schapire, Y. Singer, and M.K. Warmuth. Using and combining predictors that specialize. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, pages 334–343, 1997.

[72] J.H. Friedman. Stochastic gradient boosting. *Computational Statistics and Data Analysis*, 38:367–378, 1999.

[73] J.H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2000.

[74] J.H. Friedman, T. Hastie, and R. Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2 2010.

[75] P. Gaillard and Y. Goude. Forecasting the electricity consumption by aggregating experts; how to design a good set of experts. In Anestis Antoniadis, Xavier Brossat, and Jean-Michel

Poggi, editors, *Modeling and Stochastic Learning for Forecasting in High Dimension*. Springer, 2014. To appear.

[76] P. Gaillard, Y. Goude, and G. Stoltz. A further look at the forecasting of the electricity consumption by aggregation of specialized experts. Technical report, 2011. URL pierre.gaillard.me/doc/GaGoSt-report.pdf.

[77] P. Gaillard, G. Stoltz, and T. van Erven. A second-order bound with excess losses. In *Proceedings of COLT*, volume 35, pages 176–196, 2014.

[78] S. Gerchinovitz. *Prediction of individual sequences and prediction in the statistical framework: some links around sparse regression and aggregation techniques*. PhD thesis, Université Paris-Sud 11, Orsay, 2011.

[79] S. Gerchinovitz and J.Y. Yu. Adaptive and optimal online linear regression on ℓ^1-balls. *Theoretical Computer Science*, 519:4–28, 2014.

[80] Y. Goude. *Melange de prédicteurs et application à la prévision de consommation électrique*. PhD thesis, Université Paris-Sud 11, 2008.

[81] Y. Goude, R. Nedellec, and N. Kong. Local short and middle term electricity load forecasting with semi-parametric additive models. *submitted to IEEE transactions on smart grid*, 2012.

[82] P. Grünwald. *The Minimum Description Length Principle*. MIT Press, 2007.

[83] L. Györfi and G. Ottucsák. Sequential prediction of unbounded stationary time series. *IEEE Transactions on Information Theory*, 53(5):1866–1872, 2007.

[84] L. Györfi, W. Härdle, P. Sarda, and P. Vieu. *Nonparametric curve estimation from time series*. Number 60 in Lecture notes in statistics. Springer-Verlag, Berlin, 1989.

[85] L. Györfi, G. Lugosi, and R.T. Fargas. Strategies for sequential prediction of stationary time series, 2001.

[86] T. J. Hastie, R. J. Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York Inc., New York, NY, USA, 2001.

[87] T.J. Hastie and R. Tibshirani. Generalized additive models (with discussion). *Statistical Science*, 1:297–318., 1986.

[88] T.J. Hastie and R. Tibshirani. *Generalized Additive Models*. Chapman & Hall/CRC, 1990.

[89] E. Hazan and S. Kale. Extracting certainty from uncertainty: regret bounded by variation in costs. *Machine Learning*, 80(2-3):165–188, 2010.

[90] E. Hazan and C. Seshadhri. Efficient learning algorithms for changing environments. *Proceedings of the 26th International Conference of Machine Learning (ICML)*, 2009.

[91] D.P. Helmbold and M.K. Warmuth. Learning permutations with exponential weights. *Journal of Machine Learning Research*, 10:1687–1718, 2009.

[92] M. Herbster and M. Warmuth. Tracking the best expert. *Machine Learning*, 32:151–178, 1998.

[93] M. Herbster and M. Warmuth. Tracking the best linear predictor. *Journal of Machine Learning Research*, 1:281–309, 2001.

[94] A. Hoerl and R. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12:55–67, 1970.

[95] T. Hong, P. Pinson, and S. Fan. Global energy forecasting competition 2012. *International Journal of Forecasting*, 30(2):357–363, 2014.

[96] V. Kanade, B. McMahan, and B. Bryan. Sleeping experts and bandits with stochastic action availability and adversarial rewards. In *Proceedings of the 12th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 272–279, 2009.

[97] J. Kivinen and M.K. Warmuth. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*, 132(1):1–63, 1997.

[98] R. Kleinberg, A. Niculescu-Mizil, and Y. Sharma. Regret bounds for sleeping experts and bandits. *Machine Learning*, 80(2-3):245–272, 2010.

[99] R.D. Kleinberg, A. Niculescu-Mizil, and Y. Sharma. Regret bounds for sleeping experts and bandits. In *Proceedings of the 21st Annual Conference on Learning Theory (COLT)*, pages 425–436, 2008.

[100] R. Koenker. *quantreg: Quantile Regression*, 2013. URL http://CRAN.R-project.org/package=quantreg. R package version 5.05.

[101] R. W Koenker and G.W. Bassett. Regression quantiles. *Econometrica*, 46(1):33–50, 1978.

[102] A.N. Kolmogorov and V.M. Tikhomirov. ε-entropy and ε-capacity of sets in function spaces. *Translations of the American Mathematical Society*, 17:277–364, 1961.

[103] W. M. Koolen, D. Adamskiy, and M. K. Warmuth. Putting Bayes to sleep. In *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, pages 135–143, 2013.

[104] W. Kotlowski and P. Grünwald. Maximum likelihood vs. sequential normalized maximum likelihood in on-line density estimation. In *Proceedings of COLT*, volume 24, pages 761–779, 2011.

[105] T. Launay. *Bayesian methods for electricity load forecasting.* PhD thesis, Université de Nantes, Orsay, 2012.

[106] A. Liaw and M. Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002.

[107] N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–261, 1994.

[108] G.G. Lorentz. Metric entropy, widths, and superpositions of functions. *Amer. Math. Monthly*, 69(6):469–485, 1962.

[109] V. Mallet. Prévision d'ensemble. Lecture notes, 2008.

[110] V. Mallet. Ensemble forecast of analyses: Coupling data assimilation and sequential aggregation. *Journal of Geophysical Research*, 115(D24303), 2010.

[111] V. Mallet, G. Stoltz, and B. Mauricette. Ozone ensemble forecast with machine learning algorithms. *Journal of Geophysical Research*, 114(D05307), March 2009.

[112] S. Mannor and G. Stoltz. A geometric proof of calibration, 2010.

[113] P. Massart. *Concentration Inequalities and Model Selection*, volume 1896 of *Lecture Notes in Mathematics*. Springer, Berlin, 2007.

[114] N. Merhav and M. Feder. Universal prediction. *IEEE Transactions on Information Theory*, 44(6):2124–2147, 1998.

[115] C. Monteleoni, G. A. Schmidt, S. Saroha, and E. Asplund. Tracking climate models. *Statistical Analysis and Data Mining*, 4(4):372–392, 2011.

[116] G. Morvai and B. Weiss. Nonparametric sequential prediction for stationary processes. *Ann. Probab.*, 39(3):1137–1160, 2011.

[117] R. Nedellec, J. Cugliari, and Y. Goude. Gefcom2012: Electric load forecasting and backcasting with semi-parametric models. *International Journal of Forecasting*, 30(2):375 – 381, 2014.

[118] J. Nowotarski and R. Weron. Computing electricity spot price prediction intervals using quantile regression and forecast averaging. *Computational Statistics*, pages 1–13, 2014. ISSN 0943-4062.

[119] A. Paterek. *Predicting movie ratings and recommender systems - a monograph*. 2012.

[120] A. Pichavant. Construction modèle gam edf. Note interne EDF R&D, Oct. 2014.

[121] A. Pierrot and Y. Goude. Short-term electricity load forecasting with generalized additive models. In *Proceedings of ISAP power, pp 593-600*, 2011.

[122] P. Pompey, A. Bondu, Y. Goude, and M. Sinn. Massive-scale simulation of electrical load in smart grids using generalized additive models. In Anestis Antoniadis, Xavier Brossat, and Jean-Michel Poggi, editors, *Modeling and Stochastic Learning for Forecasting in High Dimension*. Springer, 2014. To appear.

[123] M. Raginsky, R.F. Marcia, J. Silva, and R.M. Willett. Sequential probability assignment via online convex programming using exponential families. In *IEEE International Symposium on Information Theory*, pages 1338–1342. IEEE, 2009.

[124] A. Rakhlin and K. Sridharan. Online nonparametric regression. *JMLR W&CP*, 35 (Proceedings of COLT 2014):1232–1264, 2014.

[125] A. Rakhlin, K. Sridharan, and A.B. Tsybakov. Empirical entropy, minimax regret and minimax risk. *Bernoulli*, 2013. URL http://arxiv.org/abs/1308.1147. To appear.

[126] S. Rakhlin, O. Shamir, and K. Sridharan. Relax and randomize : From value to algorithms. In F. Pereira, C.J.C. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 2141–2149. Curran Associates, Inc., 2012. URL http://papers.nips.cc/paper/4638-relax-and-randomize-from-value-to-algorithms.pdf.

[127] G. Ridgeway. Generalized boosted models: A guide to the gbm package, 2005.

[128] J. Rissanen. Universal coding, information, prediction, and estimation. *Information Theory, IEEE Transactions on*, 30(4):629–636, 1984.

[129] V. Rivoirard and G. Stoltz. *Statistique mathématique en action*. Vuibert, 2012.

[130] W. Rudin. *Functional Analysis*. McGraw-Hill Science, 1991.

[131] G. Shafer and V. Vovk. *Probability and finance: it's only a game!*, volume 491. John Wiley & Sons, 2005.

[132] Y.M. Shtarkov. Universal sequential coding of single messages. *Problems of Information Transmission*, 23(3):3–17, 1987.

[133] G. Stoltz and G. Lugosi. Learning correlated equilibria in games with compact sets of strategies. *Games and Economic Behavior*, 59:187–208, 2007.

[134] M. Talagrand. *The generic chaining*. Springer, 2005.

[135] V. Thouvenot, A. Pichavant, Y. Goude, A. Antoniadis, and J.M. Poggi. Electricity forecasting using multi-step estimators of nonlinear additive models. 2015.

[136] R. Tibshirani. Regression shrinkage and selection via the lasso. *J. Roy. Statist. Soc. Ser. B*, 58(1):267–288, 1996.

[137] A.B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer, 2009.

[138] T. van Erven, P. Grünwald, W.M. Koolen, and S. de Rooij. Adaptive Hedge. In *Advances in Neural Information Processing Systems 25 (NIPS 2012)*, 2011.

[139] V. Vovk. Aggregating strategies. In *Proceedings of the Third Workshop on Computational Learning Theory*, pages 371–386, 1990.

[140] V. Vovk. A game of prediction with expert advice. *Journal of Computer and System Sciences*, 56(2):153–173, 1998.

[141] V. Vovk. Derandomizing stochastic prediction strategies. *Machine Learning*, 35(3):247–282, Jun. 1999.

[142] V. Vovk. Competitive on-line statistics. *International Statistical Review*, 69(2):213–248, 2001.

[143] V. Vovk. On-line regression competitive with reproducing kernel hilbert spaces. *arXiv*, 2005. URL http://arxiv.org/abs/cs.LG/0511058.

[144] V. Vovk. Metric entropy in competitive on-line prediction. *arXiv*, 2006. URL http://arxiv.org/abs/cs.LG/0609045.

[145] R. Weron and A. Misiorek. Forecasting spot electricity prices: A comparison of parametric and semiparametric time series models. *International Journal of Forecasting*, 24(4):744 – 763, 2008.

[146] O. Wintenberger. Optimal learning with bernstein online aggregation. Extended version available at arXiv:1404.1356 [stat.ML], 2014.

[147] S.N. Wood. *Generalized Additive Models: An Introduction with R*. Chapman and Hall/CRC, 2006.

[148] S.N. Wood, Y. Goude, and S. Shaw. Generalized additive models for large datasets. *Journal of Royal Statistical Society, Series C*, 2014. to appear.

[149] S. Yasutake, K. Hatano, S. Kijima, E. Takimoto, and M. Takeda. Online linear optimization over permutations. *Algorithms and Computation*, 7074:534–543, 2011.

[150] Z-H. Zhou. *Ensemble Methods: Foundations and Algorithms*. Chapman & Hall/CRC, 1st edition, 2012. ISBN 1439830037, 9781439830031.

[151] M. Zinkevich. Online convex programming and generalized infinitesimal gradient ascent. In *Proceedings of the 20th International Conference on Machine Learning, ICML 2003*, 2003.