Delft University of Technology
Faculty of EEMCS

Online Nonparametric Regression for Non-Stationary Time Series with Dependent Noise

by
Ivan Krylov

to obtain the degree of Master of Science in Applied Mathematics
at the Delft University of Technology,
to be defended publicly on 11 July 2025.

Student number:	5046017		
Project duration:	November 1, 2024 – July 11, 2025		
Thesis committee:	Dr. F. Mies,	TU Delft,	Daily Supervisor
	Prof. dr. ir. Jongbloed,	TU Delft,	Responsible Supervisor
	Dr. ir. G.F. Nane,	TU Delft	

TU Delft

Abstract

We consider the problem of online nonparametric regression for signals of length n with total variation at most C_n whose observations are contaminated by σ-subgaussian noise. While there exist many algorithms which achieve optimal performance under the assumption of independent noise, this work focusses on the less explored general case of dependent noise. We focus on the Follow-the-Leading-History (FLH) [1] algorithm, a powerful meta-aggregation method for online learning.

We prove that under mild assumptions of weak long-range dependence, we may apply FLH to $m \approx \log n$ partitioned data streams to mitigate high correlations. We show that the resulting algorithm Thinned-FLH (TFLH) achieves the minimax optimal cumulative error rate of $\tilde{\mathcal{O}}(n^{1/3}C_n^{2/3})$ with high probability, matching the performance in the independent case up to logarithmic factors. We also conduct a simulation study, which validates our theoretical findings and demonstrates that TFLH may outperform FLH in high dependence environments in spite of the data thinning.

Acknowledgments

The greatest thanks go to Professor Fabian Mies without whom this thesis would not exist. He showed me, more than any other teacher, what it is truly like to do mathematics research and best of all he did in such a way that it felt equal parts approachable, stimulating, and fun. Our weekly meetings were an anchor in my otherwise schedule-less year and I will miss them very much. I am very grateful to have had him as a supervisor.

Next, I would like to thank Professor Geurt Jongbloed and Professor Tina Nane for their flexibility with the defence data and being able to fill out my exam committee.

I would also like to take this moment to thank my friends who have made my student life so memorable. In no particular order, my teammates, the guys from Rotterdam, the friends on the 4th floor, the Dutch and the Italians, my roommates, and all the other people I have met in my 6 years of studying. I also give my thanks to Thomas for his company this year and help with the manuscript.

A special mention goes to my partner Charelle who was with me through all the ups and downs and supported me though my entire graduate studies. Lastly, I want to thank my family - my mother, my father, my brother, my grandfather - for their support and love.

Experience with real-world data soon convinces one that both stationarity and Gaussianity are fairy tales invented for the amusement of undergraduates.

- David J. Thomson

Contents

Chapter 1

Introduction

A common objective in statistics is to determine the relationship between some outcomes and predictors. This can be done in a number of ways, but perhaps the most popular is to assume that there exists some underlying regression function that best describes the relationship between these two quantities and to in turn estimate such a function.

In practice, we often only have access to noisy observations and so the standard setup is to assume the data is generated according to the model

$$y_i = f(x_i) + \epsilon_i \tag{1.1}$$

for $i = 1, \dots, n$, where $f : X \to \mathbb{R}$ is the regression function to be estimated, $x_1, \dots, x_n \in X \subseteq \mathbb{R}^d$ are the predictors, and ϵ_i are the random errors which are typically assumed to be i.i.d. with zero mean.

In this work, we consider online nonparametric regression. In traditional nonparametric regression, we have access to all the data and estimate our regression function 'in hindsight', however in the online setting, data arrives in a continuous stream, one sample at a time. Specifically, we will consider $t = 1, \dots, n$ rounds during which we, in succession, make a prediction $\widehat{\mu}_t$ from some convex domain $K \subset \mathbb{R}$, receive a noisy output generated according to the univariate model

$$y_i = \theta_i + Z_i, \tag{1.2}$$

and suffer a loss based on our prediction. Here $\theta_{1:n} := (\theta_1, \dots, \theta_n)$ is the true signal to be estimated and $Z_{1:n}$ is the noise signal of σ-subgaussian error entries, see Definition 2.1 for a definition of σ-subgaussian random variables. We see that the formulations of (1.1) and (1.2) are equivalent if we imagine that $\theta_{1:n}$ corresponds to the evaluations of the regression function f on a sorted set of predictors. In a similar vein, our vector of predictions $\widehat{\mu}_{1:n}$ describes our estimated regression function.

To evaluate our predictions, we aim to minimise the squared cumulative error (cumulative error) defined by

$$\mathcal{R}_n(\widehat{\mu}_{1:n}, \theta_{1:n}) := \sum_{t=1}^{n} \mathbb{E}\left((\widehat{\mu}_t - \theta_t)^2\right). \tag{1.3}$$

In this way, we are in fact in the business of mean estimation. Without any additional constraints, this error will grow linearly in n. For this reason, we assume that the signal is in some sense regular and is bounded in total variation. For a real-valued vector $\theta_{1:n} \in \mathbb{R}^n$, we define the total variation by

$$TV(\theta_{1:n}) := \sum_{t=2}^{n} \|\theta_t - \theta_{t-1}\|_1.$$

We also assume that there exists some $B > 0$ such that $\|\theta_t\|_1 \leq B$ for all $1 \leq t \leq n$.

This work extends the results obtained by [2, 3] for independent errors to dependent errors. The central contribution of this thesis is demonstrating that under mild conditions on the decay of dependence, the Follow-the-Leading-History (FLH) [1] algorithm, when coupled with a data thinning scheme, preserves the minimax cumulative error rate of $\tilde{\mathcal{O}}(n^{\frac{1}{3}} C_n^{\frac{2}{3}})$, where $\tilde{\mathcal{O}}$ hides a logarithmic factor in n.

In Chapter 1 we give a detailed description of the problem and review the relevant existing literature, first independently for both nonparametric regression and online learning, and then their intersection. In Chapter 2 we show that the FLH algorithm attains optimal cumulative error bounds in the independent case. In Chapter 3 we extend these results to the dependent case and show that we still obtain optimal bounds minus an extra $\log n^{\frac{2}{3}}$ factor and we conduct a simulation study to empirically review our results in Chapter 4. Finally, we discuss our results and possible future avenues of research in Chapter 5.

1.1 Nonparametric Regression

Consider the (fixed or random) d-vector of covariates/predictors $X \in \mathcal{X} \subseteq \mathbb{R}^d$ and the outcome/response $Y \in \mathbb{R}$ which are distributed according to some unknown distribution $\mathbb{P}$ over $\mathcal{X} \times \mathbb{R}$. It can be shown that the conditional mean $f(X) := \mathbb{E}_{\mathbb{P}}(Y|X)$ is the optimal function for predicting Y from X with respect to the mean squared error [4]. However, because the underlying distribution is unknown to us, we instead estimate this function using the empirical distribution determined by n observations $(x_1, y_1), \ldots, (x_n, y_n)$ of our (random) variables.

Furthermore, because the L^2 space is too big to search [4], it is standard practice to also assume that f belongs to some function class $f \in \mathcal{F} \subsetneq L^2$ which imposes weak regularity conditions on the smoothness and/or regularity of f. If $\mathcal{F}$ may be indexed by some finite dimensional parameter set $\Theta \subset \mathbb{R}^p$, then we say that $\mathcal{F}$ is a parametric class, consider for example the class of all linear functions of X given by $\mathcal{F} = \{X^\top \beta : \beta \in \mathbb{R}^d\}$. Such parametric classes can be very powerful if correctly specified, however they may also fail to accurately capture the complexities of the underlying relationship and be too restrictive for practical applications. As a result, more flexible nonparametric classes which do not assume a priori any relationship between the response and covariate variables have grown in popularity.

Nonparametric regression has been extensively studied and many techniques have been developed to tackle this problem, perhaps the most classical of which is the class of linear smoothers. Here, smoothing refers to the attempt to remove higher frequency terms, in this case the error, whilst retaining the low frequencies, in this case the signal. Linear smoothers can be defined as estimators which are linear functions of the data and produce fitted values $\widehat{\theta}_{1:n}$ of the form $\widehat{\theta}_{1:n} = S^{(\lambda)} y_{1:n}$ for some smoothing matrix $S^{(\lambda)} \in \mathbb{R}^{n \times n}$ depending on the covariates and a tuning parameter λ. Note that if a data-driven technique such as cross-validation is used to select the tuning parameter, then we lose this linear relationship. This is because under such a regime, the tuning parameter becomes a function of the response variable. As a result, the smoothing matrix, which depends on the tuning parameter, is no longer independent of the response variable and we lose the linear property.

The advantages of linear smoothers are their mathematical and computational simplicity, however it is exactly this simplicity which makes them fundamentally limited for estimating functions which have variations in local smoothness. More precisely, suppose that the underlying regression function f^* lies in the function class

$$\mathcal{F}_k(C) := \{f : TV(f^{(k)}) \leq C\}$$

for some constant $C > 0$, where $f^{(k)}$ is the k^{th} weak derivative of f. Then [5] showed that the minimax cumulative error rate for estimation of f^* over $\mathcal{F}_k(C)$ is $\Omega(n^{\frac{1}{2k+3}})$ and in the same paper they also showed that linear smoothers have a minimax cumulative error rate of $\Omega(n^{\frac{1}{2k+2}})$. This difference is highly non trivial for our setting of $k = 0$, and shows that we can do much better. When we speak of minimax error rates, we refer to the best possible performance guarantee of an estimator in the worst case scenario.

To improve on linear smothers, we need estimators which recognise and adapt to variations in the smoothness of the signal from region to region; such estimators are said to be locally adaptive [6] (or spatially adaptive in the terminology of [5]). [5] developed a method which works in the wavelet domain by nonlinear shrinkage of the empirical wavelet coefficients. In their paper, they showed that this wavelet shrinkage can be tuned to near minimax rates over a wide range of smoothness constraints. Around the same time, [6] proposed to use the l_1-norm, defined by $\|\theta_{1:n}\|_1 := \sum_{i=1}^n |\theta_i|$, for least squares penalised estimation with a total variation penalty.

They called the resulting estimators locally adaptive regression splines and showed that they were minimax optimal over the class of functions of bounded total variation [6].

A relatively new 1-dimensional locally adaptive method is trend filtering. Given a fixed integer $r \geq 1$ and a tuning parameter $\lambda > 0$, the r^{th} order trend filtering estimator for $\theta_{1:n}$ in the constrained form is given by

$$\widehat{\theta}_{\lambda}^{(r)} := \underset{\theta^* \in \mathbb{R}^n}{\arg\min} \left\{ \frac{1}{2} \|y_{1:n} - \theta^*\|^2 : \left\| D^{(r)} \theta^* \right\|_1 \leq \lambda n^{1-r} \right\}$$

where $D^{(0)}\theta := \theta$, $D^{(1)}\theta := (\theta_2 - \theta_1, \ldots, \theta_n - \theta_{n-1})$, and $D^{(r)}\theta$ is recursively defined for $r \geq 2$ as $D^{(r)}\theta := D^{(1)}(D^{(r-1)}\theta)$. Then, whereas locally adaptive regression splines seek to perform penalized least square by penalizing the l_1 norm of a given order derivative of the fitted function, trend filtering instead penalises the absolute k^{th} order discrete derivatives. In this way trend filtering may be viewed as a discrete analog of locally adaptive regression splines.

Originally proposed by [7] under the name of higher-order TV regularization, the method was later independently rediscovered by [8], who coined the term trend filtering. [9] would later study the statistical properties of the trend filtering method and show that it achieves the optimal cumulative error bound of $\mathcal{O}(n^{\frac{1}{2k+3}})$ for an optimal choice of tuning parameter. Trend filtering proved so popular that a large body of literature has been published on its applications to adjacent fields such as functional data analysis, [10], additive models [11], and graphs [12].

Interestingly, the 0th order trend filtering estimate reduces to 1-dimensional total variation denoising (TVD) [13], also called the 1-dimension fused LASSO estimator [14]. [15] showed that total variation denoising was also minimax optimal on a 2D grid when bounded in discrete total variation, see also [16]. [17] identified a new local minmax/maxmin formula which provides a pointwise definition of the univariate TVD estimator as a minmax/maxmin of penalized local averages over intervals of varying scales. This local perspective was then generalized to define a new class of estimators called minmax trend filtering, which use local polynomial regressions instead of simple averages [17].

There has also been much work done on nonparametric regression for data with dependent errors. [18] showed that, under the assumption of second-order stationarity of errors, the convergence rates for regression mean estimates under the assumption of independence still hold if and only $\sum_j \gamma(j) < \infty$ where $\gamma(h) = \text{Cov}(\epsilon_i, \epsilon_{t+h})$ is the autocovariance function. In other words, convergence rates established under independence continue to hold asymptotically if and only if the long-range dependence is sufficiently weak. [19] would later show that the minimax cumulative error of regression function estimation for error terms with an arbitrary dependence structure whose second moments are uniformly bounded is no worse than that of i.i.d. Gaussian errors.

Much of the literature in this area involves adapting existing techniques to handle the dependence of errors. [20] proposed a two-step procedure which works by first nonparametrically estimating the error covariance matrix and then using this information for a modified regression fit which takes into account the estimated dependence structure. In a similar vein, [21] estimate the covariance matrix of a locally stationary process with a smoothly varying trend by way of local linear smoothing and then use this estimator to arrive at consistent predictors for more general non-stationary time series.

Parallel to these developments, [22] studied Empirical Risk Minimization (ERM) under both dependence of errors and the presence of heavy-tailed data. [22] extended the 'learning without concentration' framework (see [23]) to handle data from strictly stationary and exponentially β-mixing processes, providing probabilistic risk bounds for ERM with convex loss functions.

It is important to note, however, that many of these results rely on the crucial assumption that the errors, while potentially dependent on each other, are independent of the covariates. [24] demonstrated that when this independence is violated, the convergence rate of the least squares estimator can degrade significantly, and no universal moment condition on the errors alone is sufficient to guarantee rate-optimality.

1.2 Online Learning

The locally adaptive methods discussed previously provide powerful tools for estimating a function from a fixed dataset. However, these are inherently batch algorithms, designed to be run only once all data has been collected. In our setting, data arrives sequentially in a stream, and an estimate must be produced after each new observation is received. This sequential access to data is the key characteristic of online learning. Understanding the principles and performance measures of online learning is therefore essential to our work.

A generalisation of the model introduced in [25], online learning for univariate time series in its original form is formulated as follows. We consider a learner which has access to $i = 1 \dots, d$ experts $\mathcal{E}_i$. Across a series of trials $t = 1, \dots, n$, the learner successively receives $x_t \in \mathbb{R}^d$, the vector of predictions of all experts, makes a prediction $\widehat{\mu}_t$, which is some convex combination of expert predictions, and receives the true output y_t. Furthermore, the learner and each expert incur a loss at the end of each trial. It is clear to see that the cumulative loss of the learner is a poor performance measure, because any convex combination of experts which minimises the loss may still suffer a 'big' loss since we make no assumptions on the relationship between x_t and y_t, and so there always exists some sequence $y_1, \dots, y_n$ which is 'far away' from any possible choices for $\widehat{\mu}_1, \dots, \widehat{\mu}_n$. A better performance measure is to minimise the difference between the cumulative loss of the learner and the loss of the best expert in hindsight.

Bounds on this difference are referred to as static bounds on the loss because the best expert in hindsight, the comparator, is time-invariant. Initially, the literature focused mostly on such static bounds, see [26, 27, 28]. However, this definition is limited because it cannot capture a changing environment, similarly to linear smoothers in the nonparametric regression framework. Indeed, in non-stationary environments where the optimal decision evolves, a static comparator is inadequate, as it fails to capture the potential for different experts to be locally optimal over different time intervals. [29] partly explored this issue, building on the results from [28], by considering the difference between the total regret for the best expert in hindsight, which is allowed to change k times, and a static best predictor.

The work of [29] shows that it can be more appropriate to allow the comparator to evolve with time and instead consider a sequence of comparators $u_1, \dots, u_n$ - bounds with respect to these sequences are called shifting bounds. Unlike the static bound, which is only dependent on n, the shifting bound is also dependent on the choice of comparator sequence. For this reason, it is usually assumed that the comparator sequence is bounded in total variation (notice the parallels to nonparametric regression). [30] explored shifting bounds for linear regression using projected mirror descent and introduced methods to lift known static bounds to shifting bounds. [31] built upon the work of [26, 29] and introduced a generalisation of the fixed-share algorithm, an algorithm with a novel weight update method, for these shifting bounds under the constraint that the pool of possible choices of comparators be much smaller than the total number of experts.

In their foundational paper on online convex optimisation, [32] reformulate online learning as a convex optimisation problem. At each round $t = 1, \dots, n$ the learner in succession makes a prediction $\widehat{\mu}_t$ from some convex domain $K^d \subset \mathbb{R}$ and suffers some loss measured by a loss function $f : K^d \to \mathbb{R}$. Note the (near) equivalence of the two formulations of online learning when K is understood as the set of all convex combinations of expert predictions. In this instance however, the two formulations are subtly different because we consider only a single unknown cost function whereas we previously considered n data points $y_1, \dots, y_n$. To remedy this, consider instead a sequence of, not necessarily distinct, unknown cost functions $f_1, \dots, f_n : K \to \mathbb{R}$. The general case where these cost functions are distinct is also called the non-stationary online convex optimisation problem.

Similarly to the framework of [30], it is clear to see that we cannot hope to choose some $\widehat{\mu}_t$ to minimise $f_t(\widehat{\mu}_t)$, because f_t is arbitrary. Instead, [32] defines the now standard performance measure of regret. Regret can be intuitively understood as the difference between the cumulative loss incurred by an online learner and the best decision in hindsight, whether that be a single expert or a convex combination of experts. Formally,

we can define the regret incurred by some algorithm $\mathcal{A}$ as

$$\text{Regret}_n(\mathcal{A}) := \sum_{t=1}^{n} f(\widehat{\mu}_t) - \min_{y^* \in K} \sum_{t=1}^{n} f_t(y^*). \tag{1.4}$$

This is often referred to as the static regret because the best decision in hindsight, the comparator, is time-invariant and this definition is equivalent to the aforementioned static bound. In their paper, [32] showed that online gradient descent (OGD) achieves a static regret of $\mathcal{O}(\sqrt{n})$ for convex loss functions and [1] later showed this bound can be improved to $\mathcal{O}(\log n)$ for the class of strongly convex functions. These bounds were shown to be minimax optimal by [33]. Furthermore, [1] also showed that for α-exp-concave loss functions, the online Newton step algorithm achieves a static regret of $\mathcal{O}\left(\frac{d}{\alpha} \log n\right)$, where d is the dimensionality of K.

To address the limitations of static regret, [32] compare the learner against a sequence of local minimisers. This is defined by

$$\text{Dynamic-Regret}_n(\mathcal{A}) := \sum_{t=1}^{n} f(\widehat{\mu}_t) - \sum_{t=1}^{n} f_t(y_t^*), \tag{1.5}$$

where $y_t^* := \arg\min_{y \in K} f_t(y)$, and is called the (worst-case) dynamic regret. A generalisation of (1.5) which is more analogous to shifting bounds is to use some arbitrary $u_1, \ldots, u_n \in K$ comparator sequence in place of the local minimisers, and we denote this general dynamic regret by $\text{Dynamic-Regret}_n(\mathcal{A}, u_1, \ldots, u_n)$.

It has been shown that it is not possible to achieve sub-linear worst-case dynamic regret unless we make further assumptions on the comparator or the function sequence [34, 35]. In their work, [32] define the path length, the variation of the comparator sequence, by

$$P_n := \sum_{t=2}^{n} |u_{t-1} - u_t|.$$

If this value is known in advance, then OGD achieves a dynamic regret of $\mathcal{O}(\sqrt{n}(1 + P_n^*))$ for convex functions [32, 36], where P_n^* denotes the path length of the local minimiser sequence. Later [36] showed that, under the further assumption that all y_t^* are in the interior of K, this bound can be improved to $\mathcal{O}(P_n^*)$ for convex and smooth functions. For strongly convex functions and smooth functions, [37] showed that the dynamic regret is at most $\mathcal{O}(P_n^*)$.

If we assume a bound on the functional variation, defined by

$$V_n := \sum_{t=2}^{n} \max_{y \in K} |f_t(y) - f_{t-1}(y)|,$$

and we know this value in advance, then [34] showed that restarted OGD achieves $\mathcal{O}(n^{\frac{2}{3}} V_n^{\frac{1}{3}})$ worst-case dynamic regret for convex functions. This bound was later improved to $\mathcal{O}(n^{\frac{1}{3}} V_n^{\frac{1}{3}})$ for 1-dimensional square loss using trend filtering techniques [2].

Note that though $\text{Dynamic-Regret}_n(\mathcal{A}, u_1, \ldots, u_n) \leq \text{Dynamic-Regret}_n(\mathcal{A})$ it does not mean that the worst-case dynamic regret is necessarily a better performance metric, because a bound for the worst-case may be very loose for well-behaved comparator sequences. If P_n is known ahead of time, [32] showed that one may choose an optimal step size and obtain an $\mathcal{O}(\sqrt{n(1 + P_n)}$ universal dynamic regret for convex functions. Later, [38] proposed an online algorithm to search for the optimal step size and obtained the same bound of $\mathcal{O}(\sqrt{n(1 + P_n)})$ as with oracle step size tuning, and they furthermore showed that this bound was minimax optimal.

An alternative performance measure proposed by [1] is the adaptive regret which measures the maximum static regret obtained by a learner over any continuous time interval. Formally

$$\text{Adaptive-Regret}_n(\mathcal{A}) := \sup_{I=[r,s] \subseteq [n]} \left(\sum_{t=r}^{s} f(\widehat{\mu}_t) - \min_{y^* \in K} \sum_{t=r}^{s} f(y^*) \right). \tag{1.6}$$

The advantage of adaptive regret over dynamic regret is that it better captures the dynamics of local optimality. Notice also that since the optimal y^* for different intervals can be different, the learner is essentially competing with a changing comparator as in dynamic regret. [1] developed the Follow-the-Leading-History (FLH) algorithm and showed that it obtained $\mathcal{O}(\frac{d}{\alpha}\log n)$ adaptive regret for α-exp-concave functions and $\mathcal{O}(\sqrt{n\log n})$ for convex and bounded functions. They also showed that a more computationally efficient version of their algorithm, Advanced Follow-the-Leading-History (AFLH), obtained $\mathcal{O}(\frac{d}{\alpha}\log^2 n)$ and $\mathcal{O}(\sqrt{n}\log^3 n)$ adaptive regrets respectively [1]. [39] would later investigative the relationship between adaptive and dynamic regrets and show that general dynamic regret can be upper bounded by adaptive regret and the functional variation.

The previous results in the online learning literature have all been under the assumption that the learner has uninhibited access to the true data and/or cost function. [34] analysed settings with noisy feedback of both the function value and its gradient. They established minimax dynamic regret bounds for OGD with noisy gradient feedback, $\mathcal{O}(V_n^{1/3}n^{2/3})$ for convex functions and $\mathcal{O}(\sqrt{V_n n})$ for strongly convex functions, demonstrating that deterministic results may be lifted to random environments.

1.3 Online Nonparametric Regression

The problem considered in this work lies at the intersection of online learning and nonparametric regression, often called online nonparametric regression. Recall that we do not have access to all our data as in the batch nonparametric regression framework. Instead, as in the online learning setting, data arrives sequentially, and an estimate must be made at each step. This sequential protocol presents a significant challenge for many classical nonparametric methods, such as trend filtering, as refitting such models from scratch after each new observation is computationally infeasible, especially for large datasets. Therefore, solutions to online nonparametric regression problems are focused on developing computationally efficient sequential algorithms which may be iteratively updated while also still retaining the flexibility and adaptivity of their batch counterparts.

When the function class $\mathcal{F}$ is infinite-dimensional, as is the case for Hölder or Sobolev spaces, standard online learning algorithms often struggle because they are designed to work with a finite set of experts. A natural first solution might be to discretise the function space by taking an ϵ-net and then applying a standard algorithm to this finite number of experts, such as the Exponentially Weighted Average (EWA) forecaster [40]. This method however, leads to suboptimal regret bounds due to approximation and complexity errors [41].

A breakthrough towards achieving optimal rates came by incorporating the ideas of the chaining technique [42]. Rather than using a single discretization, chaining involves creating a sequence of refining approximations, from coarse to fine, and bounding the error by summing the small incremental errors between each level of the chain. In their non-constructive work, [41, 43] established the minimax rates for online nonparametric regression, demonstrating that the minimax rates are determined by a sequential analogue of Dudley's entropy integral [41, 42]. Their work also crucially demonstrated that this rate matches the statistical i.i.d. learning rate whenever the corresponding sequential and empirical entropies coincide [41]. Building on these theoretical foundations, [42] designed the first explicit, constructive algorithm based on the chaining technique that achieves these optimal rates for online regression over Hölder balls. For a Hölder class with regularity $\beta > \frac{1}{2}$, their algorithm achieves a regret of $\mathcal{O}(n^{\frac{1}{2\beta+1}})$, matching the optimal rates derived from statistical learning theory with i.i.d. data [42]. Notice that for a sequence with bounded total variation, which is the special case of a Hölder class with $\beta = 1$, we arrive at our optimal $\mathcal{O}(n^{\frac{1}{3}})$. While this was a significant step forward, the proposed algorithm was computationally intensive, requiring the maintenance of exponentially many weights [42].

More recent work has focused on developing computationally efficient algorithms for the class of functions of bounded total variation. Early progress was made by [2] as they developed the Adaptive Restarting Rule for Online averaging using Wavelet Shrinkage (ARROWS) policy, an algorithm which works by using moving averages with an adaptive restart schedule calculated using wavelet shrinkage techniques, see Chapter 4 for more detail on the algorithm. This was the first polynomial-time algorithm to achieve the optimal cumulative error bound of $\tilde{\mathcal{O}}(n^{\frac{1}{3}}C_n^{\frac{2}{3}})$ in this setting, however the method was restricted to the 1-dimensional setting and required advance knowledge of the noise σ level. [3] extended the results of [2] to general d-dimensional non-

stationary nonparametric regression by way of the FLH algorithm, which does not require advanced knowledge of σ, and showed optimality in this more general setting. The policy ARROWS itself was later generalized in [44] to forecast sequences with piecewise polynomial structure. The proposed polynomial time policy Adaptive Vovk Azoury Warmuth (Ada-VAW) forecaster also used wavelets to detect change points and was shown to be adaptively minimax optimal for sequences which have k^{th} order bounded total variation.

More recently, [45] proposed a computationally efficient algorithm based on chaining trees that achieves locally adaptive minimax regret, dynamically adjusting its partitioning of the input space to align with local smoothness variations without prior knowledge of the function's regularity [45].

A common thread throughout the existing literature on online nonparametric regression, from the foundational theoretical work to the latest efficient and adaptive algorithms, is the underlying assumption that the noise terms are independent. The performance guarantees for these methods rely on this assumption, leaving a gap in the literature as to how these methods behave when the errors exhibit temporal dependence. This thesis confronts this critical gap. We rigorously establish that, under weak assumptions on the error dependence structure, optimal performance guarantees are still attainable for a computationally efficient online algorithm.

Chapter 2

Preliminaries

This chapter lays the theoretical groundwork for our main contributions in Chapter 3 by investigating the case of (1.2) with independent errors. Establishing a performance bound in this setting is a crucial first step. We begin by introducing an oracle estimator an idealised benchmark with access to information unavailable in practical settings. We then present the FLH algorithm and leverage the oracle's performance to derive the minimax upper bound on the cumulative error of the FLH estimates.

The performance guarantees derived in this work are all with respect to the cumulative squared error

$$\mathcal{R}_n(\widehat{\mu}_{1:n}, \theta_{1:n}) := \sum_{t=1}^{n} \mathbb{E}\left((\widehat{\mu}_t - \theta_t)^2\right).$$

This choice of metric is deliberate. While regret measures relative performance against a comparator, our goal aligns with the classical statistical objective of estimation accuracy. The cumulative error provides a direct measure of the estimator's accuracy that is, how close our estimates $\widehat{\mu}_{1:n}$ are to the true underlying signal $\theta_{1:n}$. This contrasts with metrics common in online learning, such as dynamic or adaptive regret, which quantify relative performance.

Regret measures how well an algorithm adapts to a data stream compared to a powerful, often clairvoyant, benchmark. Consequently, in a scenario with a highly erratic signal, an algorithm might achieve low regret simply because no comparator could have performed well. The metric effectively 'grades on a curve' by accounting for the problem's intrinsic difficulty. However, our objective is different. We are not competing against a benchmark; we are trying to find the relationship between the predictors and covariates. The cumulative error precisely captures this notion of absolute accuracy, penalizing estimators that are far from the true signal, irrespective of that signal's complexity.

2.1 Oracle Forecaster

Let $m \geq 1$ and consider the hypothetical forecaster which produces moving averages with at most m restarts. This forecaster is predicated on a sequence of restart times $1 = t_1 \leq t_2 \leq \cdots \leq t_{m+1} = n + 1$ defined such that

$$TV(\theta_{t_i:(t_{i+1}-1)}) \leq \frac{TV(\theta_{1:n})}{m} \tag{2.1}$$

for all $1 \leq i \leq m$. These restart times are used to inform our oracle forecaster so that for all $t \in \{t_i+1, \ldots, t_{i+1}\}$, we define the next forecast by

$$\widetilde{\mu}_t := \bar{y}_{t_i:(t-1)}, \text{ where } \bar{y}_{t_i:(t-1)} := \frac{1}{t - t_i} \sum_{k=t_i}^{t-1} y_k. \tag{2.2}$$

Then the oracle forecasts are the moving average predictions with as restart times $t_1, \ldots, t_{m+1}$.

In Theorem 2.2 we show that this hypothetical forecaster achieves an optimal cumulative error of

$$\mathcal{R}_n(\widetilde{\mu}_{1:n}, \theta_{1:n}) \leq \tilde{\mathcal{O}}(n^{\frac{1}{3}} C_n^{\frac{2}{3}})$$

for $m \approx n^{\frac{1}{3}} C_n^{\frac{2}{3}}$ and σ-subgaussian noise terms.

Definition 2.1. Let X be a real valued random variable. X is said to be subgaussian with variance proxy σ^2 if $\mathbb{E}(X) = 0$ and

$$\mathbb{E}(e^{\lambda X}) \leq e^{\frac{\lambda^2 \sigma^2}{2}}$$

is satisfied for all $\lambda \in \mathbb{R}$. In such cases, we call X σ-subgaussian.

For proving Theorem 2.2, we will need the following lemma about the variance of σ-subgaussian random variables.

Lemma 2.1. Suppose X is σ-subgaussian. Then $\mathrm{Var}(X) \leq \sigma^2$.

Proof. Performing Taylor expansion around $\lambda = 0$, we have by the Dominated Convergence Theorem that

$$\sum_{n=0}^{\infty} \frac{\lambda^n}{n!} \mathbb{E}(X^n) \overset{\text{DCT}}{=} \mathbb{E}(e^{\lambda X}) \leq e^{\frac{\lambda^2 \sigma^2}{2}} = \sum_{n=0}^{\infty} \frac{\lambda^{2n} \sigma^{2n}}{2^n n!}.$$

Hence

$$1 + \lambda \mathbb{E}(X) + \frac{\lambda^2}{2} \mathbb{E}(X^2) \leq 1 + \frac{\lambda^2 \sigma^2}{2} + o(\lambda^2).$$

Since $\mathbb{E}(X) = 0$, the result follows after dividing by λ^2 and letting $\lambda \to 0$. □

We are now ready to prove the error bound on the hypothetical forecaster. The key idea is to decompose the total cumulative error into the sum of errors within each of the m batches. For each batch, we will separate the error into a bias component, stemming from the deviation of the signal θ_t, and a variance component, stemming from the σ-subgaussian noise. By bounding these components using the total variation constraint (2.1) and Lemma 2.1, and then summing over all batches, we arrive at the desired result.

Theorem 2.2. [3] Let $n, m \geq 1$, $\sigma > 0$, and $C_n > 0$. Let $\theta_1, \ldots, \theta_n \in \mathbb{R}$ be any sequence such that $TV(\theta_{1:n}) \leq C_n$ and $|\theta_1| \leq B$. Suppose that $1 = t_1 \leq \cdots \leq t_{m+1} = n + 1$ are defined such that (2.1) is satisfied. Then the hypothetical forecasts $\widetilde{\mu}_t$ defined in (2.2) satisfy

$$\mathcal{R}_n(\widetilde{\mu}_{1:n}, \theta_{1:n}) \leq B^2 + TV(\theta_{1:n})^2 + m\sigma^2(1 + \log n) + \frac{n}{m^2} TV(\theta_{1:n})^2.$$

Proof. Let $m \geq 1$ be the total number of batches and number the batches $1, \ldots, m$ according to the restart times $1 = t_1 \leq \cdots \leq t_{m+1} = n + 1$. By (2.1) we have for all $1 \leq i \leq m$

$$TV(\theta_{t_i:(t_{i+1}-1)}) = \sum_{t=t_i+1}^{t_{i+1}-1} |\theta_t - \theta_{t-1}| \leq \frac{TV(\theta_{1:n})}{m} \leq \frac{C_n}{m}.$$

Fix a batch i. Using that $\widetilde{\mu}_t = \bar{y}_{t_k:(t-1)}$ for $t \in \{t_k + 1, \ldots, t_{k+1}\}$ for all $1 \leq k \leq m$, the cumulative error within

this batch can be written as

$$\begin{aligned}
\mathcal{R}_i &:= \sum_{t=t_i}^{t_{i+1}-1} \mathbb{E}\left[(\widetilde{\mu}_t - \theta_t)^2\right] \\
&= \mathbb{E}\left[(\widetilde{\mu}_{t_i} - \theta_{t_i})^2\right] + \sum_{t=t_i+1}^{t_{i+1}-1} \mathbb{E}\left[(\widetilde{\mu}_t - \theta_t)^2\right] \\
&= \mathbb{E}\left[(\bar{y}_{t_{i-1}:(t_i-1)} - \theta_{t_i})^2\right] + \sum_{t=t_i+1}^{t_{i+1}-1} \mathbb{E}\left[(\bar{y}_{t_i:(t-1)} - \theta_t)^2\right] \\
&= \mathbb{E}\left[(\bar{\theta}_{t_{i-1}:(t_i-1)} + \bar{Z}_{t_{i-1}:(t_i-1)} - \theta_{t_i})^2\right] + \sum_{t=t_i+1}^{t_{i+1}-1} \mathbb{E}\left[(\bar{\theta}_{t_i:(t-1)} + \bar{Z}_{t_i:(t_{i+1}-1)} - \theta_t)^2\right]
\end{aligned}$$

since $y_t = \theta_t + Z_t$. Rewriting, we have

$$\begin{aligned}
\mathcal{R}_i &= \mathbb{E}\left[(\bar{\theta}_{t_{i-1}:(t_i-1)} + \bar{Z}_{t_{i-1}:(t_i-1)} - \theta_{t_i})^2\right] + \sum_{t=t_i+1}^{t_{i+1}-1} \mathbb{E}\left[(\bar{\theta}_{t_i:(t-1)} + \bar{Z}_{t_i:(t-1)} - \theta_t)^2\right] \\
&= (\bar{\theta}_{t_{i-1}:(t_i-1)} - \theta_{t_i})^2 + 2(\bar{\theta}_{t_{i-1}:(t_i-1)} - \theta_{t_i})\mathbb{E}\left(\bar{Z}_{t_{i-1}:(t_i-1)}\right) + \mathbb{E}\left(\bar{Z}^2_{t_{i-1}:(t_i-1)}\right) \\
&\quad + \sum_{t=t_i+1}^{t_{i+1}-1} \left[(\bar{\theta}_{t_i:(t-1)} - \theta_t)^2 + 2(\bar{\theta}_{t_i:(t-1)} - \theta_t)\mathbb{E}\left(\bar{Z}_{t_i:(t_i-1)}\right) + \mathbb{E}\left(\bar{Z}^2_{t_i:(t-1)}\right)\right] \\
&= (\bar{\theta}_{t_{i-1}:(t_i-1)} - \theta_{t_i})^2 + \mathbb{E}\left(\bar{Z}^2_{t_{i-1}:(t_i-1)}\right) + \sum_{t=t_i+1}^{t_{i+1}-1} \left[(\bar{\theta}_{t_i:(t-1)} - \theta_t)^2 + \mathbb{E}\left(\bar{Z}^2_{t_i:(t-1)}\right)\right]
\end{aligned}$$

since Z_t are i.i.d. with $\mathbb{E}(Z_t) = 0$. Then, since Z_t are σ-subgaussian, we have

$$\begin{aligned}
\mathcal{R}_i &= (\bar{\theta}_{t_{i-1}:(t_i-1)} - \theta_{t_i})^2 + \mathbb{E}\left(\bar{Z}^2_{t_{i-1}:(t_i-1)}\right) + \sum_{t=t_i+1}^{t_{i+1}-1} \left[(\bar{\theta}_{t_i:(t-1)} - \theta_t)^2 + \mathbb{E}\left(\bar{Z}^2_{t_i:(t-1)}\right)\right] \\
&= (\bar{\theta}_{t_{i-1}:(t_i-1)} - \theta_{t_i})^2 + \frac{1}{(t_i - t_{i-1})^2} \sum_{k=t_{i-1}}^{t_i-1} \mathbb{E}\left(Z_t^2\right) + \sum_{t=t_i+1}^{t_{i+1}-1} \left[(\bar{\theta}_{t_i:(t-1)} - \theta_t)^2 + \frac{1}{(t-t_i)^2} \sum_{k=t_i}^{t-1} \mathbb{E}\left(Z_t^2\right)\right] \\
&\leq (\bar{\theta}_{t_{i-1}:(t_i-1)} - \theta_{t_i})^2 + \frac{\sigma^2}{t_i - t_{i-1}} + \sum_{t=t_i+1}^{t_{i+1}-1} \left[(\bar{\theta}_{t_i:(t-1)} - \theta_t)^2 + \frac{\sigma^2}{t - t_i}\right]
\end{aligned}$$

where the last line follows from Lemma 2.1. Denoting $\theta_0 = 0$ and summing over all batches gives the upper-bound on the cumulative error

$$\begin{aligned}
\mathcal{R}_n(\widetilde{\mu}_{1:n}, \theta_{1:n}) := \sum_{i=1}^{m} \mathcal{R}_i &\leq \sum_{i=1}^{m} \left[(\bar{\theta}_{t_{i-1}:(t_i-1)} - \theta_{t_i})^2 + \frac{\sigma^2}{t_i - t_{i-1}} + \sum_{t=t_i+1}^{t_{i+1}-1} \left((\bar{\theta}_{t_i:(t-1)} - \theta_t)^2 + \frac{\sigma^2}{t - t_i}\right)\right] \\
&\leq |\theta_1|^2 + \left(\sum_{i=2}^{m} \left|\bar{\theta}_{t_{i-1}:(t_i-1)} - \theta_{t_i}\right|\right)^2 + \sum_{i=1}^{m} \left[\frac{\sigma^2}{t_i - t_{i-1}} + \sum_{t=t_i+1}^{t_{i+1}-1} \left((\bar{\theta}_{t_i:(t-1)} - \theta_t)^2 + \frac{\sigma^2}{t - t_i}\right)\right] \\
&= |\theta_1|^2 + \left(\sum_{i=2}^{m} \left|\bar{\theta}_{t_{i-1}:(t_i-1)} - \theta_{t_i}\right|\right)^2 + \sum_{i=1}^{m} \left[\sum_{t=t_i+1}^{t_{i+1}-1} (\bar{\theta}_{t_i:(t-1)} - \theta_t)^2 + \sum_{t=t_i+1}^{t_{i+1}} \frac{\sigma^2}{t - t_i}\right].
\end{aligned}$$

Isolating the last term, we see that

$$\begin{aligned}\sum_{i=1}^{m}\sum_{t=t_i+1}^{t_{i+1}}\frac{\sigma^2}{t-t_i} &= \sigma^2\sum_{i=1}^{m}\sum_{k=1}^{t_{i+1}-t_i}\frac{1}{k} = \sigma^2\sum_{i=1}^{m}\left[1+\sum_{k=2}^{t_{i+1}-t_i}\frac{1}{k}\right] = \sigma^2\sum_{i=1}^{m}\left[1+\sum_{k=2}^{t_{i+1}-t_i}\int_{k-1}^{k}\frac{1}{k}dx\right] \\ &\leq \sigma^2\sum_{i=1}^{m}\left[1+\sum_{k=2}^{t_{i+1}-t_i}\int_{k-1}^{k}\frac{1}{x}dx\right] = \sigma^2\sum_{i=1}^{m}\left[1+\int_{1}^{t_{i+1}-t_i}\frac{1}{x}dx\right] \\ &= \sigma^2\sum_{i=1}^{m}1+\log\left(t_{i+1}-t_i\right) \leq \sigma^2\sum_{i=1}^{m}1+\log n = m\sigma^2(1+\log n).\end{aligned}$$

We also see for the second and third terms that

$$\begin{aligned}\left|\bar{\theta}_{t_i:(t-1)}-\theta_t\right| &= \left|\frac{1}{t-t_i}\sum_{k=t_i}^{t-1}\theta_k-\theta_t\right| \leq \frac{1}{t-t_i}\sum_{k=t_i}^{t-1}\left|\theta_k-\theta_t\right| \leq \max_{k\in\{t_i,\ldots,t-1\}}\left|\theta_k-\theta_t\right| \\ &= \max_{k\in\{t_i,\ldots,t-1\}}\left|\sum_{l=k}^{t-1}\theta_l-\theta_{l+1}\right| \leq \max_{k\in\{t_i,\ldots,t-1\}}\sum_{l=k}^{t-1}\left|\theta_l-\theta_{l+1}\right| = \sum_{k=t_i}^{t-1}\left|\theta_k-\theta_{k+1}\right|.\end{aligned}$$

Filling back in, we have

$$\begin{aligned}\mathcal{R}_n(\widetilde{\mu}_{1:n},\theta_{1:n}) &\leq \left|\theta_1\right|^2+\left(\sum_{i=2}^{m}\left|\bar{\theta}_{t_{i-1}:(t_i-1)}-\theta_{t_i}\right|\right)^2+\sum_{i=1}^{m}\left[\sum_{t=t_i+1}^{t_{i+1}-1}(\bar{\theta}_{t_i:(t-1)}-\theta_t)^2+\sum_{t=t_i+1}^{t_{i+1}}\frac{\sigma^2}{t-t_i}\right] \\ &\leq B^2+\left(\sum_{i=2}^{m}\sum_{k=t_{i-1}}^{t_i-1}\left|\theta_k-\theta_{k+1}\right|\right)^2+\sum_{i=1}^{m}\sum_{t=t_i+1}^{t_{i+1}-1}\left(\sum_{k=t_i}^{t-1}\left|\theta_k-\theta_{k+1}\right|\right)^2+m\sigma^2(1+\log n) \\ &\leq B^2+\left(\sum_{i=2}^{m}\frac{C_n}{m}\right)^2+\sum_{i=1}^{m}\sum_{t=t_i+1}^{t_{i+1}-1}\left(\frac{C_n}{m}\right)^2+m\sigma^2(1+\log n) \\ &= B^2+\left(\frac{m-1}{m}C_n\right)^2+\frac{C_n^2}{m^2}\sum_{i=1}^{m}(t_{i+1}-(t_i+1))+m\sigma^2(1+\log n) \\ &\leq B^2+C_n^2+\frac{C_n^2}{m^2}(t_{i+1}-(t_i+1))+m\sigma^2(1+\log n) \\ &\leq B^2+C_n^2+\frac{nC_n^2}{m^2}+m\sigma^2(1+\log n).\end{aligned}$$

By the first-order optimality condition, we find that this bound is minimised for

$$\frac{\partial}{\partial m}\left(B^2+C_n^2+\frac{nC_n^2}{m^2}+m\sigma^2(1+\log n)\right) = -\frac{2n}{m^3}C_n^2+\sigma^2(1+\log n) = 0 \Leftrightarrow m = \left(\frac{2nC_n^2}{\sigma^2(1+\log n)}\right)^{\frac{1}{3}}.$$

Finally, this choice for m yields

$$\mathcal{R}_n(\widetilde{\mu}_{1:n},\theta_{1:n}) \leq B^2+C_n^2+\frac{3}{2^{\frac{2}{3}}}n^{\frac{1}{3}}C_n^{\frac{2}{3}}\sigma^{\frac{4}{3}}(1+\log n)^{\frac{2}{3}}. \qquad \square$$

Thus, Theorem 2.2 establishes a the theoretical benchmark for the optimal rate of convergence, $\tilde{\mathcal{O}}(n^{\frac{1}{3}}C_n^{\frac{2}{3}})$ However, this oracle is predicated on knowing the optimal restart times $t_1,\ldots,t_{m+1}$ in advance, a condition that cannot be met in a true online setting. This motivates the central challenge of the next section: designing a practical algorithm that can adaptively learn these restart points and achieve a comparable performance guarantee.

2.2 Follow-the-Leading-History

The challenge in the practical application of this oracle forecaster is to efficiently compute the restart times $1 = t_1, \ldots, t_{m+1} = n + 1$ which satisfy (2.1). The problem is that if we want to preserve the benefits of an online approach, we cannot simply use the naive approach of checking all possible choices of $t_1, \ldots, t_{m+1}$ after the fact, never mind that this would be very computationally expensive. This is precisely the challenge that the Follow-the-Leading-History (FLH) algorithm is designed to address.

Any online algorithm which makes use of moving averages must then also compute the restart times in an online fashion. This can be accomplished in a number of ways, such as change-point methods for example, however the simplest way is to just not do it directly. Instead, we may combine the predictions of many moving averages with different look-back windows, this is called meta-aggregation. By running multiple simple forecasters in parallel, each corresponding to a different potential restart time, a meta-aggregation algorithm such as FLH can dynamically learn to favour the best-performing expert(s) at any given moment thereby gaining the ability to adapt to shifts in the data's underlying structure.

At its core, FLH is an online meta-aggregation algorithm that combines and manages the weights of multiple online learning algorithms, or experts. Each expert is assigned a corresponding weight which is iteratively updated at the end of each round with exponential reweighting. By viewing each potential start time k as a separate expert, FLH provides a mechanism for adaptively learning the best optimal restart times $t_1, \ldots, t_m$ in an online fashion, thereby mimicking the behaviour of our clairvoyant oracle.

Algorithm 1 Follow-the-Leading-History [3, 1]

Input: black box algorithm $\mathcal{A}$, learning parameter $\alpha > 0$

1: **Init:** Weight vector $\boldsymbol{v}_0 = (v_0^{(1)}, \ldots, v_0^{(n)}) = (0, \ldots, 0)$

2: **for** $t = 1, \ldots, n$ **do**

3: Start a new instance of algorithm $\mathcal{A}$ denoted by $\mathcal{A}_t$ and assign the weights $v_t^{(t)} = \frac{1}{t}$ and $\widehat{v}_t^{(t)} = \frac{1}{t}$.

4: Normalise the weight of each expert $i \in \{1, \ldots, t-2\}$ so that

$$\widehat{v}_t^{(i)} = \left(1 - \frac{1}{t-1}\right) \frac{v_t^{(i)}}{\sum_{j=1}^{t-2} v_t^{(j)}} \tag{2.3}$$

and update $\widehat{v}_t^{(t-1)} = \widehat{v}_{t-1}^{(t-1)}$.

5: Receive the prediction $\widehat{\mu}_t^{(i)}$ from each black box algorithm $\mathcal{A}_i$, $i \in \{1, \ldots, t-1\}$.

6: Predict

$$\widehat{\mu}_t = \sum_{i=1}^{t-1} \widehat{v}_t^{(i)} \widehat{\mu}_t^{(i)}. \tag{2.4}$$

and observe $y_t \in \mathbb{R}$.

7: Set $v_{t+1}^{(t)} = v_t^{(t)}$ and update the weights for each $i \in \{1, \ldots, t-1\}$

$$v_{t+1}^{(i)} = v_t^{(i)} \exp\left(-\alpha f_t(\widehat{\mu}_t^{(i)})\right).$$

At each round $t = 1, \ldots, n$, the algorithm carries out a series of steps to achieve its goal of forecasting. At step 3, a new base online algorithm $\mathcal{A}_t$ is created for the data starting from the same round t, and is assigned a weight of $\frac{1}{t}$. Crucially, this expert ignores earlier observations and focuses exclusively on minimizing static regret over its own time segment. These experts are combined using an exponentially weighted average, whose weights are calculated in step 4, calculated in step 5 and step 6 to produce the final prediction. Afterwards, the weights are updated in step 7 and the loop restarts.

There is one key difference between the formulation of Algorithm 1 and that which may be found in [1].

This difference is a result of the black box algorithm $\mathcal{A}$. We have kept this notation for generality, however in this work the black box algorithm $\mathcal{A}_k$ refers to a moving average with the 'start' of its look back window set at $t = k$. Observe then that at $t = 1$, while we have indeed initialised $\mathcal{A}_1$, it has no data on which it may make a prediction. As a consequence, we have a pool of only $k - 1$ moving averages to combine at time step $t = k$. This has a knock-on effect and changes the manner in which we perform the normalisation step. As a result, the specific details of the algorithm formulation are changed, but the 'spirit' of FLH is not. For clarity, Example 2.1 walks through the first 3 iterations of the algorithm in explicit detail and the sceptic reader is furthermore referred to Appendix C.

Example 2.1. For $t = 1$, we do

1. We initialise $\mathcal{A}_1$ with weights $v_1^{(1)} = \frac{1}{1} = 1$ and $\widehat{v}_1^{(1)} = 1$. We have 0 experts for predictions.
2. We have no weights to normalise and no update to perform.
3. We receive no predictions from any experts.
4. Since we received no expert prediction, we do not make any prediction. We observe y_1.
5. We update $v_2^{(1)} = v_1^{(1)} = 1$.

For $t = 2$, we do

1. We initialise $\mathcal{A}_2$ with weights $v_2^{(2)} = \frac{1}{2}$ and $\widehat{v}_2^{(2)} = \frac{1}{2}$. We have 1 expert for predictions $\{\mathcal{A}_1\}$.
2. We have no weights to normalise. We perform the update $\widehat{v}_2^{(1)} = \widehat{v}_1^{(1)} = 1$.
3. We receive prediction $\widehat{\mu}_2^{(1)} = y_1$ from $\mathcal{A}_1$.
4. We predict $\widehat{\mu}_2 = \widehat{v}_2^{(1)}\widehat{\mu}_2^{(1)} = y_1$. We observe y_2.
5. We perform weight updates $v_3^{(2)} = v_2^{(2)} = \frac{1}{2}$ and update
$$v_3^{(1)} = v_2^{(1)} \exp\left(-\alpha\left(y_2 - \widehat{\mu}_2^{(1)}\right)^2\right) = \exp\left(-\alpha\left(y_2 - y_1\right)^2\right).$$

For $t = 3$, we do

1. We initialise $\mathcal{A}_3$ with weights $v_3^{(3)} = \frac{1}{3}$ and $\widehat{v}_3^{(3)} = \frac{1}{3}$. We have 2 experts for predictions $\{\mathcal{A}_1, \mathcal{A}_2\}$.
2. We perform the update $\widehat{v}_3^{(2)} = \widehat{v}_2^{(2)} = \frac{1}{2}$. We normalise
$$\widehat{v}_3^{(1)} = \left(1 - \frac{1}{3-1}\right)\frac{v_3^{(1)}}{v_3^{(1)}} = \frac{1}{2}$$
3. We receive prediction $\widehat{\mu}_3^{(1)} = \frac{y_1+y_2}{2}$ from $\mathcal{A}_1$ and $\widehat{\mu}_3^{(2)} = y_2$ from $\mathcal{A}_2$.
4. We predict $\widehat{\mu}_3 = \widehat{v}_3^{(1)}\widehat{\mu}_3^{(1)} + \widehat{v}_3^{(2)}\widehat{\mu}_3^{(2)} = \frac{y_1+y_2}{4} + \frac{y_2}{2} = \frac{y_1+3y_2}{4}$. We observe y_3.
5. We perform weight update $v_4^{(3)} = v_3^{(3)} = \frac{1}{3}$. We also perform updates
$$v_4^{(1)} = v_3^{(1)} \exp\left(-\alpha\left(y_3 - \widehat{\mu}_3^{(1)}\right)^2\right) = \exp\left(-\alpha\left(\left(y_3 - \frac{y_1+y_2}{2}\right)^2 + \left(y_2 - y_1\right)^2\right)\right)$$
and
$$v_4^{(2)} = v_3^{(2)} \exp\left(-\alpha\left(y_3 - \widehat{\mu}_3^{(2)}\right)^2\right) = \frac{1}{2}\exp\left(-\alpha\left(y_3 - y_2\right)^2\right).$$

2.2.1 α-exp-concave Loss Functions

Note that Algorithm 1 is designed to only work for the class of α-exp-concave loss functions. We say a function f is α-exp-concave if $e^{-\alpha f}$ is a concave function for some choice of $\alpha > 0$. There exists a version of Algorithm 1 which is suitable for general convex loss functions - the key difference is that instead of predicting a convex combinations of expert predictions, we choose one single expert for prediction according to some probability vector v_t - however, we may use the formulation for α-exp-concave loss functions because of the results of Lemma 2.3.

Lemma 2.3. For $1 \leq t \leq n$, define $y_t = \theta_t + Z_t$ where $|\theta_t| \leq B$ for some $B > 0$ and Z_t are σ-subgaussian random variables. Let $0 < \delta < 1$. For all $1 \leq t \leq n$, the mapping $y \mapsto e^{-\alpha(y-y_t)^2}$ restricted to $D = \left[-\frac{1}{\sqrt{2\alpha}} + y_t, \frac{1}{\sqrt{2\alpha}} + y_t\right]$ is concave with $\alpha = \frac{C'}{\log \frac{2n}{\delta}}$ for some $C' > 0$.

Essentially, the results of Lemma 2.3 imply that that squared loss $f_t(y) = (y - y_t)^2$ belongs to the class of α-exp-concave loss functions with probability $1-\delta$ for any $t = 1, \ldots, n$. As a result, we may use the formulation of Algorithm 1 for α-exp-concave functions in our problem setting. The proof of Lemma 2.3 may be split into 2 additional lemmas and a final argument.

Lemma 2.4. Suppose X is σ-subgaussian. Then

$$\mathbb{P}\left(X > t\right) \leq e^{-\frac{t^2}{2\sigma^2}}.$$

Proof. By Markov's inequality, we have for any $s > 0$

$$\mathbb{P}(X > t) = \mathbb{P}\left(e^{sX} > e^{st}\right) \leq \frac{\mathbb{E}(e^{sX})}{e^{st}} \leq e^{\frac{s^2\sigma^2}{2} - st}.$$

By the first-order optimality condition, this bound is minimised for $s = \frac{t}{\sigma^2}$ and the result follows. □

Lemma 2.5. Let $y_1, \ldots, y_n$ be defined as in Lemma 2.3. Then for all $0 < \delta < 1$, there exists some $C > 0$ such that

$$\mathbb{P}\left(\max_{1 \leq t \leq n} |y_t| \leq C\sqrt{\log \frac{2n}{\delta}}\right) = 1 - \delta.$$

Proof. Since $|y_t| = |\theta_t + Z_t| \leq |\theta_t| + |Z_t| \leq B + |Z_t|$, we have

$$\begin{aligned}\mathbb{P}\left(\max_{1 \leq t \leq n} |y_t| > K\right) &= \mathbb{P}\left(\max_{1 \leq t \leq n} |Z_t| > K - B\right) = \mathbb{P}\left(\max_{1 \leq t \leq 2n} Z_t > K - B\right) \quad (Z_{n+i} := -Z_i \text{ for } i = 1, \ldots, n) \\ &\leq \sum_{t=1}^{2n} \mathbb{P}\left(Z_t > K - B\right) \leq 2n \exp\left(-\frac{(K-B)^2}{2\sigma^2}\right)\end{aligned}$$

where the last inequality is Lemma 2.4 applied to each identically distributed Z_t. Setting $2n \exp\left(-\frac{(K-B)^2}{2\sigma^2}\right) = \delta$ and solving for K, we find that $K = B + \sqrt{2\sigma^2 \log \frac{2n}{\delta}} \leq C\sqrt{\log \frac{2n}{\delta}}$. Now, we may divide by $\sqrt{\log \frac{2n}{\delta}}$ to find that

$$\frac{B}{\sqrt{\log \frac{2n}{\delta}}} + \sqrt{2\sigma^2} \leq C.$$

The left hand side is maximised as $\log \frac{2n}{\delta} \to \log 2$, hence the result is obtained for $C = \frac{B}{\sqrt{\log 2}} + \sqrt{2\sigma^2}$. □

Proof of Lemma 2.3. The mapping $y \mapsto e^{-\alpha(y-y_t)^2}$ is concave if $\frac{\mathrm{d}^2}{\mathrm{d}y^2} e^{-\alpha(y-y_t)^2} \leq 0$. Then, since

$$\frac{\mathrm{d}^2}{\mathrm{d}y^2} e^{-\alpha(y-y_t)^2} = -2\alpha e^{-\alpha(y-y_t)^2}(1 - 2\alpha(y-y_t)^2),$$

we see that this is satisfied for $|y - y_t| \leq \frac{1}{\sqrt{2\alpha}}$. Restricting y to $\left[-\frac{1}{\sqrt{2\alpha}} + y_t, \frac{1}{\sqrt{2\alpha}} + y_t\right]$, we see that

$$-\frac{1}{\sqrt{2\alpha}} = -\frac{1}{\sqrt{2\alpha}} + y_t - y_t \leq y - y_t \leq \frac{1}{\sqrt{2\alpha}} + y_t - y_t = \frac{1}{\sqrt{2\alpha}}.$$

Deriving α, by Lemma 2.5 we have for all $1 \leq t \leq n$ that

$$|y - y_t| \leq 2C\sqrt{\log \frac{2n}{\delta}} \leq \frac{1}{\sqrt{2\alpha}} \Rightarrow \ \alpha = \frac{1}{8C^2 \log \frac{2n}{\delta}}$$

with probability $1 - \delta$, which completes the proof. □

2.2.2 Performance Guarantees

In their paper in which they first introduce FLH, [1] show that it has quite strong adaptive regret guarantees. For our purposes however, we are only interested in the adaptive regret achieved by any black box algorithm $\mathcal{A}_r$. For completeness, we include the statement and and adapt their proof to the formulation of Algorithm 1.

Lemma 2.6. [1] Let $1 \leq r \leq s \leq n$ with $r, s \in \mathbb{N}$. For any $I = [r+1, s]$, the regret incurred by FLH in I with respect to $\mathcal{A}_r$, is at most $\frac{2}{\alpha}(\ln r + \ln |I|)$.

We will require an additional lemma for the proof of Lemma 2.6.

Lemma 2.7. [1] Define $\widehat{p}_t^{(i)} := \frac{\widehat{v}_t^{(i)} e^{-\alpha f_t(\widehat{\mu}_t^{(i)})}}{\sum_{j=1}^{t-1} \widehat{v}_t^{(j)} e^{-\alpha f_t(\widehat{\mu}_t^{(j)})}}$ for any $i = 1, \ldots, t-1$. For any α-exp-concave loss function f_t, we have

$$f_t(\widehat{\mu}_t) - f_t(\widehat{\mu}_t^{(t-1)}) \leq \frac{1}{\alpha}\left(\log \widehat{p}_t^{(t-1)} + \log(t-1)\right) \tag{2.5}$$

and for any $i \in \{3, 4 \ldots, t-2\}$, we also have

$$f_t(\widehat{\mu}_t) - f_t(\widehat{\mu}_t^{(i)}) \leq \frac{1}{\alpha}\left(\log \widehat{p}_t^{(i)} + \log \widehat{p}_{t-1}^{(i)} + \frac{2}{t}\right). \tag{2.6}$$

Proof. By the α-exp-concavity of f_t, we have

$$e^{-\alpha f_t(\widehat{\mu}_t)} = e^{-\alpha f_t\left(\sum_{j=1}^{t-1} \widehat{v}_t^{(j)} \widehat{\mu}_t^{(j)}\right)} \geq \sum_{j=1}^{t-1} \widehat{v}_t^{(j)} e^{-\alpha f(\widehat{\mu}_t^{(j)})}$$

$$\Leftrightarrow f_t(\widehat{\mu}_t) \leq \frac{1}{\alpha} \log \sum_{j=1}^{t-1} \widehat{v}_t^{(j)} e^{-\alpha f_t(\widehat{\mu}_t^{(j)})}.$$

Hence

$$\begin{aligned} f_t(\widehat{\mu}_t) - f_t(\widehat{\mu}_t^{(i)}) &\leq \frac{1}{\alpha}\left(\log e^{-\alpha f_t(\widehat{\mu}_t^{(i)})} - \log \sum_{j=1}^{t-1} \widehat{v}_t^{(j)} e^{-\alpha f_t(\widehat{\mu}_t^{(j)})}\right) \\ &= \frac{1}{\alpha} \log\left(\frac{e^{-\alpha f_t(\widehat{\mu}_t^{(i)})}}{\sum_{j=1}^{t-1} \widehat{v}_t^{(j)} e^{-\alpha f_t(\widehat{\mu}_t^{(j)})}}\right) \\ &= \frac{1}{\alpha} \log\left(\frac{1}{\widehat{v}_t^{(i)}} \frac{\widehat{v}_t^{(i)} e^{-\alpha f_t(\widehat{\mu}_t^{(i)})}}{\sum_{j=1}^{t-1} \widehat{v}_t^{(j)} e^{-\alpha f_t(\widehat{\mu}_t^{(j)})}}\right) = \frac{1}{\alpha} \log \frac{\widehat{p}_t^{(i)}}{\widehat{v}_t^{(i)}}. \end{aligned}$$

Now, we have by definition that $\widehat{v}_t^{(t-1)} = \widehat{v}_{t-1}^{(t-1)} = \frac{1}{t-1}$, hence $\log \widehat{v}_t^{(t-1)} = -\log(t-1)$ and we see that for $i = t-1$

$$f_t(\widehat{\mu}_t) - f_t(\widehat{\mu}_t^{(t-1)}) \leq \frac{1}{\alpha} \log \frac{\widehat{p}_t^{(t-1)}}{\widehat{v}_t^{(t-1)}} = \frac{1}{\alpha} \left(\log \widehat{p}_t^{(t-1)} + \log(t-1) \right)$$

which proves (2.5). We also have for any $i \in \{3, 4, \ldots, t-2\}$, by definition, that

$$\begin{aligned}
\widehat{v}_t^{(i)} &= \left(1 - \frac{1}{t-1}\right) \frac{v_t^{(i)}}{\sum_{j=1}^{t-2} v_t^{(j)}} = \left(1 - \frac{1}{t-1}\right) \frac{v_{t-1}^{(i)} e^{-\alpha f_{t-1}(\widehat{\mu}_{t-1}^{(i)})}}{\sum_{j=1}^{t-2} v_{t-1}^{(j)} e^{-\alpha f_{t-1}(\widehat{\mu}_{t-1}^{(j)})}} \\
&= \left(1 - \frac{1}{t-1}\right) \frac{\widehat{v}_{t-1}^{(i)} e^{-\alpha f_{t-1}(\widehat{\mu}_{t-1}^{(i)})}}{\sum_{j=1}^{t-2} \widehat{v}_{t-1}^{(j)} e^{-\alpha f_{t-1}(\widehat{\mu}_{t-1}^{(j)})}} = \left(1 - \frac{1}{t-1}\right) \widehat{p}_{t-1}^{(i)},
\end{aligned}$$

and hence

$$\log \widehat{v}_t^{(i)} = \log \widehat{p}_t^{(i)} + \log \left(1 - \frac{1}{t-1}\right) \geq \log \widehat{p}_t^{(i)} - \frac{2}{t}.$$

Filling in, we see that

$$f_t(\widehat{\mu}_t) - f_t(\widehat{\mu}_t^{(i)}) \leq \frac{1}{\alpha} \log \frac{\widehat{p}_t^{(i)}}{\widehat{v}_t^{(i)}} \leq \frac{1}{\alpha} \left(\log \widehat{p}_t^{(i)} - \log \widehat{p}_{t-1}^{(i)} + \frac{2}{t} \right). \qquad \square$$

Proof of Lemma 2.6. We have

$$\begin{aligned}
\sum_{t=r+1}^{s} f_t(\widehat{\mu}_t) - f_t(\widehat{\mu}_t^{(r)}) &= f_{r+1}(\widehat{\mu}_{r+1}) - f_{r+1}(\widehat{\mu}_{r+1}^{(r)}) + \sum_{t=r+2}^{s} f_t(\widehat{\mu}_t) - f_t(\widehat{\mu}_t^{(r)}) \\
&\leq \frac{1}{\alpha} \left(\log \widehat{p}_{r+1}^{(r)} + \log r + \sum_{t=r+2}^{s} \left(\log \widehat{p}_t^{(r)} - \log \widehat{p}_{t-1}^{(r)} + \frac{2}{t} \right) \right) \\
&= \frac{1}{\alpha} \left(\log r + \log \widehat{p}_s^{(r)} + \sum_{t=r+2}^{s} \frac{2}{t} \right)
\end{aligned}$$

where the second inequality follows from (2.5) and (2.6), and the last equality is because of the telescoping sum.

It is clear to see that $\widehat{p}_s^{(r)} \leq 1$, because of the normalisation constant, and so $\log \widehat{p}_s^{(r)} \leq 0$. Zooming in on the last term, we use a similar integral trick as in the proof of Theorem 2.2 to show that $\sum_{t=r+2}^{s} \frac{2}{t} \leq 2 \log(s-r) = 2 \log |I|$. Filling in, we obtain

$$\begin{aligned}
\sum_{t=r}^{s} f_t(\widehat{\mu}_t) - f_t(\widehat{\mu}_t^{(r)}) &\leq \frac{1}{\alpha} \left(\log r + \log \widehat{p}_s^{(r)} + \sum_{t=r+2}^{s} \frac{2}{t} \right) \\
&\leq \frac{1}{\alpha} \left(\log r + 2 \log |I| \right) \leq \frac{2}{\alpha} \left(\log r + \log |I| \right). \qquad \square
\end{aligned}$$

Finally, we arrive at our main result. In Theorem 2.8, we show that Algorithm 1 with moving averages as sub-routines does indeed achieve a cumulative error of optimal order with independent σ-subgaussian terms.

Theorem 2.8. [3] For $1 \leq t \leq n$, define $y_t = \theta_t + Z_t$ where Z_t are independent σ-subgaussian random variables. Let $\theta_1, \ldots, \theta_n \in \mathbb{R}$ be such that $TV(\theta_{1:n}) \leq C_n$ and furthermore assume that $|\theta_t| \leq B$ for all $t \geq 1$. If moving average predictions are used as sub-routine of Algorithm 1, the cumulative error is upper-bounded as

$$\mathcal{R}_n(\widehat{\mu}_{1:n}, \theta_{1:n}) \leq \mathcal{O}(n^{\frac{1}{3}} C_n^{\frac{2}{3}} (\log n)^2)$$

with probability $1 - \delta$ for $0 < \delta < 1$.

Proof. By lemma 2.5, for all $1 \le t \le n$, each $|y_t| = |y_t + \delta_t|$ is bounded by $C\sqrt{\log \frac{2n}{\delta}}$ with probability $1-\delta$. Then by Lemma 2.3, we see that the mapping $y \mapsto (y - y_t)^2$ is α-exp-concave with $\alpha = \frac{C'}{\log \frac{2n}{\delta}}$ with high probability. Define $t_1, \dots, t_m$ as in Theorem 2.2. Applying Lemma 2.6 with $r = t_i$ and $s = t_{i+1} - 1$, we see that

$$\sum_{t=t_i}^{t_{i+1}-1} (\widehat{\mu}_t - y_t)^2 - (\widehat{\mu}_t^{(t_i)} - y_t)^2 \le \frac{2}{\alpha}\left(\log t_i + \log\left(t_{i+1} - 1 - t_i\right)\right) \le \frac{4 \log n}{\alpha} \le \mathcal{O}((\log n)^2).$$

Note that the subroutines in Algorithm 1 are moving averages i.e. $\widehat{\mu}_t^{(t_i)} = \bar{y}_{t_i:(t-1)}$. Hence, we can use (2.2) and that we have m restart times to sum over $i = 1, \dots, n$ and obtain

$$\begin{aligned}\sum_{t=1}^{n} (\widehat{\mu}_t - y_t)^2 - (\widetilde{\mu}_t - y_t)^2 &\le \sum_{i=1}^{m} \sum_{t=t_i}^{t_{i+1}-1} (\widehat{\mu}_t - y_t)^2 - (\widehat{\mu}_t^{(t_i)} - y_t)^2 \\ &\le \mathcal{O}(m(\log n)^2).\end{aligned}$$

Now, because $Z_t = y_t - \theta_t$, we have

$$\begin{aligned}\mathcal{R}_n(\widehat{\mu}_{1:n}, \theta_{1:n}) &:= \sum_{t=1}^{n} \mathbb{E}\left((\widehat{\mu}_t - \theta_t)^2\right) = \sum_{t=1}^{n} \mathbb{E}\left((\widehat{\mu}_t - y_t + y_t - \theta_t)^2\right) \\ &= \sum_{t=1}^{n} \mathbb{E}\left((\widehat{\mu}_t - y_t)^2 + (y_t - \theta_t)^2 + 2(\widehat{\mu}_t - y_t)(y_t - \theta_t)\right) \\ &= \sum_{t=1}^{n} \mathbb{E}\left((\widehat{\mu}_t - y_t)^2 + (y_t - \theta_t)^2 - 2y_t(y_t - \theta_t)\right) + \mathbb{E}\left(2\widehat{\mu}_t(y_t - \theta_t)\right) \\ &\overset{\text{Ind.}}{=} \sum_{t=1}^{n} \mathbb{E}\left((\widehat{\mu}_t - y_t)^2 - (y_t - \theta_t)^2 + 2(y_t - \theta_t)^2 - 2y_t(y_t - \theta_t)\right) + 2\mathbb{E}\left(\widehat{\mu}_t\right)\mathbb{E}\left(y_t - \theta_t\right) \\ &= \sum_{t=1}^{n} \mathbb{E}\left((\widehat{\mu}_t - y_t)^2 - (y_t - \theta_t)^2 + 2(y_t - \theta_t)^2 - 2y_t(y_t - \theta_t)\right) + \mathbb{E}\left(2\theta_t(y_t - \theta_t)\right)\end{aligned}$$

where the last equality uses that $\mathbb{E}(y_t - \theta_t) = \mathbb{E}(Z_t) = 0$. Continuing,

$$\begin{aligned}&= \sum_{t=1}^{n} \mathbb{E}\left((\widehat{\mu}_t - y_t)^2 - (y_t - \theta_t)^2 + 2(y_t - \theta_t)^2 + 2(\theta_t - y_t)(y_t - \theta_t)\right) \\ &= \sum_{t=1}^{n} \mathbb{E}\left((\widehat{\mu}_t - y_t)^2 - (y_t - \theta_t)^2\right) \\ &= \sum_{t=1}^{n} \mathbb{E}\left((\widehat{\mu}_t - y_t)^2 - (\widetilde{\mu}_t - \theta_t)^2 + (\widetilde{\mu}_t - \theta_t)^2 - (y_t - \theta_t)^2\right) \\ &= \sum_{t=1}^{n} \mathbb{E}\left((\widehat{\mu}_t - y_t)^2 - (\widetilde{\mu}_t - \theta_t)^2\right) + \mathbb{E}\left((\widetilde{\mu}_t - \theta_t)^2 - (y_t - \theta_t)^2\right) \\ &\le \mathcal{O}(m(\log n)^2) + \sum_{t=1}^{n} \mathbb{E}\left((\widetilde{\mu}_t - \theta_t)^2 - (y_t - \theta_t)^2\right).\end{aligned}$$

Isolating the second term, we have

$$\begin{aligned}
\sum_{t=1}^{n} \mathbb{E}\left((\widetilde{\mu}_t - \theta_t)^2 - (y_t - \theta_t)^2\right) &= \sum_{t=1}^{n} \mathbb{E}\left(\widetilde{\mu}_t^2 - 2\widetilde{\mu}_t^2 y_t + y_t^2 - (y_t^2 - 2y_t\theta_t + \theta_t^2)\right) \\
&= \sum_{t=1}^{n} \mathbb{E}\left(\widetilde{\mu}_t^2 + 2y_t(\theta_t - \widetilde{\mu}_t) - \theta_t^2\right) \\
&= \sum_{t=1}^{n} \mathbb{E}\left(\widetilde{\mu}_t^2 + 2(\theta_t + Z_t)(\theta_t - \widetilde{\mu}_t) - \theta_t^2\right) \\
&= \sum_{t=1}^{n} \mathbb{E}\left(\widetilde{\mu}_t^2 - 2\widetilde{\mu}_t\theta_t + \theta_t^2\right) + 2\left(\mathbb{E}\left(Z_t\theta_t\right) + \mathbb{E}\left(Z_t\widetilde{\mu}_t\right)\right) \\
&= \sum_{t=1}^{n} \mathbb{E}\left((\widetilde{\mu}_t - \theta_t)^2\right)
\end{aligned}$$

where the last equality holds because Z_t is independent from θ_t and $\widetilde{\mu}_t$, and $\mathbb{E}(Z_t) = 0$. Then we we have by Theorem 2.2 that

$$\begin{aligned}
\mathcal{R}_n(\widehat{\mu}_{1:n}, \theta_{1:n}) &\leq \mathcal{O}(m(\log n)^2) + \sum_{t=1}^{n} \mathbb{E}\left((\widetilde{\mu}_t - \theta_t)^2 - (y_t - \theta_t)^2\right) \\
&= \mathcal{O}(m(\log n)^2) + \mathcal{R}_n(\widetilde{\mu}_{1:n}, \theta_{1:n}) \\
&\leq \mathcal{O}(m(\log n)^2) + \mathcal{O}(n^{\frac{1}{3}} C_n^{\frac{2}{3}}) = \mathcal{O}(n^{\frac{1}{3}} C_n^{\frac{2}{3}} (\log n)^2)
\end{aligned}$$

where we let $m \approx n^{\frac{1}{3}} C_n^{\frac{2}{3}}$ - the optimal m found in Theorem 2.2. □

Chapter 3

Extension to Dependant Data

The results of Chapter 2 are all under the assumption that the error terms are independent, σ-subgaussian variables. In this chapter, we drop the assumption of strict independence and show that cumulative error achieved by FLH with moving averages as sub-routines for this more general case is the optimal $\tilde{\mathcal{O}}(n^{\frac{1}{3}}C_n^{\frac{2}{3}})$. Our plan of attack will be to adapt the proof of Theorem 2.8 to take into account dependent errors and for this we will need a considerable amount of machinery.

First, we introduce a formal dependence framework based on causal representations and physical dependence measures, allowing us to construct an m-dependent process that approximates the original error sequence. Second, we establish the stability of the FLH algorithm, proving that small perturbations in the input data result in controllably small changes in the output predictions. Finally, we introduce a Thinned-FLH (TFLH) scheme that leverages the approximated m-dependent structure to achieve near-optimal performance by balancing the approximation error against the statistical error from using less data.

3.1 Dependence Structure

The first step is to translate the dependent case into the independent case. We could perhaps choose to use the naive approach and construct independent copies of each Z_t. Then we indeed obtain an independent sequence, however at the cost of large approximation errors because we essentially approximate all sources of randomness in our model. Instead, if we make the reasonable assumption that errors which are 'close' to each other will be more dependent than errors that are 'far' apart, we can try and construct a sequence in which errors which are sufficiently 'far' apart are independent. This is formally called m-dependence.

Definition 3.1. Let $(X_t)_{t\in\mathbb{Z}}$ be a stochastic process. We say $(X_t)_{t\in\mathbb{Z}}$ is m-dependant if for all $k \in \mathbb{Z}$ the joint stochastic variables $(X_t)_{t\leq k}$ are independent of the joint stochastic variables $(X_t)_{t\geq k+m}$.

To transform $(Z_t)_{t\geq 1}$ into an m-dependent process, we need a way to isolate and manipulate sources of interdependencies. To do this, we represent the error process as a causal function of independent and identically distributed innovations. This causal representation is known as the Bernoulli shift model.

Definition 3.2. Let $\epsilon_t \overset{\text{iid}}{\sim} U(0,1)$ and $g : \mathbb{R}^\infty \to \overline{\mathbb{R}}$ be a measurable function. Then Z_t is said to be a causal Bernoulli shift model if

$$Z_t := g(\epsilon_t, \epsilon_{t-1}, \dots) \tag{3.1}$$

Notice that setting the distribution $\epsilon_t \sim U(0,1)$ is not a restriction because ϵ_t can be transformed to any distribution of choice using the quantile transform.

Lemma 3.1. Let $U \sim U(0,1)$ and let X be a random variable with the distribution function F_X. Then $F_X^{-1}(U) \overset{d}{=} X$.

Proof. Let $Y = F_X^{-1}(U)$, then

$$F_Y(y) = \mathbb{P}(Y \leq y) = \mathbb{P}(F_X^{-1}(U) \leq y) = \mathbb{P}(U \leq F_X(y)) = F_X(y). \qquad \square$$

Let Z_t be defined as according to (3.1). It is clear to see that Z_{t+m} is dependent on Z_t through the $\epsilon_t, \epsilon_{t-1}, \dots$ terms. It follows that we can transform $(Z_t)_{t\geq 1}$ into an m-dependant process by strategically introducing independent copies of ϵ_t. Let $\epsilon_k^{(t)}$ be an independent, identically distributed copy of ϵ_k assigned uniquely to Z_t. Define $Z_t^{(m)}$ as a copy of Z_t, defined such that all $(\epsilon_k)_{k\geq m}$ are replaced by $(\epsilon_k^{(t)})_{k\geq m}$. Then $Z_t^{(m)}$ can be written according to the formulation of (3.1) as

$$Z_t^{(m)} = g(\epsilon_t, \dots, \epsilon_{t-m+1}, \epsilon_{t-m}^{(t)}, \epsilon_{t-m-1}^{(t)}, \dots) \tag{3.2}$$

We see that that $Z_t^{(m)}$ is now independent of $Z_{t+m}^{(m)}$. Indeed, it is clear to see that $(Z_t^{(m)})_{t\leq k}$ are now independent of $(Z_t^{(m)})_{t\geq k+m}$ so that $(Z_t^{(m)})_{t\in\mathbb{Z}}$ is an m-dependant process.

We have now constructed $(Z_t^{(m)})_{t\in\mathbb{Z}}$ so that any interdependence are originating solely from the first m terms of any given $Z_t^{(m)}$, because each $Z_t^{(m)}$ has its own unique set of independent copies. However, this approach introduces an approximation error and we use the physical dependence measure to control for this error.

To define the physical dependence measure, we first introduce some useful notation. Let $(\epsilon'_k)_{k\in\mathbb{Z}}$ be independent copies of $(\epsilon_k)_{k\in\mathbb{Z}}$. Then we may define the vectors

$$\begin{aligned}
\boldsymbol{\epsilon}_k &= (\epsilon_k, \epsilon_{k-1}, \dots), \\
\boldsymbol{\epsilon}'_{k,h} &= (\epsilon_k, \dots, \epsilon_{k-h+1}, \epsilon'_{k-h}, \epsilon_{k-h-1}, \dots), \\
\boldsymbol{\epsilon}^{*}_{k,h} &= (\epsilon_k, \dots, \epsilon_{k-h+1}, \epsilon'_{k-h}, \epsilon'_{k-h-1}, \dots), \\
\boldsymbol{\epsilon}^{*,(t)}_{k,h} &= (\epsilon_k, \dots, \epsilon_{k-h+1}, \epsilon^{(t)}_{k-h}, \epsilon^{(t)}_{k-h-1}, \dots).
\end{aligned}$$

We see that $\boldsymbol{\epsilon}'_{k,h}$ replaces only a single ϵ_h by an independent copy whereas $\boldsymbol{\epsilon}^{*}_{k,h}$ replaces all random seeds more than h lags in the past by independent copies. Finally, $\boldsymbol{\epsilon}^{*,(t)}_{k,h}$ denotes the instance in which all random seeds more than h lags in the past have been replaced by the unique independent copies assigned to Z_t. In this notation, we would write (3.1) as $Z_t = g(\boldsymbol{\epsilon}_t)$ and (3.2) as $Z_t^{(m)} = g(\boldsymbol{\epsilon}^{*,(t)}_{t,m})$.

Definition 3.3. Let $p \geq 1$. Let $(X_t)_{t\in\mathbb{Z}}$ be a stochastic process defined as in (3.1) such that $\mathbb{E}(|X_t|^p) < \infty$. Let ϵ'_{t-h} be an independent, identically distributed copy of ϵ_{t-h}. Then we define the physical dependence measure of X_t as

$$\delta_p(h) = \delta_{X_t,p}(h) := \left\| g(\boldsymbol{\epsilon}'_{t,h}) - g(\boldsymbol{\epsilon}_t) \right\|_{L^p} = \mathbb{E}\left(\left| g(\boldsymbol{\epsilon}'_{t,h}) - g(\boldsymbol{\epsilon}_t) \right|^p \right)^{\frac{1}{p}}.$$

Now, setting $\boldsymbol{\epsilon}^{*,(t)}_{k,\infty} := \boldsymbol{\epsilon}_k$, we can express the difference $Z_t^{(m)} - Z_t$ as telescoping sum. Then

$$\begin{aligned}
\left\| Z_t^{(m)} - Z_t \right\|_{L^p} &= \left\| g(\boldsymbol{\epsilon}^{*,(t)}_{t,m}) - g(\boldsymbol{\epsilon}_t) \right\|_{L^p} \\
&= \left\| g(\boldsymbol{\epsilon}^{*,(t)}_{t,m}) - g(\boldsymbol{\epsilon}^{*,(t)}_{t,m+1}) + g(\boldsymbol{\epsilon}^{*,(t)}_{t,m+1}) - g(\boldsymbol{\epsilon}_t) \right\|_{L^p} \\
&= \left\| \sum_{l=0}^{\infty} g(\boldsymbol{\epsilon}^{*,(t)}_{t,m+k}) - g(\boldsymbol{\epsilon}^{*,(t)}_{t,m+k+1}) \right\|_{L^p} \\
&\leq \sum_{l=0}^{\infty} \left\| g(\boldsymbol{\epsilon}^{*,(t)}_{t,m+l}) - g(\boldsymbol{\epsilon}^{*,(t)}_{t,m+l+1}) \right\|_{L^p} = \sum_{l=m}^{\infty} \delta_{Z_t,p}(l) =: \Delta_p(m)
\end{aligned}$$

In essence, $\Delta_p(m)$ here represents an upper bound on the approximation error we introduce by creating the m-dependent process $(Z_t^{(m)})_{t\geq 1}$. By truncating the dependencies beyond lag m, we are effectively ignoring

the influence of all random shocks from $\epsilon_{t-m}, \epsilon_{t-m-1}, \dots$ and so on. $\Delta_p(m)$ provides a measure of the total effect of this simplification, capturing the cumulative dependence of Z_t on its entire distant history beyond a certain point. A small $\Delta_p(m)$ would imply that the approximation $Z_t^{(m)}$ is close to the original process Z_t, as the influence of the distant past is negligible.

We have now constructed an m-dependent, identical in distribution sequence of errors and can control the approximation errors by imposing some decaying assumption on the physical dependence measure. The next step necessarily is to take a detour and investigate what effects these approximations have on the vector of estimators produced by Algorithm 1.

3.2 Stability of Algorithm 1

In this section, we will investigate the stability of algorithm 1. That is, we will prove that if we use two vectors of data which are in some sense 'close' to each other, then so must the vectors of predictions obtained also be 'close'.

Let us drop any assumptions on the framework of our data and instead let $y_{1:n} = (y_1, \dots, y_n)$ be arbitrary data. Now denote by $y^*_{1:n} = (y^*_1, \dots, y^*_n)$ the vector of independent copies of our data. We can formalise this idea of data which is 'close' to our original data by assuming that there exists some finite $D_n > 0$ such that

$$\|y_{1:n} - y^*_{1:n}\|_\infty := \sup_{1 \le i \le n} |y_i - y^*_i| < D_n.$$

Note that D_n necessarily increases by some rate proportional to n as n increases.

Denote now by $\widehat{\mu}_{1:n}(y_{1:n}) = (\widehat{\mu}_1(y_{1:n}), \dots, \widehat{\mu}_n(y_{1:n}))$ the vector of weighted estimators produced by Algorithm 1 at each round with respect to the input data $y_{1:n}$. For our algorithm to be considered stable, we want to show that the difference between $\widehat{\mu}_{1:n}(y_{1:n})$ and $\widehat{\mu}_{1:n}(y^*_{1:n})$ may be bounded by the difference between $y_{1:n}$ and $y^*_{1:n}$. In this way, as long as the data is sufficiently similar, so too must the predictions produced by Algorithm 1 be.

The following theorem is the main result of this section.

Theorem 3.2. There exists some $L > 0$ depending on $n^2 \log n$ such that

$$\|\widehat{\mu}_{1:n}(y_{1:n}) - \widehat{\mu}_{1:n}(y^*_{1:n})\|_\infty \le L \|y_{1:n} - y^*_{1:n}\|_\infty$$

where $L = 8\alpha n(1 + n \log n) \max\{\|y_{1:n}\|_\infty, \|y_{1:n}\|_\infty\}(\|y_{1:n}\|_\infty + \|y^*_{1:n}\|_\infty) + 1$.

To prove this statement, we will first need a lemma on the expert weights. It can be shown that the difference between the normalised weights belonging to Algorithm 1 with respect to either $y_{1:n}$ or $y^*_{1:n}$, as defined in (2.3), can be bounded in the 1-norm. For this, we denote the vector of normalised weights used for prediction in (2.4) by $\widehat{v}_t(y_{1:n}) := (\widehat{v}_t^{(1)}(y_{1:n}), \dots, \widehat{v}_t^{(t-1)}(y_{1:n}))$. Then also we have analogously $v_t(y_{1:n}) := (v_t^{(1)}(y_{1:n}), \dots, v_t^{(t-1)}(y_{1:n}))$.

Lemma 3.3. For all $t = 1, \dots, n$ we have

$$\|\widehat{v}_t(y_{1:n}) - \widehat{v}_t(y^*_{1:n})\|_1 \le 8\alpha t(1 + \log t)\|y_{1:n} - y^*_{1:n}\|_\infty (\|y_{1:n}\|_\infty + \|y^*_{1:n}\|_\infty)$$

Proof. Note that

$$\begin{aligned}
v_t^{(i)}(y_{1:n}) &= v_{t-1}^{(i)}(y_{1:n}) \exp\left(-\alpha f_{t-1}(\widehat{\mu}_{t-1}^{(i)}(y_{1:n}))\right) \\
&= v_{t-2}^{(i)}(y_{1:n}) \exp\left(-\alpha f_{t-2}(\widehat{\mu}_{t-2}^{(i)}(y_{1:n}))\right) \exp\left(-\alpha f_{t-1}(\widehat{\mu}_{t-1}^{(i)}(y_{1:n}))\right) = \dots \\
&= v_{i+1}^{(i)}(y_{1:n}) \exp\left(-\alpha \sum_{k=i+1}^{t-1} f_k(\widehat{\mu}_k^{(i)}(y_{1:n}))\right) = v_i^{(i)}(y_{1:n}) \exp\left(-\alpha \sum_{k=i+1}^{t-1} f_k(\widehat{\mu}_k^{(i)}(y_{1:n}))\right) \\
&= \frac{1}{i} \exp\left(-\alpha \sum_{k=i+1}^{t-1} f_k(\widehat{\mu}_k^{(i)}(y_{1:n}))\right).
\end{aligned}$$

Hence

$$\left|v_t^{(i)}(y_{1:n}) - v_t^{(i)}(y^*_{1:n})\right| = \frac{1}{i}\left|\exp\left(-\alpha\sum_{k=i+1}^{t-1} f_k(\widehat{\mu}_k^{(i)}(y_{1:n}))\right) - \exp\left(-\alpha\sum_{k=i+1}^{t-1} f_k(\widehat{\mu}_k^{(i)}(y^*_{1:n}))\right)\right|.$$

Let $h(x) = e^{-\alpha x}$ and suppose without loss of generality that $a := \sum_{k=i+1}^{t-1} f_k(\widehat{\mu}_k^{(i)}(y_{1:n})) \le \sum_{k=i+1}^{t-1} f_k(\widehat{\mu}_k^{(i)}(y^*_{1:n})) := b$. Then by the mean value theorem, there exists some $\zeta \in [a, b]$ such that

$$|h(a) - h(b)| = \left|h'(\zeta)\,(b-a)\right| = |\alpha h(\zeta)\,(b-a)| \le \alpha|b-a|$$

where the last inequality follows from the positiveness of a and b, and thus $\zeta \ge 0$. This in turn implies $h(\zeta) \le 1$, since $\alpha > 0$. Filling in, we can write

$$\begin{aligned}\left|v_t^{(i)}(y_{1:n}) - v_t^{(i)}(y^*_{1:n})\right| &\le \frac{\alpha}{i}\left|\sum_{k=i+1}^{t-1} f_k(\widehat{\mu}_k^{(i)}(y^*_{1:n})) - \sum_{k=i+1}^{t-1} f_k(\widehat{\mu}_k^{(i)}(y_{1:n}))\right| \\ &= \frac{\alpha}{i}\sum_{k=i+1}^{t-1}\left|\left(y_k - \frac{1}{k-(i+1)}\sum_{s=i+1}^{k-1} y_s\right)^2 - \left(y^*_k - \frac{1}{k-(i+1)}\sum_{s=i+1}^{k-1} y^*_s\right)^2\right|.\end{aligned}$$

Set $a_k := y_k - \frac{1}{k-(i+1)}\sum_{s=i+1}^{k-1} y_s$ and $b_k := y^*_k - \frac{1}{k-(i+1)}\sum_{s=i+1}^{k-1} y^*_s$, then we can write $\left|a_k^2 - b_k^2\right| = |a_k - b_k||a_k + b_k|$ where

$$\begin{aligned}|a_k - b_k| &= \left|y_k - \frac{1}{k-(i+1)}\sum_{s=i+1}^{k-1} y_s - \left(y^*_k - \frac{1}{k-(i+1)}\sum_{s=i+1}^{k-1} y^*_s\right)\right| \\ &= \left|y_k - y^*_k - \frac{1}{k-(i+1)}\sum_{s=i+1}^{k-1} y_s - y^*_s\right| \le 2\|y_{1:n} - y^*_{1:n}\|_\infty.\end{aligned}$$

and also $|a_k + b_k| \le |a_k| + |b_k| \le 2\|y_{1:n}\|_\infty + 2\|y^*_{1:n}\|_\infty$. Filling back in, we see that

$$\begin{aligned}\left|v_t^{(i)}(y_{1:n}) - v_t^{(i)}(y^*_{1:n})\right| &\le 4\alpha\frac{t-(i+1)}{i}\|y_{1:n} - y^*_{1:n}\|_\infty(\|y_{1:n}\|_\infty + \|y^*_{1:n}\|_\infty) \\ &\le 4\alpha\frac{t-i}{i}\|y_{1:n} - y^*_{1:n}\|_\infty(\|y_{1:n}\|_\infty + \|y^*_{1:n}\|_\infty)\end{aligned}$$

Taking the l_1 norm

$$\begin{aligned}\|v_t(y_{1:n}) - v_t(y^*_{1:n})\|_1 &\le \sum_{i=1}^{t-1} 4\alpha\frac{t-i}{i}\|y_{1:n} - y^*_{1:n}\|_\infty(\|y_{1:n}\|_\infty + \|y^*_{1:n}\|_\infty) \\ &= 4\alpha\|y_{1:n} - y^*_{1:n}\|_\infty(\|y_{1:n}\|_\infty + \|y^*_{1:n}\|_\infty)\sum_{i=1}^{t-1}\frac{t-i}{i}\end{aligned}$$

and by a similar integral trick as in the proof of Theorem 2.2, we see that $\sum_{i=1}^{t-1}\frac{t-i}{i} \le 1 + t\log t$. Hence

$$\|v_t(y_{1:n}) - v_t(y^*_{1:n})\|_1 \le 4\alpha(1 + t\log t)\|y_{1:n} - y^*_{1:n}\|_\infty(\|y_{1:n}\|_\infty + \|y^*_{1:n}\|_\infty).$$

Note that this is a bound on the unnormalised weights and we now handle the normalisation. We see that

$$\begin{aligned}
\left|\widehat{v}_t^{(i)}(y_{1:n}) - \widehat{v}_t^{(i)}(y^*_{1:n})\right| &= \left|\frac{v_t^{(i)}(y_{1:n})}{\sum_{j=1}^{t-1} v_t^{(j)}(y_{1:n})} - \frac{v_t^{(i)}(y^*_{1:n})}{\sum_{j=1}^{t-1} v_t^{(j)}(y^*_{1:n})}\right| \\
&= \left|\frac{v_t^{(i)}(y_{1:n})}{\|v_t(y_{1:n})\|_1} - \frac{v_t^{(i)}(y^*_{1:n})}{\|v_t(y^*_{1:n})\|_1}\right| \\
&\leq \left|\frac{v_t^{(i)}(y_{1:n}) - v_t^{(i)}(y^*_{1:n})}{\|v_t(y_{1:n})\|_1} + v_t^{(i)}(y^*_{1:n})\left(\frac{1}{\|v_t(y_{1:n})\|_1} - \frac{1}{\|v_t(y^*_{1:n})\|_1}\right)\right| \\
&\leq \left|\frac{v_t^{(i)}(y_{1:n}) - v_t^{(i)}(y^*_{1:n})}{\|v_t(y_{1:n})\|_1} + v_t^{(i)}(y^*_{1:n})\left(\frac{\|v_t(y^*_{1:n})\|_1 - \|v_t(y_{1:n})\|_1}{\|v_t(y_{1:n})\|_1\|v_t(y^*_{1:n})\|_1}\right)\right|.
\end{aligned}$$

Taking the 1 norm of the inequality and using that $\|v_t(y_{1:n})\|_1 \geq \frac{1}{t}$, we see by the reverse triangle inequality that

$$\begin{aligned}
\|\widehat{v}_t(y_{1:n}) - \widehat{v}_t(y^*_{1:n})\|_1 &\leq \left\|\frac{v_t(y_{1:n}) - v_t(y^*_{1:n})}{\|v_t(y_{1:n})\|_1} + v_t(y^*_{1:n})\left(\frac{\|v_t(y^*_{1:n})\|_1 - \|v_t(y_{1:n})\|_1}{\|v_t(y_{1:n})\|_1\|v_t(y^*_{1:n})\|_1}\right)\right\|_1 \\
&\leq \frac{\|v_t(y_{1:n}) - v_t(y^*_{1:n})\|_1}{\|v_t(y_{1:n})\|_1} + \frac{\|v_t(y_{1:n}) - v_t(y^*_{1:n})\|_1}{\|v_t(y_{1:n})\|_1} \\
&\leq 2t\|v_t(y_{1:n}) - v_t(y^*_{1:n})\|_1 \\
&\leq 8\alpha t(1 + t\log t)\|y_{1:n} - y^*_{1:n}\|_\infty(\|y_{1:n}\|_\infty + \|y^*_{1:n}\|_\infty).
\end{aligned}$$

□

Proof of Theorem 3.2. Assume without loss of generality that $\|y_{1:n}\|_\infty \leq \|y^*_{1:n}\|_\infty$. For any $t = 1, \ldots, n$

$$\begin{aligned}
|\widehat{\mu}_t(y_{1:n}) - \widehat{\mu}_t(y^*_{1:n})| &= \left|\sum_{i=1}^{t-1} \widehat{v}_t^{(i)}(y_{1:n})\widehat{\mu}_t^{(i)}(y_{1:n}) - \widehat{v}_t^{(i)}(y^*_{1:n})\widehat{\mu}_t^{(i)}(y^*_{1:n})\right| \\
&= \left|\sum_{i=1}^{t-1} (\widehat{v}_t^{(i)}(y_{1:n}) - \widehat{v}_t^{(i)}(y^*_{1:n}))\widehat{\mu}_t^{(i)}(y_{1:n}) + \widehat{v}_t^{(i)}(y^*_{1:n})(\widehat{\mu}_t^{(i)}(y_{1:n}) - \widehat{\mu}_t^{(i)}(y^*_{1:n}))\right|.
\end{aligned}$$

Since $\widehat{\mu}_t^{(i)}$ denotes an averaging scheme, we see that

$$\begin{aligned}
\left|\widehat{\mu}_t^{(i)}(y_{1:n}) + \widehat{\mu}_t^{(i)}(y^*_{1:n})\right| &= \left|\frac{1}{t - t_i}\sum_{k=t_i+1}^{t} y_k - y^*_k\right| \leq \frac{1}{t - t_i}\sum_{k=t_i+1}^{t} |y_k - y^*_k| \\
&\leq \frac{1}{t - t_i}\sum_{k=t_i+1}^{t} \|y_{1:n} - y^*_{1:n}\|_\infty = \|y_{1:n} - y^*_{1:n}\|_\infty.
\end{aligned}$$

Then, because $\widehat{v}_t^{(i)}$ are normalised weights, clearly $\sum_{i=1}^{t-1} |\widehat{v}_t^{(i)}(y^*_{1:n})| = 1$. We also see that

$$\left|\widehat{\mu}_t^{(i)}(y_{1:n})\right| = \left|\frac{1}{t - t_i}\sum_{k=t_i}^{t-1} y_k\right| \leq \frac{1}{t - t_i}\sum_{k=t_i}^{t-1} |y_k| \leq \frac{1}{t - t_i}\sum_{k=t_i}^{t-1} \|y_{1:n}\|_\infty = \|y_{1:n}\|_\infty.$$

Hence, using the triangle inequality

$$\begin{aligned}
|\widehat{\mu}_t(y_{1:n}) - \widehat{\mu}_t(y^*_{1:n})| &\leq \left|\sum_{i=1}^{t-1}(\widehat{v}_t^{(i)}(y_{1:n}) - \widehat{v}_t^{(i)}(y^*_{1:n}))\widehat{\mu}_t^{(i)}(y_{1:n}) + \widehat{v}_t^{(i)}(y^*_{1:n})(\widehat{\mu}_t^{(i)}(y_{1:n}) + \widehat{\mu}_t^{(i)}(y^*_{1:n}))\right| \\
&\leq \sum_{i=1}^{t-1}\left|\widehat{v}_t^{(i)}(y_{1:n}) - \widehat{v}_t^{(i)}(y^*_{1:n})\right|\left|\widehat{\mu}_t^{(i)}(y_{1:n})\right| + \left|\widehat{v}_t^{(i)}(y^*_{1:n})\right|\left|\widehat{\mu}_t^{(i)}(y_{1:n}) + \widehat{\mu}_t^{(i)}(y^*_{1:n})\right| \\
&\leq \|y_{1:n}\|_\infty \sum_{i=1}^{t-1}\left|\widehat{v}_t^{(i)}(y_{1:n}) - \widehat{v}_t^{(i)}(y^*_{1:n})\right| + \|y_{1:n} - y^*_{1:n}\|_\infty \\
&\leq 8\alpha t(1 + t\log t)\|y_{1:n} - y^*_{1:n}\|_\infty \|y_{1:n}\|_\infty(\|y_{1:n}\|_\infty + \|y^*_{1:n}\|_\infty) + \|y_{1:n} - y^*_{1:n}\|_\infty
\end{aligned}$$

where the last inequality follows from Lemma 3.3. Finally

$$\|\widehat{\mu}_{1:n}(y_{1:n}) - \widehat{\mu}_{1:n}(y^*_{1:n})\|_\infty \leq 8\alpha n(1 + n\log n)\|y_{1:n} - y^*_{1:n}\|_\infty \|y_{1:n}\|_\infty(\|y_{1:n}\|_\infty + \|y^*_{1:n}\|_\infty) + \|y_{1:n} - y^*_{1:n}\|_\infty. \tag{3.3}$$

Setting $L = 8\alpha n(1 + n\log n)\|y_{1:n}\|_\infty(\|y_{1:n}\|_\infty + \|y^*_{1:n}\|) + 1$ completes the proof. □

3.3 Performance Guarantees with Dependent Errors

Having established the stability of the FLH algorithm which ensures that small perturbations in the input data lead to correspondingly small changes in the output predictions we now return to our model framework. Then, because the errors are the only source of randomness, we may define data which is close to y_t and m-dependent, in the m-dependence notation, by

$$y_t^{(m)} := \theta_t + Z_t^{(m)}$$

where $Z_t^{(m)}$ is defined as in (3.2).

Then, using this notation, we may adapt the results of Theorem 3.2 to our data model.

Corollary 3.4. For $1 \leq t \leq n$, define $y_t = \theta_t + Z_t$ where $|\theta_t| \leq B$ for some $B > 0$ and Z_t are σ-subgaussian random variables. Let $0 < \delta < 1$. Then

$$\left\|\widehat{\mu}_{1:n}(y_{1:n}) - \widehat{\mu}_{1:n}(y^{(m)}_{1:n})\right\|_\infty \leq O_{\mathbb{P}}(n^{2+\frac{1}{p}}\log n\Delta_p(m))$$

with probability $1 - \delta$.

Proof. Recall that Lemma 2.5 implies that both $\|y_{1:n}\|_\infty$ and $\left\|y^{(m)}_{1:m}\right\|_\infty$ are bounded by $C\sqrt{\log\frac{2n}{\delta}}$ with probability $1 - \delta$ for some $0 < \delta < 1$. Recall also that we have chosen $\alpha = \frac{1}{8C^2\log\frac{2n}{\delta}}$ in Lemma 2.3 so that the mapping $y \mapsto e^{-\alpha(y-y_t)^2}$ is α-exp-concave with probability $1 - \delta$. Then (3.3) can be rewritten as

$$\begin{aligned}
\left\|\widehat{\mu}_{1:n}(y_{1:n}) - \widehat{\mu}_{1:n}(y^{(m)}_{1:n})\right\|_\infty &\leq 8\alpha n(1 + n\log n)\left\|y_{1:n} - y^{(m)}_{1:n}\right\|_\infty \|y_{1:n}\|_\infty(\|y_{1:n}\|_\infty + \left\|y^{(m)}_{1:n}\right\|_\infty) + \left\|y_{1:n} - y^{(m)}_{1:n}\right\|_\infty \\
&\leq \frac{1}{C^2\log\frac{2n}{\delta}}n(1 + n\log n)2C^2\log\frac{2n}{\delta}\left\|y_{1:n} - y^{(m)}_{1:n}\right\|_\infty + \left\|y_{1:n} - y^{(m)}_{1:n}\right\|_\infty \\
&= 2n(1 + n\log n)\left\|y_{1:n} - y^{(m)}_{1:n}\right\|_\infty + \left\|y_{1:n} - y^{(m)}_{1:n}\right\|_\infty \\
&\leq \mathcal{O}(n^2\log n\left\|y_{1:n} - y^{(m)}_{1:n}\right\|_\infty).
\end{aligned}$$

We have by Markov's inequality that

$$\mathbb{P}\left(\max_{1\le t\le n}|y_t|>K\right)\le\sum_{t=1}^{n}\mathbb{P}\left(|y_t|>K\right)\le\sum_{t=1}^{n}\frac{\|y_t\|_{L^p}^p}{K^p}\le\frac{n}{K^p}\max_{1\le t\le n}\|y_t\|_{L^p}^p.$$

Hence $\max_{1\le t\le n}\left\|y_t-y_t^{(m)}\right\|_{L^p}=\max_{1\le t\le n}\left\|Z_t-Z_t^{(m)}\right\|_{L^p}\le\Delta_p(m)$ implies

$$n^2\log n\left\|y_t-y^{(m)}\right\|_{\infty}\le\mathcal{O}_{\mathbb{P}}(n^{2+\frac{1}{p}}\log n\Delta_p(m)) \tag{3.4}$$

which completes the proof. □

In order to exploit the m-dependence structure we have just created, we first need to partition the data into m rows $R_1,\dots,R_m$ like so:

$$\begin{array}{c|ccccccccccc}
R_1 & y_1 & & & & y_{m+1} & & & & \cdots & y_{lm+1} & \\
R_2 & & y_2 & & & & y_{m+2} & & & \cdots & & \ddots \\
\vdots & & & \ddots & & & & \ddots & & \cdots & & & y_n \\
R_m & & & & y_m & & & & y_{2m} & \cdots & &
\end{array}$$

The idea is that we may run multiple parallel instances of FLH on each, now independent, row of data. Let us formalise this.

Denote by $(\frac{n}{m})_i:=\lfloor\frac{n}{m}\rfloor+\mathbb{1}_{\{\lfloor\frac{n}{m}\rfloor+i\le n\}}$, then each row can be concisely defined as $R_i:=(y_{km+i})_{k=0}^{(\frac{n}{m})_i}$ and we may exploit the m-dependence structure to create

$$R_i^{(m)}:=\left(y_{km+i}^{(m)}\right)_{k=0}^{(\frac{n}{m})_i}$$

where each $R_i^{(m)}$ is an independent row of data.

We want to use $R_1^{(m)},\dots,R_m^{(m)}$ for predictions. A naive approach is to simply use the 'most recent independently observed' row for prediction. That is, we predict $\widehat{\mu}_t$ using Algorithm 1 with R_i as input data where i is chosen so that $y_{t-m}\in R_i$. This allows us to exploit the independence properties of $R_1^{(m)},\dots,R_m^{(m)}$ and that y_t is independent of $y_{t-m}^{(m)}$. Let us formalise this. We can define $\widehat{\mu}_{1:n}(R_{1:m})$ element wise by

$$\widehat{\mu}_t(R_{1:m}):=\widehat{\mu}_j(R_i)$$

for all $t=1,\dots,n$ where j and i are the unique $j\in\mathbb{Z}$ and $i\in\{1,\dots,m\}$ such that $t=jm+i$.

Algorithm 2 Thinned-Follow-the-Leading-History (TFLH)

Input: Black box algorithm $\mathcal{A}$, learning parameter $\alpha>0$, number of bins m

1: **Init:** m independent instances of the FLH algorithm, $\mathrm{FLH}_1,\dots,\mathrm{FLH}_m$, each with learning parameter α and black box algorithm $\mathcal{A}$
2: **for** $t=1,\dots,n$ **do**
3: Determine the active bin index $j=(t-1)\pmod m$
4: Obtain prediction $\widehat{\mu}_t$ from instance FLH_j
5: Instance FLH_j observes y_t

The draw-back to Algorithm 2 is that we are in effect thinning our data, and thus have less data on which to make our predictions. For this reason, we should expect to see some sort of difference in performance between this scheme and the independent case, however we see in Theorem 3.5 that this difference is quite small and does not stop us from achieving the optimal cumulative error rate.

We are now ready to prove the main result of this work. The core idea of the proof will be to use a similar argument as in the proof of Theorem 2.8, except this time the forecaster against which we compares ourselves is the one obtained under the independent case regime.

Theorem 3.5. Let $n, m \geq 1$, $\sigma > 0$, and $C_n > 0$. Let $\theta_1, \ldots, \theta_n \in \mathbb{R}$ be any sequence such that $TV(\theta_{1:n}) \leq C_n$ and $|\theta_1| \leq B$. For $1 \leq t \leq n$, let $y_t = \theta_t + Z_t$ where Z_t are dependent identically distributed random variables defined according to (3.1) which are furthermore σ-subgaussian. Suppose $\Delta_p(m) = \mathcal{O}(e^{-cm})$ for some $c > 0$ and $p > 2$. If moving average predictions are used as sub-routines of Algorithm 2, then

$$\mathcal{R}_n(\widehat{\mu}_{1:n}(R_{1:m}), \theta_{1:n}) \leq \mathcal{O}(n^{\frac{1}{3}}(\log n)^{2+\frac{2}{3}} C_n^{\frac{2}{3}}).$$

Proof. We can rewrite

$$\begin{aligned}
\mathcal{R}_n(\widehat{\mu}_{1:n}(R_{1:m}), \theta_{1:n}) &= \sum_{t=1}^{n} \mathbb{E}\left((\widehat{\mu}_t(R_{1:m}) - \theta_t)^2\right) \\
&= \sum_{j=1}^{(\frac{n}{m})_i} \sum_{i=1}^{m} \mathbb{E}\left((\widehat{\mu}_j(R_i) - \theta_j)^2\right) \\
&= \sum_{j=1}^{(\frac{n}{m})_i} \sum_{i=1}^{m} \mathbb{E}\left((\widehat{\mu}_j(R_i) - \widehat{\mu}_j(R_i^{(m)}) + \widehat{\mu}_j(R_i^{(m)}) - \theta_j)^2\right) \\
&\quad \sum_{j=1}^{(\frac{n}{m})_i} \sum_{i=1}^{m} \mathbb{E}\left((\widehat{\mu}_j(R_i) - \widehat{\mu}_j(R_i^{(m)}) + \widehat{\mu}_j(R_i^{(m)}) - \theta_j)^2\right) \\
&= \sum_{j=1}^{(\frac{n}{m})_i} \sum_{i=1}^{m} \mathbb{E}\left((\widehat{\mu}_j(R_i) - \widehat{\mu}_j(R_i^{(m)}))^2\right) + \mathbb{E}\left((\widehat{\mu}_j(R_i^{(m)}) - \theta_j)^2\right) \\
&\quad + 2\mathbb{E}\left((\widehat{\mu}_j(R_i) - \widehat{\mu}_j(R_i^{(m)}))(\widehat{\mu}_j(R_i^{(m)}) - \theta_j)\right).
\end{aligned}$$

We can handle the first term by Theorem 3.2, the second term by Theorem 2.8, and the cross term by Cauchy-Schwartz like so

$$\begin{aligned}
\mathcal{R}_n(\widehat{\mu}_{1:n}(R_{1:m}), \theta_{1:n}) &= \sum_{j=1}^{(\frac{n}{m})_i} \sum_{i=1}^{m} \mathbb{E}\left((\widehat{\mu}_j(R_i) - \widehat{\mu}_j(R_i^{(m)}))^2\right) + \mathbb{E}\left((\widehat{\mu}_j(R_i^{(m)}) - \theta_j)^2\right) \\
&\quad + 2\mathbb{E}\left((\widehat{\mu}_j(R_i) - \widehat{\mu}_j(R_i^{(m)}))(\widehat{\mu}_j(R_i^{(m)}) - \theta_j)\right) \\
&\leq \mathcal{O}_{\mathbb{P}}\left(\left(\frac{n}{m}\right)^{2+\frac{1}{p}} \log n \Delta_p(m)\right) + \mathcal{O}\left(\left(\frac{n}{m}\right)^{\frac{1}{3}} m C_n^{\frac{2}{3}} (\log n)^2\right) \\
&\quad + 2\sum_{j=1}^{(\frac{n}{m})_i} \sum_{i=1}^{m} \sqrt{\mathbb{E}\left((\widehat{\mu}_j(R_i) - \widehat{\mu}_j(R_i^{(m)}))^2\right) \mathbb{E}\left((\widehat{\mu}_j(R_i^{(m)}) - \theta_j)^2\right)} \\
&\leq \mathcal{O}_{\mathbb{P}}\left(\left(\frac{n}{m}\right)^{2+\frac{1}{p}} \log n \Delta_p(m)\right) + \mathcal{O}\left(\left(\frac{n}{m}\right)^{\frac{1}{3}} m C_n^{\frac{2}{3}} (\log n)^2\right). \\
&\quad + 2\sqrt{\sum_{j=1}^{(\frac{n}{m})_i} \sum_{i=1}^{m} \mathbb{E}\left((\widehat{\mu}_j(R_i) - \widehat{\mu}_j(R_i^{(m)}))^2\right)} \sqrt{\sum_{j=1}^{(\frac{n}{m})_i} \sum_{i=1}^{m} \mathbb{E}\left((\widehat{\mu}_j(R_i^{(m)}) - \theta_j)^2\right)} \\
&\leq \mathcal{O}_{\mathbb{P}}\left(\left(\frac{n}{m}\right)^{2+\frac{1}{p}} \log n \Delta_p(m)\right) + \mathcal{O}\left(\left(\frac{n}{m}\right)^{\frac{1}{3}} m C_n^{\frac{2}{3}} (\log n)^2\right) \\
&\quad + 2\max\left\{\sum_{j=1}^{(\frac{n}{m})_i} \sum_{i=1}^{m} \mathbb{E}\left((\widehat{\mu}_j(R_i) - \widehat{\mu}_j(R_i^{(m)}))^2\right), \sum_{j=1}^{(\frac{n}{m})_i} \sum_{i=1}^{m} \mathbb{E}\left((\widehat{\mu}_j(R_i^{(m)}) - \theta_j)^2\right)\right\}.
\end{aligned}$$

Then, we obtain

$$\mathcal{R}_n(\widehat{\mu}_{1:n}(R_{1:m}), \theta_{1:n}) \leq \mathcal{O}_{\mathbb{P}}\left(\left(\frac{n}{m}\right)^{2+\frac{1}{p}} \log n \Delta_p(m)\right) + \mathcal{O}\left(n^{\frac{1}{3}} m^{\frac{2}{3}} C_n^{\frac{2}{3}} (\log n)^2\right).$$

Finally, by the assumption on $\Delta_p(m)$, we obtain

$$\mathcal{R}_n(\widehat{\mu}_{1:n}(R_{1:m}), \theta_{1:n}) \leq \mathcal{O}_{\mathbb{P}}\left(\left(\frac{n}{m}\right)^{2+\frac{1}{p}} e^{-cm} \log n\right) + \mathcal{O}\left(n^{\frac{1}{3}} m^{\frac{2}{3}} C_n^{\frac{2}{3}} (\log n)^2\right).$$

It remains to optimise this bound with respect to m. We can do this by solving

$$\left(\frac{n}{m}\right)^{2+\frac{1}{p}} e^{-cm} \log n = \left(\frac{n}{m}\right)^{\frac{1}{3}} m C_n^{\frac{2}{3}} (\log n)^2 \tag{3.5}$$

with respect to m.

The final bound consists of two terms: an approximation error from using m-dependent data, which decreases exponentially with m, and a statistical error from data thinning, which increases polynomially with m. To optimize the bound, we must choose m to balance these competing terms. Let us try to express m as the ansatz $m := b \log n$ for some $b > 0$ and consider how (3.5) behaves asymptomatically. Filling this ansatz in, (3.5) can be reduced to

$$n^{2+\frac{1}{p}-cb} (\log n)^{-1-\frac{1}{p}} = n^{\frac{1}{3}} (\log n)^{2+\frac{2}{3}} C_n^{\frac{2}{3}}$$

where we omit the constant terms. For $b > \frac{2+\frac{1}{p}}{c}$, the left hand side vanishes as $n \to \infty$. Conversely, for all choices $b < \frac{1}{pc}$, the left hand side grows faster than $n^2 (\log n)^{-1-\frac{1}{p}}$. However, notice that the asymptotic order of magnitude of the right hand side is independent of b. Then for $m = b \log n$, where $b > \frac{2+\frac{1}{p}}{c}$, we find that

$$\mathcal{R}_n(\widehat{\mu}_{1:n}(R_{1:m}), \theta_{1:n}) \leq \mathcal{O}\left(n^{\frac{1}{3}} (\log n)^{2+\frac{2}{3}} C_n^{\frac{2}{3}}\right).$$

□

Chapter 4

Simulation Study

In this chapter we evaluate the results of Chapter 2 and Chapter 3 empirically in a simulation study. To do this, we compare the theoretical bounds against our algorithm and ARROWS [2].

Before we begin however, we need to address a key limitation of FLH with respect to its formulation in Algorithm 1. It may be shown that if the base black box algorithm used in FLH has a running time of V per round, the total running time for FLH is then $\mathcal{O}(Vn)$ [1]. This linear growth in complexity stems from the need to keep track of so many experts and makes large-scale simulations computationally infeasible. To overcome this limitation, there exists a more computationally efficient variant, the Improved Follow-the-Leading-History (IFLH) algorithm, which runs in $\mathcal{O}(V \log n)$ time [1]. However, preliminary simulations showed that the run time of the standard FLH algorithm was manageable for the time horizons considered in our study. Therefore, we opted for this implementation to align as closely as possible with the theoretical analysis in the preceding chapters, leaving the practical implementation of TFLH with an IFLH backbone as a direction for future work. See Appendix B for more discussion on the IFLH algorithm.

Consider also that there exists a computationally efficient implementation of moving averages with a running time of $\mathcal{O}(1)$. To see this notice that each expert $\mathcal{A}_t$ need only keep track of 2 values: a running sum of the observations it has seen and a count of how many observations it has seen. Then at each new time step t, the expert performs two simple operations: it adds the latest observation to its running sum and increments its count. The prediction is then the sum divided by the count. This update process takes constant time, regardless of how many data points the expert has already processed.

4.1 ARROWS

We now introduce the algorithm against which we will benchmark FLH. The policy Adaptive Restarting Rule for Online averaging using Wavelet Shrinkage (ARROWS), a slightly modified version of the soft thresholding estimator introduced in [46], was first proposed by [2] for online forecasting of sequences of length n whose total variation is bounded. In their paper, the authors showed that the ARROWS policy achieves the optimal cumulative error bound $\tilde{\mathcal{O}}(n^{\frac{1}{3}} C_n^{\frac{2}{3}})$ with high probability. We chose ARROWS because it is also designed for online nonparametric regression, achieves the optimal cumulative error bound, and runs in polynomial time.

Recall the oracle forecaster introduced Chapter 2, which relied on perfect information of the restart times for moving average predictions. In FLH, we attempt to learn these restart times indirectly using meta-aggregation principles; however, ARROWS takes a more direct approach. The core principle of ARROWS is to maintain an online average of the observed data. In areas of low total variation, this works perfectly well, however this strategy breaks down when the signal exhibits a sudden change or jump, violating the implicit assumption of local constancy.

ARROWS determines when to perform a restart in an online fashion by employing a statistical test based on the discrete Haar wavelet transform. At each time step t, the algorithm considers the segment of observations in its current averaging window. The observations in the current window are mean-centered and transformed

into the wavelet domain using the Haar basis. The Haar transform is particularly well-suited for this task as its coefficients capture localized differences in the signal, which are directly related to its total variation.

The resulting Haar coefficients, which are contaminated by noise, are then denoised using a soft-thresholding rule. This step, also known as wavelet shrinkage, effectively removes coefficients that are likely to be pure noise while preserving those that represent the true signal. A weighted l_1-norm of the denoised coefficients is computed. This quantity serves as a lower bound for the total variation of the underlying signal within the current window. If this lower-bound on the total variation exceeds a predefined threshold (which depends on the noise level σ), the algorithm concludes that a change point has occurred. It then triggers a restart, discarding the current averaging window and beginning a new one at the next time step.

This non-linear, data-adaptive restarting mechanism allows ARROWS to approximate the oracle forecaster that knows the true change points and enables the algorithm to escape the fundamental limitations of linear forecasters. A significant drawback, however, is that we require knowledge of noise level σ in order to run the algorithm. A solution to this may be to reserve some data to estimate this value, but this is not ideal. We however also require advance knowledge of the time horizon n for the soft-thresholding. For the purposes of this simulation study, we will simply assume that both σ and n are known quantities ahead of time.

To define ARROWS properly we will need to introduce some notation.

- We let t_B denote the start time of the current bin.
- We denote by $\text{pad}_0(y_{t_B}, \dots, y_t)$ the vector $(y_{t_B} - \bar{y}_{t_B:t}, \dots, y_t - y_{t_B:t})^T$ which is zero-padded at the end such that it is in $\mathbb{R}^k$ for $k = 2^l$ where we choose the smallest $l \in \mathbb{Z}$ for which the relation holds.
- We denote by $T : \mathbb{R}^k \to \mathbb{R}^k$ the element-wise thresholding of a vector with threshold $\tau := \sigma\sqrt{\beta \log n}$. Then for $x \in \mathbb{R}^k$
$$T(x) = (\text{sign}(x_1)\max\{|x_1| - \tau, 0\}, \dots, \text{sign}(x_k)\max\{|x_k| - \tau, 0\})^\top.$$
Here $\text{sign} : \mathbb{R} \to \{-1, 0, 1\}$ is the sign function defined by $\text{sign}(z) = (-1)^{\mathbb{1}\{z<0\}}$ for $z \in \mathbb{R} \setminus \{0\}$ and $\text{sign}(0) = 0$.
- Let H denote the orthogonal discrete wavelet transform matrix of proper dimensions.
- Let $Hx = \alpha_{1:k} = (\alpha_1, \alpha_2, \dots, \alpha_k)^\top \in \mathbb{R}^k$ be the vector of Haar coefficients. Then the vector $(\alpha_2, \dots, \alpha_k)^\top$ can be viewed as a concatenation of $\log_2 k$ contiguous blocks represented by $\alpha[l]$, for $l = 0, \dots, \log_2(k) - 1$. Each block $\alpha[l]$ then contains 2^l coefficients.

Algorithm 3 ARROWS [2]

Input: standard deviation $\sigma > 0$, time horizon n, learning parameter $\beta > 24$

Init: $t_B = 1$, $newBin = 1$, $y_0 = 0$.
for $t = 1, \dots, n$ **do**
  if $newBin == 1$ **then**
    Predict $\widehat{\mu}_t^{t_B} = y_{t-1}$
  else
    Predict $\widehat{\mu}_t^{t_B} = \bar{y}_{t_B:(t-1)}$
  Set $newBin = 0$, observe y_t, and suffer loss $f_t(x_t^{t_B})$
  Set $\tilde{y} = \text{pad}_0(y_{t_B}, \dots, y_t) \in \mathbb{R}^k$
  Set $\alpha_{1:k} = T(H\tilde{y})$
  if $\frac{1}{\sqrt{k}} \sum_{l=0}^{\log_2(k)-1} 2^{\frac{l}{2}} \|\alpha_{1:k}[l]\|_1 > \frac{\sigma}{\sqrt{k}}$ **then**
    Set $newBin = 1$
    Set $t_B = t + 1$

As a final remark, we note that there exists a computationally efficient implementation of ARROWS which reduces the time complexity from $\mathcal{O}(n^2)$ to $\mathcal{O}(n \log n)$, see Proposition 10 of [2].

4.2 Data Generation

In this section, we discuss the method by which the synthetic data sets were generated.

4.2.1 Signal

We consider two regimes for signal generation, taking inspiration from [3]. The first is hard shifting. We split the signal $\theta_1, \dots, \theta_n$ into N sections such that n_i is the index of the start of section i. Then the value of θ_t is constant within each chunk and we start with $\theta_0 = 0$. To go between chunks, we take independent samples $\phi_1, \dots, \phi_{N-1} \sim N(0, \rho^2)$ for $\rho > 0$ and set $\theta_{n_i} = \theta_{n_i - 1} + \phi_i$. Then we find that $TV(\theta_{1:n}) = \sum_{t=2}^{n} |\theta_t - \theta_{t-1}| = \sum_{i=2}^{N-1} |\theta_{n_i} - \theta_{n_{i-1}}| = \sum_{i=2}^{N-1} |\phi_i - \phi_{i-1}|$.

The second regime is soft shifting which seeks to emulate a type of random walk where the underlying signal changes continuously. Here we start at $\theta_1 = 0$ and then subsequently set $\theta_t = \theta_{t-1} + \zeta_t$ with $\zeta_t \sim N(0, t^{-\eta})$ for some $\eta > 0$. Then we find that $TV(\theta_{1:n}) = \sum_{t=2}^{n} |\theta_t - \theta_{t-1}| = \sum_{t=2}^{n} |\zeta_t|$.

4.2.2 Errors

Recall the definition of (3.1). By Lemma 3.1, we may take $\epsilon_t \overset{\text{iid}}{\sim} N(0, \sigma^2)$ univariate normal for $t = 1, \dots, n$. These will be our independent errors. Now to obtain dependent errors we set

$$Z_t = \frac{1}{L_t} \sum_{j=1}^{t} \frac{\epsilon_j}{(t - j + 1)^p} \tag{4.1}$$

where $L_t := \sqrt{\sum_{j=1}^{t} \frac{1}{(t-j+1)^{2p}}}$ is a normalising constant so that $\text{Var}(Z_t) = \sigma^2$ for all $t = 1, \dots, n$. The formulation of (4.1) yields identically distributed dependent random errors with the crucial property that random variables which are far apart are 'less' dependent.

4.3 Results

We present in Figure A.1 and Figure A.2 example data sets generated according to the hard and soft-shift regimes respectively. We take $\sigma = 1$, $\rho = 2$, $\eta = 0.9$, $p = 1$, and a time horizon of $n = 1000$. Furthermore, we take $n_1 = 100$ and $n_i = 200(i - 1)$ for all $i \geq 2$. This is only done for cosmetic reasons as the total variation within the first segment is 0, hence the cumulative error bound of Theorem 3.5 would not work in the first segment.

We averaged the cumulative errors suffered over 15 runs by each algorithm, both in the independent and dependent setting, to obtain Figure 4.1 and Figure 4.2.

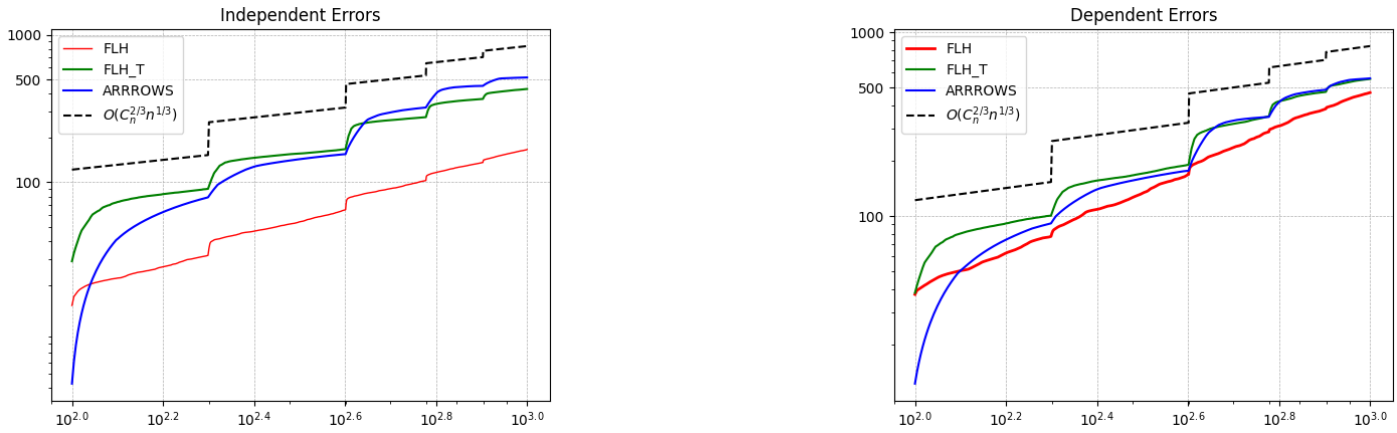

Figure 4.1: Cumulative squared error of FLH, TFLH, and ARROWS together with the upper bound of order $\mathcal{O}(n^{\frac{1}{3}}C_n^{\frac{2}{3}})$ for dependent and independent errors in the hard shifting regime. These results are the averages of 15 identical simulations. The hyperparameters chosen were $\alpha = 0.1$ and $\beta = 24.1$.

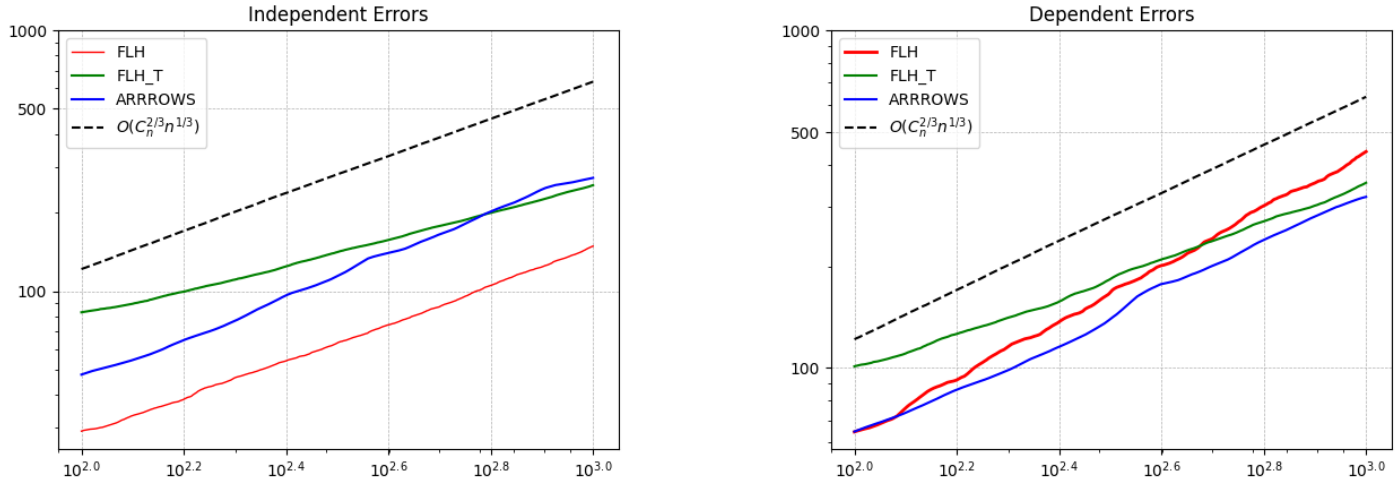

Figure 4.2: Cumulative squared error of FLH, TFLH, and ARROWS together with the upper bound of order $\mathcal{O}(n^{\frac{1}{3}}C_n^{\frac{2}{3}})$ for dependent and independent errors in the soft shifting regime. These results are the averages of 15 identical simulations. The hyperparameters chosen were $\alpha = 0.1$ and $\beta = 24.1$.

In the i.i.d. setting (left), FLH performs exceptionally well, however most importantly, the cumulative error curves of each algorithm grow sub-linearly and closely follow the theoretical $\tilde{\mathcal{O}}(n^{1/3})$ rate, confirming their near-optimality. The performance of TFLH is slightly worse, which is expected because the data thinning approach is unnecessary in the independent case and results in a smaller effective sample size for each expert.

The standard FLH algorithm, however, performs noticeable worse with dependent noise (right) and here we see the benefits of the thinning regime. This drop in performance is exaggerated if we increase the dependencies between errors, for example if we take $p = 0.8$ in (4.1), as seen in Figure 4.3.

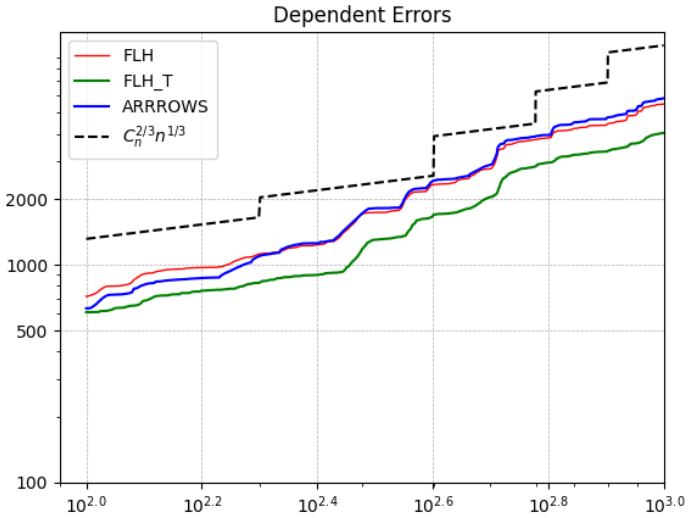

Figure 4.3: Cumulative squared error of FLH, TFLH, and ARROWS together with the upper bound of order $\mathcal{O}(n^{\frac{1}{3}}C_n^{\frac{2}{3}})$ for dependent and independent errors in the hard shifting regime. The hyperparameters chosen were $\alpha = 0.1$ and $\beta = 24.1$.

It is interesting to note, however, that FLH still appears to nonetheless respect the sub-linear cumulative error bound, indicating that there may be another method by which we may replicate the results of Chapter 3.

Chapter 5

Conclusion and Future Work

This thesis addresses a gap in the literature: the presence of temporal dependence in the error structure of online nonparametric regression models. While the problem of sequentially estimating a signal with bounded total variation from noisy observations has been extensively studied, the vast majority of existing literature operates under the assumption of independent noise. This assumption may fail in real-world applications where observations are subject to correlated fluctuations. Our work confronts this gap directly, investigating whether methods that are minimax optimal in the independent setting can be extended to the dependent setting while retaining their performance guarantees.

The central contribution of this work is the development and analysis of the Thinned-Follow-the-Leading-History (TFLH) algorithm, a modification of the Follow-the-Leading-History (FLH) algorithm. We began by reviewing the existing literature for the independent noise setting in Chapter 2 and established that the FLH algorithm indeed achieves the minimax optimal cumulative error rate of $\tilde{\mathcal{O}}(n^{1/3}C_n^{2/3})$. Unfortunately, the proof techniques of Chapter 2 are not directly extendable to the dependent setting when we introduce interdependencies among the Z_t error terms. At a heuristic level, dependencies in the error terms can mislead the weight-updating mechanism of FLH, degrading its performance.

To overcome this, the TFLH algorithm is introduced as an aggregation of many instances of FLH over data that has been split into multiple streams. By partitioning the data stream into $m \approx \log n$ independent sub-streams, we effectively break the short-range correlation structure. The core of our theoretical contribution, presented in Chapter 3, shows that this approach is not merely a heuristic but a principled method that achieves statistical optimality. We demonstrated that under mild assumptions on the decay of long-range dependence, quantified using a physical dependence measure, the TFLH algorithm successfully mitigates the impact of correlated errors. The main theoretical result of this thesis is that TFLH achieves a cumulative error rate of $\tilde{\mathcal{O}}(n^{1/3}C_n^{2/3})$ with high probability, thereby matching the minimax optimal rate for the independent case. These results are furthermore corroborated in the simulation study conducted in Chapter 4, where we show that TFLH may even outperform FLH in high-dependence settings, demonstrating empirically that the statistical error from data thinning may be outweighed by the benefit of mitigating dependence among errors.

Despite these positive results, the findings of this work are only a preliminary step in this field. First, our analysis relied on a specific causal representation of the dependent noise process and assumptions on the decay of its physical dependence measure. Future work could explore the performance of TFLH under different, perhaps more general, dependence frameworks, such as β-mixing conditions as in [22], to broaden the applicability of the method. Additionally, future research could investigate the properties of TFLH for more general function classes such as Hölder or Sobolev spaces, as in [40].

Second, while TFLH does achieve the minimax cumulative error bound, the data thinning scheme necessarily reduces the statistical accuracy of its predictions. There may exist a method to avoid the need for TFLH by first constructing estimators for the covariance matrix of the errors, as in [20, 21], and running the standard FLH algorithm with this additional information. The challenges with this method are twofold. First, how do we iteratively update such a covariance matrix estimate in the online setting? Second, how can this information best be used in an algorithm that is a meta-aggregation of moving averages?

Finally, TFLH achieves optimality for a number of partitions $m \approx \log n$, but the optimal constant scaling in a finite-sample setting was not deeply investigated. The choice of m represents a crucial trade-off between mitigating dependence and retaining statistical power, and a more adaptive, data-driven method for selecting this parameter could significantly enhance the algorithm's practical performance. Furthermore, while the simulation study validated our theoretical claims, it was conducted using the standard FLH algorithm for its sub-routines. The more computationally efficient IFLH, see Appendix B, which would be necessary for truly large-scale problems was not investigated. This highlights a gap between the current theoretical proposal and a fully practical, scalable implementation.

In summary, this thesis has contributed towards a more robust theory of online nonparametric regression. By demonstrating that optimal performance guarantees may be achieved even in the presence of dependent noise, we have shown that the power of adaptive online algorithms is not confined to idealized settings. The proposed TFLH algorithm provides a theoretically and empirically sound method for handling a common feature of real-world data, paving the way for more reliable and effective sequential estimation in non-stationary environments.

Bibliography

[1] Elad Hazan and Comandur Seshadhri. "Adaptive algorithms for online decision problems". In: *Electronic colloquium on computational complexity (ECCC)*. Vol. 14. 088. 2007.

[2] Dheeraj Baby and Yu-Xiang Wang. "Online forecasting of total-variation-bounded sequences". In: *Advances in Neural Information Processing Systems* 32 (2019).

[3] Anant Raj, Pierre Gaillard, and Christophe Saad. "Non-stationary online regression". In: *arXiv preprint arXiv:2011.06957* (2020).

[4] Mikhail Belkin et al. "Reconciling modern machine-learning practice and the classical bias–variance trade-off". In: *Proceedings of the National Academy of Sciences* 116.32 (2019), pp. 15849–15854.

[5] David L Donoho and Iain M Johnstone. "Minimax estimation via wavelet shrinkage". In: *The annals of Statistics* 26.3 (1998), pp. 879–921.

[6] Enno Mammen and Sara Van De Geer. "Locally adaptive regression splines". In: *The Annals of Statistics* 25.1 (1997), pp. 387–413.

[7] Gabriele Steidl, Stephan Didas, and Julia Neumann. "Splines in higher order TV regularization". In: *International journal of computer vision* 70 (2006), pp. 241–255.

[8] Seung-Jean Kim et al. "l_1 trend filtering". In: *SIAM review* 51.2 (2009), pp. 339–360.

[9] Ryan J Tibshirani. "Adaptive piecewise polynomial estimation via trend filtering". In: *The Annals of Statistics* (2014).

[10] Tomoya Wakayama and Shonosuke Sugasawa. "Trend filtering for functional data". In: *Stat* 12.1 (2023), e590.

[11] Veeranjaneyulu Sadhanala and Ryan J Tibshirani. "Additive models with trend filtering". In: *The Annals of Statistics* 47.6 (2019), pp. 3032–3068.

[12] Yu-Xiang Wang et al. "Trend filtering on graphs". In: *Journal of Machine Learning Research* 17.105 (2016), pp. 1–41.

[13] Leonid I Rudin, Stanley Osher, and Emad Fatemi. "Nonlinear total variation based noise removal algorithms". In: *Physica D: nonlinear phenomena* 60.1-4 (1992), pp. 259–268.

[14] Robert Tibshirani et al. "Sparsity and smoothness via the fused lasso". In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 67.1 (2005), pp. 91–108.

[15] Veeranjaneyulu Sadhanala, Yu-Xiang Wang, and Ryan J Tibshirani. "Total variation classes beyond 1d: Minimax rates, and the limitations of linear smoothers". In: *Advances in Neural Information Processing Systems* 29 (2016).

[16] Jan-Christian Hütter and Philippe Rigollet. "Optimal rates for total variation denoising". In: *Conference on Learning Theory*. PMLR. 2016, pp. 1115–1146.

[17] Sabyasachi Chatterjee. "Minmax trend filtering: A locally adaptive nonparametric regression method via pointwise min max optimization". In: *arXiv preprint arXiv:2410.03041* (2024).

[18] Peter Hall and Jeffrey D Hart. "Nonparametric regression with long-range dependence". In: *Stochastic Processes and Their Applications* 36.2 (1990), pp. 339–351.

[19] Yuhong Yang. "Nonparametric regression with dependent errors". In: *Bernoulli Society for Mathematical Statistics and Probability* (2001).

[20] Margherita Gerolimetto, Stefano Magrini, et al. "Nonparametric regression with spatially dependent data". In: *Department of Economics WP* 20 (2009), p. 2009.

[21] Holger Dette and Weichi Wu. "Prediction in locally stationary time series". In: *Journal of Business & Economic Statistics* 40.1 (2022), pp. 370–381.

[22] Abhishek Roy, Krishnakumar Balasubramanian, and Murat A Erdogdu. "On empirical risk minimization with dependent and heavy-tailed data". In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 8913–8926.

[23] Shahar Mendelson. "Learning without concentration". In: *Journal of the ACM (JACM)* 62.3 (2015), pp. 1–25.

[24] Qiyang Han and Jon A Wellner. "Convergence rates of least squares regression estimators with heavy-tailed errors". In: *The Annals of Statistics* 47.4 (2019).

[25] Nick Littlestone. "Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm". In: *Machine learning* 2 (1988), pp. 285–318.

[26] Nick Littlestone and Manfred K Warmuth. "The weighted majority algorithm". In: *Information and computation* 108.2 (1994), pp. 212–261.

[27] David Haussler, Jyrki Kivinen, and Manfred K Warmuth. "Sequential prediction of individual sequences under general loss functions". In: *IEEE Transactions on Information Theory* 44.5 (1998), pp. 1906–1925.

[28] Vladimir G Vovk. "A game of prediction with expert advice". In: *Proceedings of the eighth annual conference on Computational learning theory*. 1995, pp. 51–60.

[29] Mark Herbster and Manfred K Warmuth. "Tracking the best expert". In: *Machine learning* 32.2 (1998), pp. 151–178.

[30] Mark Herbster and Manfred K Warmuth. "Tracking the best linear predictor". In: *Journal of Machine Learning Research* 1.Sep (2001), pp. 281–309.

[31] Olivier Bousquet and Manfred K Warmuth. "Tracking a small set of experts by mixing past posteriors". In: *Journal of Machine Learning Research* 3.Nov (2002), pp. 363–396.

[32] Martin Zinkevich. "Online convex programming and generalized infinitesimal gradient ascent". In: *Proceedings of the 20th international conference on machine learning (icml-03)*. 2003, pp. 928–936.

[33] Jacob Abernethy et al. "Optimal strategies and minimax lower bounds for online convex games". In: *Proceedings of the 21st annual conference on learning theory*. 2008, pp. 414–424.

[34] Omar Besbes, Yonatan Gur, and Assaf Zeevi. "Non-stationary stochastic optimization". In: *Operations research* 63.5 (2015), pp. 1227–1244.

[35] Ali Jadbabaie et al. "Online optimization: Competing with dynamic comparators". In: *Artificial Intelligence and Statistics*. PMLR. 2015, pp. 398–406.

[36] Tianbao Yang et al. "Tracking slowly moving clairvoyant: Optimal dynamic regret of online learning with true and noisy gradient". In: *International Conference on Machine Learning*. PMLR. 2016, pp. 449–457.

[37] Aryan Mokhtari et al. "Online optimization in dynamic environments: Improved regret rates for strongly convex problems". In: *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE. 2016, pp. 7195–7201.

[38] Lijun Zhang, Shiyin Lu, and Zhi-Hua Zhou. "Adaptive online learning in dynamic environments". In: *Advances in neural information processing systems* 31 (2018).

[39] Lijun Zhang, Tianbao Yang, Zhi-Hua Zhou, et al. "Dynamic regret of strongly adaptive methods". In: *International conference on machine learning*. PMLR. 2018, pp. 5882–5891.

[40] Vladimir Vovk. "Metric entropy in competitive on-line prediction". In: *arXiv preprint cs/0609045* (2006).

[41] Alexander Rakhlin and Karthik Sridharan. "Online non-parametric regression". In: *Conference on Learning Theory*. PMLR. 2014, pp. 1232–1264.

[42] Pierre Gaillard and Sébastien Gerchinovitz. "A chaining algorithm for online nonparametric regression". In: *Conference on Learning Theory*. PMLR. 2015, pp. 764–796.

[43] Alexander Rakhlin and Karthik Sridharan. "Online nonparametric regression with general loss functions". In: *arXiv preprint arXiv:1501.06598* (2015).

[44] Dheeraj Baby and Yu-Xiang Wang. "Adaptive online estimation of piecewise polynomial trends". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 20462–20472.

[45] Paul Liautaud, Pierre Gaillard, and Olivier Wintenberger. "Minimax-optimal and locally-adaptive online nonparametric regression". In: *Proceedings of Machine Learning Research vol* 272 (2025), pp. 1–34.

[46] David L Donoho. "De-noising by soft-thresholding". In: *IEEE transactions on information theory* 41.3 (2002), pp. 613–627.

Appendix A

Plots

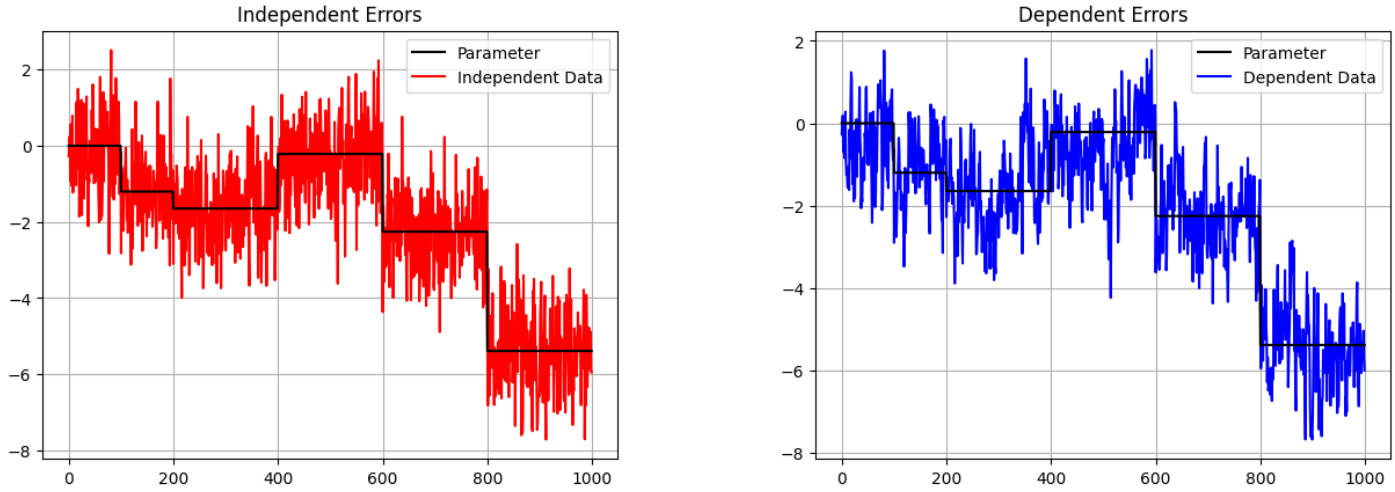

Figure A.1: Sample data generated according to the hard shift regime with independent and dependent noise respectively together with the underlying signal.

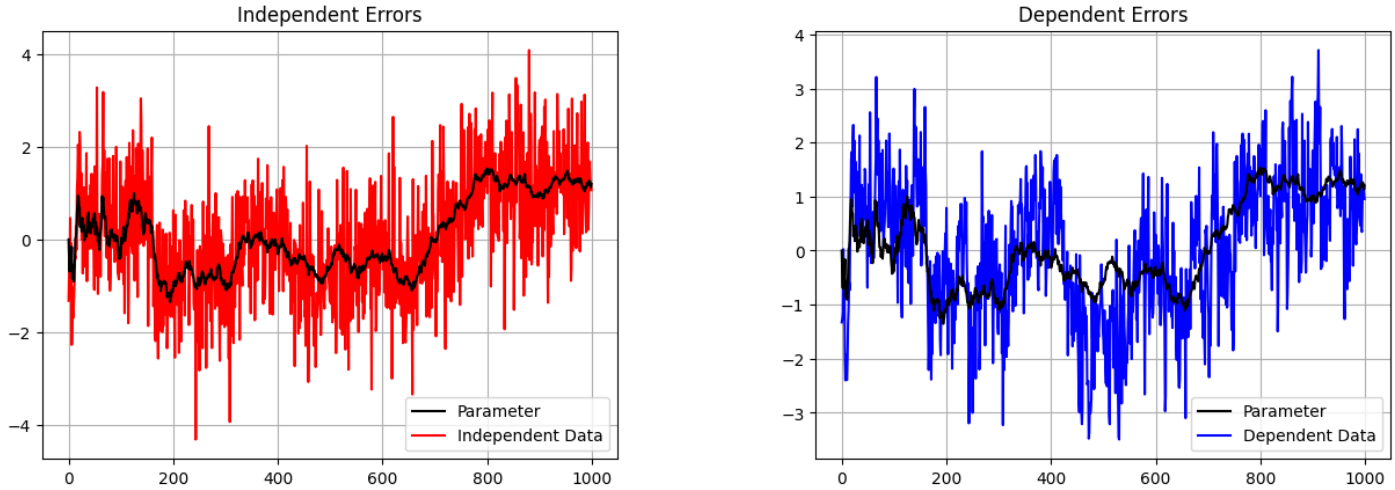

Figure A.2: Sample data generated according to the hard shift regime with independent and dependent noise respectively together with the underlying signal.

We can visually see that the presence of interdependence within the noise makes prediction of the underlying signal much more difficult. Take for example the half-way point of Figure A.2 - the signal is not even 'covered' by the actual data when the errors are dependent.

Now Figure A.3 and Figure A.4 are examples of predictions by the 3 algorithms of interest for the data in Figure A.1 and Figure A.2.

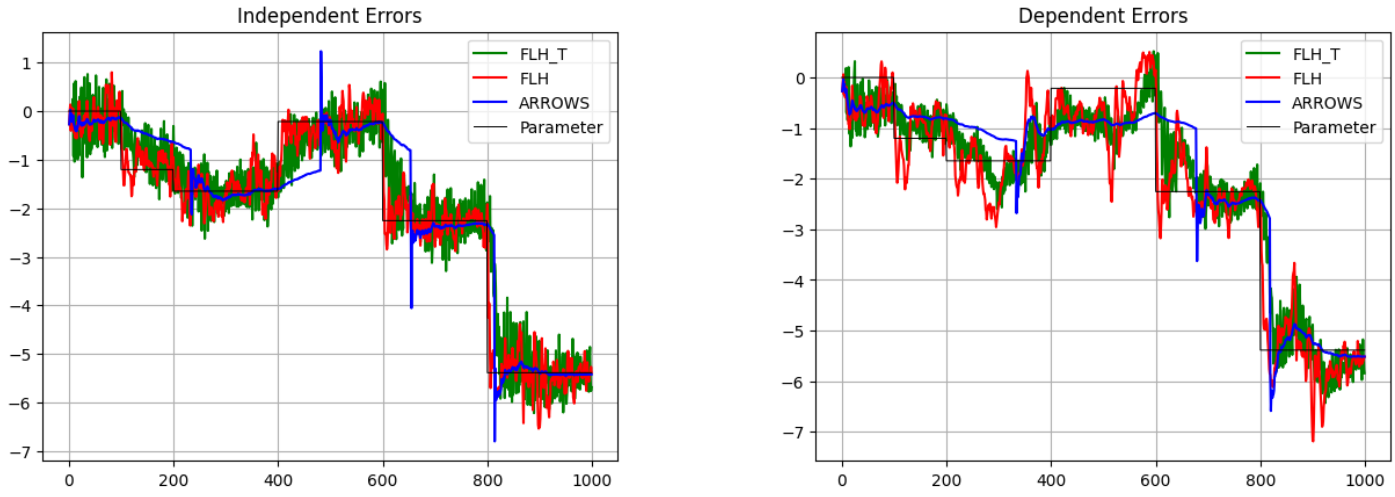

Figure A.3: Predictions generated according to FLH, TFLH, and ARROWS for data generated according to the hard shift regime together with the underlying signal.

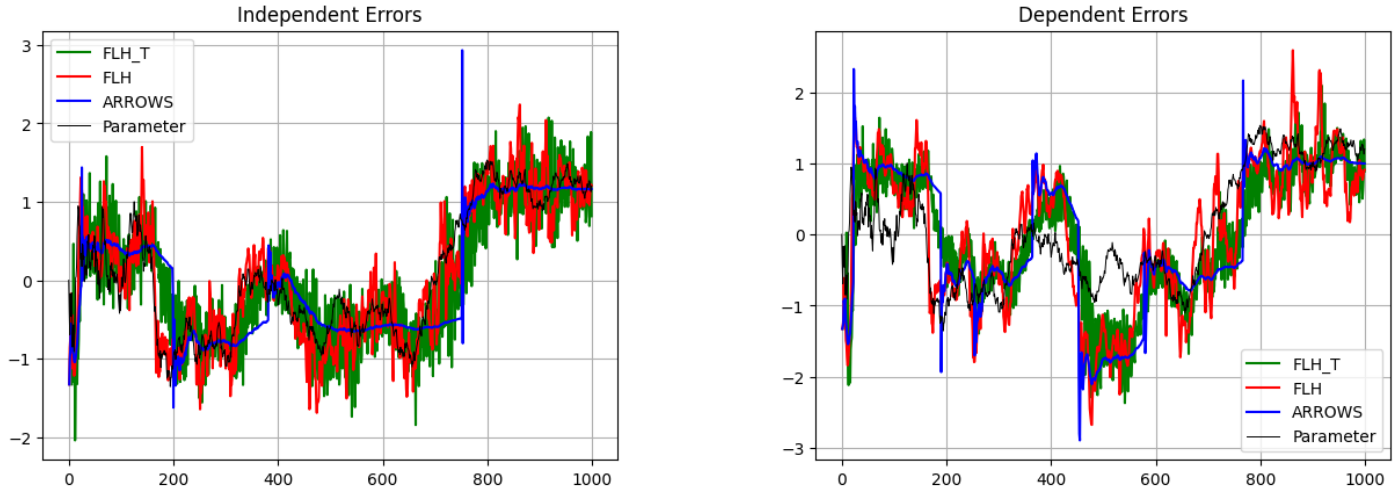

Figure A.4: Predictions generated according to FLH, TFLH, and ARROWS for data generated according to the soft shift regime together with the underlying signal.

Appendix B

Improved Follow-the-Leading-History

Algorithm 4 Improved Follow-the-Leading-History (Updated Raj 2020)

Input: black box algorithm $\mathcal{A}$, learning parameter $\alpha > 0$

1: **Init:** Weight vector $\boldsymbol{v}_0 = (v_0^{(1)}, \ldots, v_0^{(n)}) = (0, \ldots, 0)$

2: **for** $t = 1, \ldots, n$ **do**

3: Start a new instance of algorithm $\mathcal{A}$ denoted by $\mathcal{A}_t$ and assign the weights $v_t^{(t)} = \frac{1}{t}$ and $\widehat{v}_t^{(t)} = \frac{1}{t}$

4: Define $\tau_t := t + 2^k$ where $k := \min\{k \geq 0 : c_k > 0\}$ and $t := \sum_{i=1}^{\infty} c_k 2^k$

5: Define the set of active experts

$$S_t := \{1 \leq i \leq : \tau_i > t - 1\} \tag{B.1}$$

6: Normalise the weight of each expert $i \in S_t \setminus \{t\}$ so that

$$\widehat{v}_t^{(i)} = \frac{v_t^{(i)}}{\sum_{j \in S_t \setminus \{t\}} v_t^{(j)}} \tag{B.2}$$

7: Receive the prediction $\widehat{\mu}_t^{(i)}$ from each black box algorithm $\mathcal{A}_i$, $i \in S_t \setminus \{t\}$

8: Predict

$$\widehat{\mu}_t = \sum_{i \in S_t \setminus \{t\}} \widehat{v}_t^{(i)} \widehat{\mu}_t^{(i)}. \tag{B.3}$$

and observe $y_t \in \mathbb{R}$.

9: Set $v_{t+1}^{(t)} = v_t^{(t)}$ and update the weights for each $i \in S_t \setminus \{t\}$

$$v_{t+1}^{(i)} = v_t^{(i)} \exp\left(-\alpha f_t(\widehat{\mu}_t^{(i)})\right).$$

The intuition behind why IFLH works is that experts which are close to each other, i.e. starting from very close time steps, will give very similar predictions and furthermore that the accuracy of any expert decreases as they move further and further away from the present. As a result, we are often 'double-counting' and keeping track of moving averages which are either redundant or reach so far back into the past that they cannot hope to meaningfully contribute to the final prediction. Then IFLH implements a scheme by which we may prune the set of active experts at each time step, maintaining only a sparse, strategically chosen subset. The trick to make this work is that the method by which experts are pruned is deliberately not random, as seen in step 4 and 5 of Algorithm 4. Instead, it is a deterministic process designed to ensure that the set of active experts remains 'well-spread' across the history of the data stream. We can see the properties of the pruning process in Proposition B.1, the proof of which may be found in Appendix B of [1].

Proposition B.1. [1] Let S_t be defined as in (B.1). Then S_t enjoys the following properties:

1. for all $s \geq t$, we have $[s, (s+t)/2] \cap S_t \neq \emptyset$,
2. for all $t = 1, \ldots, n$ we have $|S_t| = \mathcal{O}(\log n)$,
3. for all $t = 1, \ldots, n$, we have $S_{t+1} \setminus S_t = \{t\}$.

As a consequence of this pruning of active experts, the running time of the IFLH algorithm is $\mathcal{O}(V \log n)$ [1], a significant improvement that makes large-scale simulations much more feasible. Since we found a $\mathcal{O}(1)$ running time implementation of moving average sub-routines, this gives the algorithm a $\mathcal{O}(\log n)$ running time.

Note that IFLH is able to sparsify the set of experts whilst still retaining the same regret bounds as FLH. Indeed if FLH achieves a regret of $R(n)$ over n rounds, then IFLH will have an expected regret of $\mathcal{O}(R(n) \log n)$, see [1], Theorem 4.1. That being said, the results of Chapter 2 and Chapter 3 were obtained for FLH and while the rates may be the same minus a $\log n$ factor, experimental results showed that IFLH nonetheless performed significantly worse than FLH. For these reasons, we will conduct our simulation study using FLH and keep the time horizon relatively small.

Appendix C

Equivalence of FLH Formulations

The following is the formulation of the FLH algorithm as it is written in [3].

Algorithm 5 Follow-the-Leading-History (Raj et al., 2020) [3]

Input: black box algorithm $\mathcal{A}$, learning parameter $\alpha > 0$

Init: Weight vector $\boldsymbol{v}_0 = (v_0^{(1)}, \dots, v_0^{(n)}) = (0, \dots, 0)$

for $t = 1, \dots, n$ **do**

Start a new instance of algorithm $\mathcal{A}$ denoted by $\mathcal{A}_t$ and assign the weight $\widehat{v}_t^{(t)} = \frac{1}{t}$

Normalise the weight of each expert $i \in \{1, \dots, t-1\}$ so that

$$\widehat{v}_t^{(i)} = \left(1 - \frac{1}{t}\right) \frac{v_t^{(i)}}{\sum_{j=1}^{t-1} v_t^{(j)}} \tag{C.1}$$

Observe x_t and receive the prediction $\widehat{\mu}_t^{(i)}$ from each black box algorithm $\mathcal{A}_i$, $i \in \{1, \dots, t-1\}$

Predict

$$\widehat{\mu}_t = \sum_{i=1}^{t-1} \widehat{v}_t^{(i)} \widehat{\mu}_t^{(i)} \tag{C.2}$$

and observe $y_t \in \mathbb{R}$

Update the weights for each $i \in \{1, \dots, t-1\}$

$$v_{t+1}^{(i)} = v_t^{(i)} \exp\left(-\alpha f_t(\widehat{\mu}_t^{(i)})\right) = v_t^{(i)} \exp\left(-\alpha (y_t - \widehat{\mu}_t^{(i)})^2\right).$$

It can quickly be shown that the estimator produced by Algorithm 5 is

$$\begin{aligned}
\widehat{\mu}_t^{\mathrm{R}} &= \sum_{i=1}^{t-1} \widehat{v}_t^{(i)} \widehat{\mu}_t^{(i)} \\
&= \sum_{i=1}^{t-1} \left(1 - \frac{1}{t}\right) \frac{v_t^{(i)}}{\sum_{j=1}^{t-1} v_t^{(j)}} \widehat{\mu}_t^{(i)} \\
&= \sum_{i=1}^{t-1} \left(1 - \frac{1}{t}\right) \frac{v_{t-1}^{(i)} \exp\left(-\alpha \left(y_{t-1} - \widehat{\mu}_{t-1}^{(i)}\right)^2\right)}{\sum_{j=1}^{t-1} v_{t-1}^{(j)} \exp\left(-\alpha \left(y_{t-1} - \widehat{\mu}_{t-1}^{(j)}\right)^2\right)} \widehat{\mu}_t^{(i)}.
\end{aligned}$$

Now there are two issues. The first has to do with how the weights of each expert are initialised. Algorithm 5

initialises each expert $\mathcal{A}_t$ with the weight $\widehat{v}_t^{(t)} = \frac{1}{t}$, but it is the unnormalised $v_t^{(t)}$ which is updated in step 7 of the algorithm and then the updated $v_{t+1}^{(i)}$ is used to find the normalised $\widehat{v}_{t+1}^{(i)}$. However, there is no initialisation of $v_t^{(t)}$. Let us examine why this happens.

At round $t = k$, the algorithm $\mathcal{A}_k$ is initialised with hat weight $\widehat{v}_k^{(k)} = \frac{1}{k}$. Then in step 5, we see that this expert does not give us a prediction at this round. This makes sense when we consider that each expert is a moving average. Then at this round, this expert would just be the average of the current data point, but we only observe this data point in a later step. If $k = 2$, then $\mathcal{A}_1$ gives us prediction y_1 and $\mathcal{A}_2$ has no data with which to predict. Since $\mathcal{A}_k$ does not give us a prediction at round $t = k$, we do not perform the weight update in step 7 of the algorithm for the weights of $\mathcal{A}_k$.

Now, at round $t = k+1$, A_k enters the set of active experts from which we receive a prediction and receives the normalised weight we see in (C.1). However, note the normalisation constant includes the weight $v_{k+1}^{(k)}$, but the only weight we have initialised for $\mathcal{A}_k$ is $\widehat{v}_k^{(k)}$. So we need to initialise $v_{k+1}^{(k)} = \frac{1}{k}$ in step 3 at round $t = k+1$.

The solution to this is to initialise $v_k^{(k)} = \frac{1}{k}$ and $\widehat{v}_k^{(k)} = \frac{1}{k}$ in step 3 for round $t = k$. Then, in round $t = k$ we also perform the weight update $v_{k+1}^{(k)} = v_k^{(k)}$ during step 7. The reason for this redundancy is that we still require $\widehat{v}_{k+1}^{(k)} = \frac{1}{k+1}$ for the proof of Lemma 2.7.

The second issue becomes more clear once we examine the original formulation proposed by [1].

Algorithm 6 Follow-the-Leading-History (Hazan and Seshadhri, 2007) [1]

Input: black box algorithm $\mathcal{A}$, learning parameter $\alpha > 0$

1: **Init:** $\mathcal{A}_1, \dots, \mathcal{A}_n$ instances of $\mathcal{A}$ and weight vector $\boldsymbol{v}_0 = (v_0^{(1)}, v_0^{(2)}, \dots, v_0^{(n)}) = (1, 0, \dots, 0)$.

2: **for** $t = 1, \dots, n$ **do**

3: Observe x_t and receive the prediction $\widehat{\mu}_t^{(i)}$ from each black box algorithm $\mathcal{A}_i$, $i \in \{1, \dots, t\}$

4: Predict $\widehat{\mu}_t = \sum_{i=1}^t v_t^{(i)} \widehat{\mu}_t^{(i)}$ and observe $y_t \in \mathbb{R}$

5: Set for each $i \in \{1, \dots, t\}$

$$\widehat{v}_{t+1}^{(i)} = \frac{v_t^{(i)} e^{-\alpha f_t(\widehat{\mu}_t^{(i)})}}{\sum_{j=1}^t v_t^{(j)} e^{-\alpha f_t(\widehat{\mu}_t^{(j)})}} = \frac{v_t^{(i)} e^{-\alpha (y_t - \widehat{\mu}_t^{(i)})^2}}{\sum_{j=1}^t v_t^{(j)} e^{-\alpha (y_t - \widehat{\mu}_t^{(j)})^2}}.$$

6: Set $v_{t+1}^{(t+1)} = \frac{1}{t+1}$ and update the weights for each $i \in \{1, \dots, t\}$

$$v_{t+1}^{(i)} = \left(1 - \frac{1}{t+1}\right) \widehat{v}_t^{(i)}.$$

It can quickly be shown that the estimator produced Algorithm 6 is

$$\begin{aligned} \widehat{\mu}_t^{\mathrm{H}} &= \sum_{i=1}^t v_t^{(i)} \widehat{\mu}_t^{(i)} \qquad \text{(C.3)} \\ &= \sum_{i=1}^{t-1} \left(1 - \frac{1}{t}\right) \widehat{v}_t^{(i)} \widehat{\mu}_t^{(i)} + \widehat{v}_t^{(t)} \widehat{\mu}_t^{(y)} \\ &= \sum_{i=1}^{t-1} \left(1 - \frac{1}{t}\right) \frac{v_{t-1}^{(i)} \exp\left(-\alpha \left(y_{t-1} - \widehat{\mu}_{t-1}^{(i)}\right)^2\right)}{\sum_{j=1}^{t-1} v_{t-1}^{(j)} \exp\left(-\alpha \left(y_{t-1} - \widehat{\mu}_{t-1}^{(j)}\right)^2\right)} \widehat{\mu}_t^{(i)} + \frac{1}{t} \widehat{\mu}_t^{(t)}. \qquad \text{(C.4)} \end{aligned}$$

The key difference between $\widehat{\mu}_t^{\mathrm{R}}$ and $\widehat{\mu}_t^{\mathrm{H}}$ is the last term. Examining why there is this discrepancy reveals the second issue - the weights $\widehat{v}_t^{(i)}$ of Algorithm 5 are not actually normalised weights in the sense that $\sum_{i=1}^{t-1} \widehat{v}_t^{(i)} \neq 1$.

The reason for this is that we have this $\left(1-\frac{1}{t}\right)$ factor when we normalise $v_t^{(i)}$ and in the Algorithm 6 this factor's purpose is to account for the last term in (C.4) having a weight of $\frac{1}{t}$. However we do not use this term immediately in the same round in Algorithm 5, we have instead $t-1$ terms. Instead the defined $\widehat{v}_k^{(k)}=\frac{1}{k}$ is first used in round $t=k+1$, however it is also then immediately used in the normalisation in this round. So in fact the sum of our weights at $t=k+1$ is off by a term of $\frac{1}{k}$.

The solution to this is to rewrite step 4 so that our normalised weights do indeed sum to 1. However before we do this, we need to add one extra alteration to the algorithm - we add to step 4 that $\widehat{v}_t^{(t-1)}=\widehat{v}_{t-1}^{(t-1)}=\frac{1}{t-1}$. Again, this is another redundant update, however we still require that $\widehat{v}_t^{(t)}=\frac{1}{t}$ for the proof Lemma 2.7. Now, (C.1) may be rewritten as

$$\widehat{v}_t^{(i)}=\left(1-\frac{1}{t-1}\right)\frac{v_t^{(i)}}{\sum_{j=1}^{t-2}v_t^{(j)}}$$

for each $i\in\{1,\dots,t-2\}$. In this way we see that

$$\sum_{i=1}^{t-1}\widehat{v}_t^{(i)}=\sum_{i=1}^{t-2}\left(1-\frac{1}{t-1}\right)\frac{v_t^{(i)}}{\sum_{j=1}^{t-2}v_t^{(j)}}+\frac{1}{t-1}=1-\frac{1}{t-1}+\frac{1}{t-1}=1$$

as required.

It remains to show that the estimator produced by Algorithm 1 is indeed equivalent to the Algorithm 6 estimator. For convenience, we recall Algorithm 1 here.

Algorithm 7 Follow-the-Leading-History [3, 1]

Input: black box algorithm $\mathcal{A}$, learning parameter $\alpha>0$

1: **Init:** Weight vector $\boldsymbol{v}_0=(v_0^{(1)},\dots,v_0^{(n)})=(0,\dots,0)$

2: **for** $t=1,\dots,n$ **do**

3: Start a new instance of algorithm $\mathcal{A}$ denoted by $\mathcal{A}_t$ and assign the weights $v_t^{(t)}=\frac{1}{t}$ and $\widehat{v}_t^{(t)}=\frac{1}{t}$.

4: Normalise the weight of each expert $i\in\{1,\dots,t-2\}$ so that

$$\widehat{v}_t^{(i)}=\left(1-\frac{1}{t-1}\right)\frac{v_t^{(i)}}{\sum_{j=1}^{t-2}v_t^{(j)}} \tag{C.5}$$

and update $\widehat{v}_t^{(t-1)}=\widehat{v}_{t-1}^{(t-1)}$.

5: Receive the prediction $\widehat{\mu}_t^{(i)}$ from each black box algorithm $\mathcal{A}_i$, $i\in\{1,\dots,t-1\}$.

6: Predict

$$\widehat{\mu}_t=\sum_{i=1}^{t-1}\widehat{v}_t^{(i)}\widehat{\mu}_t^{(i)}. \tag{C.6}$$

and observe $y_t\in\mathbb{R}$.

7: Set $v_{t+1}^{(t)}=v_t^{(t)}$ and update the weights for each $i\in\{1,\dots,t-1\}$

$$v_{t+1}^{(i)}=v_t^{(i)}\exp\left(-\alpha f_t(\widehat{\mu}_t^{(i)})\right).$$

Now, it can be quickly show that

$$\begin{aligned}
\widehat{\mu}_t^{\mathrm{R_{updated}}} &= \sum_{i=1}^{t-1} \widehat{v}^{(i)} \widehat{\mu}_t^{(i)} \\
&= \sum_{i=1}^{t-2} \left(1 - \frac{1}{t-1}\right) \frac{v_t^{(i)}}{\sum_{j=1}^{t-2} v_t^{(j)}} \widehat{\mu}_t^{(i)} + \widehat{v}_t^{(t-1)} \widehat{\mu}_t^{(t-1)} \\
&= \sum_{i=1}^{t-2} \left(1 - \frac{1}{t-1}\right) \frac{v_{t-1}^{(i)} \exp\left(-\alpha \left(y_{t-1} - \widehat{\mu}_{t-1}^{(i)}\right)^2\right)}{\sum_{j=1}^{t-2} v_{t-1}^{(j)} \exp\left(-\alpha \left(y_{t-1} - \widehat{\mu}_{t-1}^{(j)}\right)^2\right)} \widehat{\mu}_t^{(i)} + \frac{1}{t-1} \widehat{\mu}_t^{(t-1)}.
\end{aligned}$$

Now, we recall that

$$\widehat{\mu}_t^{\mathrm{H}} = \sum_{i=1}^{t-1} \left(1 - \frac{1}{t}\right) \frac{v_{t-1}^{(i)} \exp\left(-\alpha \left(y_{t-1} - \widehat{\mu}_{t-1}^{(i)}\right)^2\right)}{\sum_{j=1}^{t-1} v_{t-1}^{(j)} \exp\left(-\alpha \left(y_{t-1} - \widehat{\mu}_{t-1}^{(j)}\right)^2\right)} \widehat{\mu}_t^{(i)} + \frac{1}{t} \widehat{\mu}_t^{(t)}.$$

It seems now that we are back where we stared since $\widehat{\mu}_t^{\mathrm{H}}$ has this extra $\widehat{\mu}_t^{(t)}$ term with weight $\frac{1}{t}$ whereas $\widehat{\mu}_t^{\mathrm{R_{updated}}}$ assigns the weight $\frac{1}{t-1}$ to $\widehat{\mu}_t^{(t-1)}$, and on top of this we have this $\left(1 - \frac{1}{t}\right)$ factor in one and $\left(1 - \frac{1}{t-1}\right)$ in the other, however these two formulations are now practically equivalent. The crux of the equivalence argument lies in (C.6) and step 4 of the Algorithm 6, which is (C.3).

We see that Algorithm 6 is using k experts at round $t = k$, whereas the Algorithm 1 is using $k-1$ experts for predictions. This difference is negligible when talking about asymptotic behaviours as we do in our cumulative error bounds. Furthermore, we are dealing with moving averages. At $t = 1$, we have not yet observed y_1, so $\mathcal{A}_1$ cannot make a prediction. Only at $t = 2$, when we have just observed y_1 in the previous round, may $\mathcal{A}_1$ make the prediction $\widehat{\mu}_2^{(1)} = y_1$.

As a result, the amount of experts at round $t = k$ from which we receive a prediction is k and $k - 1$ for Algorithm 6 and Algorithm 1 respectively. This then informs the weight of our most recent expert, which is then either $\frac{1}{k}$ or $\frac{1}{k-1}$ depending on the number of active experts. This is because in Algorithm 6, at round $t = 2$, we only receive a prediction from $\mathcal{A}_1$, so its weight must be $\frac{1}{2-1} = 1$ - by contrast in Algorithm 6, we receive predictions from $\mathcal{A}_1$ and $\mathcal{A}_2$, so the most recent expert, A_2, receives the weight $\frac{1}{2}$. Then finally, this 'most recent' weight informs our normalisation factor - either $\left(1 - \frac{1}{t}\right)$ or $\left(1 - \frac{1}{t-1}\right)$ for Algorithm 6 and Algorithm 1 respectively. In this way we see that $\widehat{\mu}_t^{\mathrm{H}}$ is just $\widehat{\mu}_t^{\mathrm{R_{updated}}}$ one round later.