On tracking varying bounds when forecasting bounded time series.

Journal:	*Technometrics*
Manuscript ID	TCH-23-029.R1
Manuscript Type:	Original Article
Keywords:	Generalized logit-normal distribution, Normalized Gradient Descent, Online quasiconvex optimization, Inventory problem, Wind power probabilistic forecasting
Note: The following files were submitted by the author for peer review, but cannot be converted to PDF. You must view these files (e.g. movies) online.	
mybound_for_technometrics-R1.zip	

On tracking varying bounds when forecasting bounded time series

Abstract

We consider a new framework where a continuous, though bounded, random variable has unobserved bounds that vary over time. In the context of univariate time series, we look at the bounds as parameters of the distribution of the bounded random variable. We introduce an extended log-likelihood estimation and design algorithms to track the bound through online maximum likelihood estimation. Since the resulting optimization problem is not convex, we make use of recent theoretical results on stochastic quasiconvex optimization, to eventually derive an Online Normalized Gradient Descent algorithm. We illustrate and discuss the workings of our approach based on both simulation studies and a real-world wind power forecasting problem.

Keywords: Generalized logit-normal distribution; Normalized Gradient Descent; Online quasiconvex optimization; Inventory problem; Wind power probabilistic forecasting.

1 Introduction

Many statistical applications involve response variables which are both continuous and bounded. This is especially the case when one has to deal with rates, percentages or proportions, for example when interested in the spread of an epidemic (Guolo and Varin, 2014), the unemployment rates in a given country (Wallis, 1987) or the proportion of time spent by animals in a certain activity (Cotgreave and Clayton, 1994). Indeed, proportional data are widely encountered within ecology-related statistical problems, see Warton and Hui (2011) among others. Similarly, when forecasting wind power generation, the response variable is also such a continuous bounded variable. Wind power generation is a stochastic process with continuous state space which is bounded from below by zero when there is no wind, and from above by the nominal capacity of the turbine (or wind farm) for high-enough wind speeds. More generally, renewable energy generation from both wind and solar energy are bounded stochastic processes, with the same lower bound (i.e., zero energy production) and different characteristics of their upper bound (since solar energy generation has a time-varying maximum depending on the time of day and time of year), see for example Pinson (2012) and Bacher et al. (2009).

These continuous bounded random variables call for probability distributions with a bounded support such as the beta distribution, truncated distributions or distributions of transformed normal variables as discussed by Johnson (1949). Often, the response variable is first assumed to lie in the unit interval $(0, 1)$ and is then rescaled to any interval (a, b). For applications involving such response variables, these bounds (a, b) are always fixed to the same values over the sample. While this assumption makes sense in some cases, we argue

it can be misleading and negatively impacts inference when the bounds (a, b) actually vary over time or depending on exogenous variables whilst not being observed. In particular, it is highly relevant for energy applications, such as wind power forecasting, as the upper bound b may change over time, while being unknown, for example in case of curtailment actions for which information is not available or not reliable. Another application is the inventory problem of the retailer (Laderman and Littauer, 1953). Let X_t be the demand for a certain item at time t: X_t is double-bounded, from below by zero and from above by the stock available at time t, i.e., by a time-varying upper bound b_t. To prepare for demand X_{t+1}, the retailer needs to find the quantity they should order in the light of the knowledge they have of the past stocks and demands. Similarly to the problem of forecasting wind power generation, the inventory problem might then involve a double-bounded random variable, the demand for a certain item, which can be regarded as a continuous variable for large quantities being involved, and upper bounded by a bound which may vary over time whilst not being observed, for example in case of supply chain issues, information mismanagement or very large retailers that could not track the evolution of the stocks for each item or would rather benefit from an automatic data-driven tracking.

In both those applications, if the random variable happens to get very close to the upper bound, it might be the case that a higher upper bound would have resulted in a higher wind power generation or item demand. Therefore we do not observe the "true" power generation nor item demand. In that sense one could arguably think of it as being related to censoring and truncation. However, while truncation assumes the value of the response variable to be never seen (or recorded) if above the upper bound, and censoring assumes

one does not know the exact value but does know it lies above the upper bound, we assume here that an upper bound lower than the "true" response results in squeezing the observed value of the variable, and thus in reshaping the probability distribution of the variable.

There are at least two ways of looking at varying bounds which cannot be observed. They can be introduced in the model as latent random variables A and B. Hence, the distribution of the response variable is conditional given A and B. The main advantage of this approach is its generality and flexibility, with A and B being distributed according to a well-specified probability distribution, which might depend on exogenous variables. Suppose we assume a parametric model for both the bounds and the response variable. Because we do not have access to the realizations of the bounds, the maximization of the likelihood might involve complicated high-dimensional integration, possibly computationally infeasible, and would therefore call for algorithms of the Expectation-Maximization kind (Dempster et al., 1977). Moreover, for forecasting applications, one needs to first compute (good enough) forecasts of the bounds in order to be able to forecast the response variable.

In the context of stochastic processes, varying bounds which cannot be observed can alternatively be considered as scaling parameters a and b of the parametric distribution of the bounded variable. Hence, a non-stationary framework is to be used for these additional parameters to be able to evolve over time. We will focus in this paper on discrete-time stochastic processes with an upper varying bound. Among the class of distributions with a bounded support, we choose the generalized logit-normal (GLN) distribution introduced by Mead (1965). The practical use of any family of distributions depends on the possible

variation in its shape, and on the ease with which the distribution can be fitted. The GLN distribution is very flexible, and because the transformed variable is normally distributed, the probability density function (pdf) of the bounded variable X_t can be expressed as a function of the standard normal density.

We aim to estimate the parameter vector θ of the pdf of X_t which includes a bound parameter b through Maximum Likelihood Estimation (MLE). The first challenge we need to tackle when dealing with the bound as a parameter in a non-stationary setup is how to handle past observations which are out of the support $(0, b)$ of the bounded distribution of X_t and make the log-likelihood to be infinite. We introduce a new term which relies on the sigmoid function to take into account those observations in a "soft" finite way. Another challenge is that when considering the bound as a parameter, we cannot assume convexity anymore as the negative log-likelihood appears not to be convex with respect to (w.r.t.) the new bound parameter. Instead, we propose to assume quasiconvexity and use recent results from Hazan et al. (2015) about local quasiconvexity and Stochastic Normalized Gradient Descent (SNGD) to design an online algorithm. We present the statistical parametric model with a varying upper bound in Section 2 and the corresponding MLE in Section 3. In Section 4, we perform simulations of synthetic data to run the algorithm we proposed in Section 3. First we look at the performance when tracking the parameter vector θ over time, then when forecasting the probability distribution of the bounded variable. In Section 5, we provide 10-min-ahead probabilistic forecasts of the wind power generation at Anholt offshore wind farm (Denmark) using this new framework. Finally, we discuss the results, the limitations and some prospects of the methodology in Section 6.

2 Statistical model

2.1 Parametric distribution including the new parameter b

Let $\tilde{X}_t$ be a continuous bounded random variable, $\tilde{X}_t \in (0,1)$, and X_t be the corresponding variable rescaled to $(0,b)$ by applying the transformation $X_t = b\tilde{X}_t$. The generalized logit transform $Y_t \in \mathbb{R}$ of $X_t \in (0,b)$ is given by

$$Y_t = \gamma(X_t/b;\nu) = \log \frac{(X_t/b)^\nu}{1-(X_t/b)^\nu}, \quad \nu > 0,$$

where ν is the shape parameter. When Y_t is distributed according to a Gaussian distribution $\mathcal{N}(\mu,\sigma^2)$, the original variable X_t/b is then distributed according to a GLN distribution $L_\nu(\mu,\sigma^2)$, see for example Frederic and Lad (2008) and Pinson (2012). To handle dependent observations, we assume the expectation of Y_t to be an auto-regressive (AR) process of order p, $\mu_t = \sum_{k=1}^{p} \lambda_k \gamma(x_{t-k}/b;\nu)$. Note that successive time points t are assumed to be spaced at uniform intervals. The pdf of X_t conditional on the previous information set $\mathcal{F}_{t-1}$ (the σ-algebra generated by $X_1,\ldots,X_{t-1}$), with parameter vector $\theta = (\lambda_1,\ldots,\lambda_p,\sigma^2,\nu,b)$, is then

$$p_\theta(x_t|\mathcal{F}_{t-1}) = \begin{cases} \frac{1}{\sqrt{2\pi\sigma^2}} \frac{\nu}{x_t(1-(x_t/b)^\nu)} \exp\left[-\frac{1}{2}\left(\frac{\gamma(x_t/b;\nu)-\mu_t}{\sigma}\right)^2\right] & \text{if } 0 < x_{t-k} < b, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

where $k = 0,\ldots,p$.

2.2 Extended time-dependent log-likelihood function

We wish to estimate the parameter vector θ of the pdf $p_\theta(x_t|\mathcal{F}_{t-1})$ in (1) through MLE. In the case of a stationary time series and constant parameter θ, this comes to minimizing

the negative log-likelihood objective function

$$-l(\theta) = -\sum_{t=p+1}^{T} \log p_\theta(x_t|\mathcal{F}_{t-1}) \tag{2}$$

w.r.t. θ, the data sample $x_1, \ldots, x_T$ being fixed, assuming the random variables X_t are i.i.d conditionally on $\mathcal{F}_{t-1}$. Note that we do not take into account the distribution of the first p observed values $(x_1, \ldots, x_p)$ and consider instead the likelihood conditional on them. For ease of notation and because the negative log-likelihood is to be minimized w.r.t. θ, let $p_\theta(x_j|\mathcal{F}_{j-1}) = p_j(\theta)$. In a non-stationary framework, an estimate of θ at time t can be retrieved by minimizing a time-dependent negative log-likelihood, such as

$$-l_t(\theta) = -\frac{1}{n_\alpha} \sum_{j=p+1}^{t} \alpha^{t-j} \log p_j(\theta), \tag{3}$$

where $\alpha \in (0, 1]$, is an exponential forgetting factor and $n_\alpha = \frac{1}{1-\alpha}$ if $\alpha < 1$, $n_\alpha = t - p$ if $\alpha = 1$, is used to normalize the weighted negative log-likelihood. From (1) we can see that the negative log-likelihood in (2) we wish to minimize takes the value $+\infty$ as soon as an observation x_t is greater or equal to b. This is an implicit constraint on b when estimating $\hat{\theta}$. However, in a non-stationary framework we do not want b to be greater than all the observations x_t as b should be able to vary over time. Let $U_t = \{p + 1, , \ldots, t\}$, $C_t(\theta) = \{j \in U_t \mid x_{j-k} < b, k = 0, \ldots, p\}$ and $\overline{C_t}(\theta) = \{j \in U_t \mid j \notin C_t(\theta)\}$ the complement of $C_t(\theta)$ in U_t. The log-likelihood takes finite values only for observations x_j such that $j \in C_t(\theta)$. Therefore we can - informally - rewrite $\sum_{j=p+1}^{t} \alpha^{t-j} \log p_j(\theta)$ as $\sum_{j \in C_t(\theta)} \alpha^{t-j} \log p_j|_b(\theta) + \sum_{j \in \overline{C_t}(\theta)} \alpha^{t-j} \log 0$, where $p_j|_b(\theta)$ is the pdf $p_j(\theta)$ restricted to its support $(0, b)$. When estimating the parameter vector at time t, we need to take into account all past observations, including the observations for which the log-likelihood does not take a

finite value, i.e., including observations x_j such that j does not belong to $C_t(\theta)$. Therefore, we propose to replace the value 0 in $\log 0$, which originally corresponds to the value of the pdf $p_j(\theta)$ outside of its support, with a sigmoid function of $b - x_j$,

$$s_j(b) = \frac{1}{1 + \exp(-b + x_j)}. \tag{4}$$

The function s_j is illustrated in Figure 1. It can be seen as the probability of x_j to be

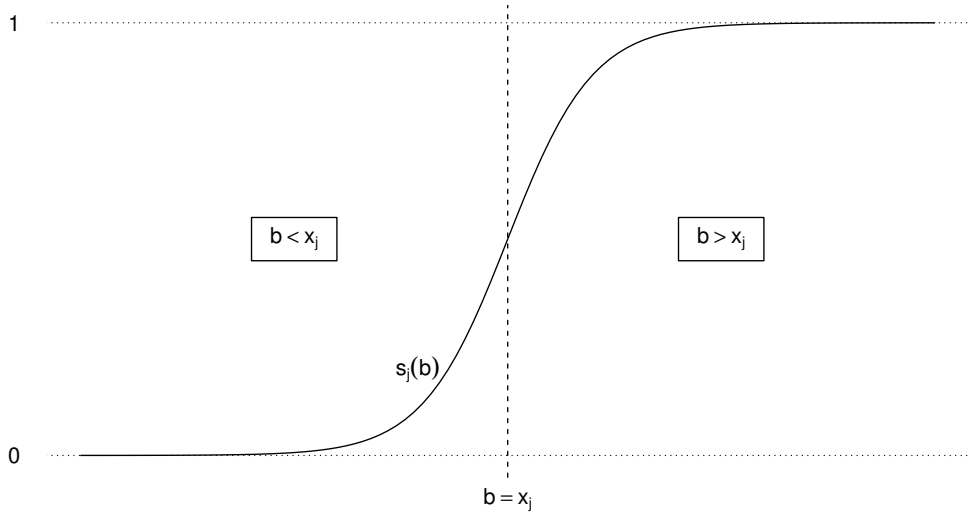

Figure 1: Sigmoid function $s_j(b)$ on the real line.

lower or equal than b: we have $s_j(b) \to 0^+$ when $b << x_j$ and $s_j(b) \to 1^-$ when $b >> x_j$. Moreover $-\log s_j(b)$ is convex and differentiable in b. Hence, we propose to use

$$-l_t^{\infty}(\theta) = -\frac{1}{n_\alpha}\left[\sum_{j \in C_t(\theta)} \alpha^{t-j} \log p_j(\theta) + \sum_{j \in \overline{C_t}(\theta)} \alpha^{t-j} \log s_j(b)\right], \tag{5}$$

which we refer to as the *extended* time-dependent negative log-likelihood. One can note that $j \in \overline{C_t}(\theta)$ does not necessarily mean $x_j \geq b$ as it can happen because a lagged observation x_{j-k} is such that $x_{j-k} \geq b$. In such a case, that is $j \in \overline{C_t}(\theta)$ and $x_j < b$, the observation x_j will still increase the value of the total log-likelihood compared to the event $\{x_j \geq b\}$, which is also a nice feature of using s_j.

2.3 (Local-)Quasiconvexity

While convexity, or pseudoconvexity, have proved to be suitable assumptions for estimating the parameter vector of a GLN distribution (Pierrot and Pinson, 2021), this is not the case anymore when introducing the bound parameter b. Various simulations of $-l_t^{\infty}$ in (5) show the function not to be convex nor pseudoconvex in b, with plateau areas away from the optimal value b^* and steep concave cliffs in the neighborhood of b^*. However, the same simulations also suggest that the function may still have a global minimum. There is a broader class of functions which include convex functions as a subclass and are still unimodal functions: quasiconvex functions. For simplicity let assume functions are differentiable. We use $\|.\|$ to denote the Euclidean norm. From Boyd and Vandenberghe (2010), a definition of quasiconvexity is

Definition 2.1 (Quasiconvexity) *A function $f : \mathbb{R}^d \to \mathbb{R}$ is called quasiconvex (or unimodal) if its domain and all its sublevel sets*

$$S_\alpha = \{\mathbf{x} \in \mathbf{dom}\ f \mid f(\mathbf{x}) \leq \alpha\},$$

for $\alpha \in \mathbb{R}$, are convex.

As an illustrative example, Figure 2 shows the negative pdf of a normal variable which is a quasiconvex function but not a convex function. While quasiconvexity is a considerable generalization of convexity, many of the properties of convex functions hold or have analogs for quasiconvex functions. However, quasiconvexity broadens but does not fully capture the notion of unimodality in several dimensions. This is the argument of Hazan et al. (2015) who introduce *local-quasiconvexity*, a property that extends quasiconvexity and captures

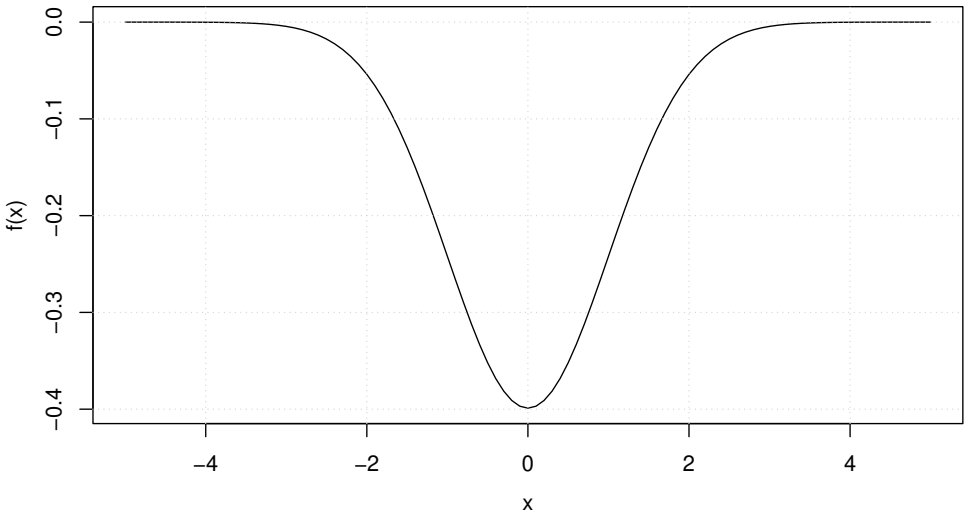

Figure 2: A quasiconvex differentiable function on $\mathbb{R}$, the negative density of a normal variable, with plateau areas when going away from the global minimum.

unimodal functions which are not quasiconvex. Let $\mathbb{B}_d(\mathbf{x}, r)$ denote the d dimensional Euclidean ball of radius r centered around $\mathbf{x}$, and $\mathbb{B}_d := \mathbb{B}_d(0, 1)$. The definition of local-quasiconvexity as introduced by Hazan et al. (2015) is the following:

Definition 2.2 (Local-quasiconvexity) *Let* $\mathbf{x}, \mathbf{z} \in \mathbb{R}^d$, $\kappa, \epsilon > 0$.

We say that $f : \mathbb{R}^d \mapsto \mathbb{R}$ *is* $(\epsilon, \kappa, \mathbf{z})$*-Strictly-Locally-QuasiConvex (SLQC) in* $\mathbf{x}$*, if at least one of the following applies:*

1. $f(\mathbf{x}) - f(\mathbf{z}) \leq \epsilon$.

2. $\|\nabla f(\mathbf{x})\| > 0$, *and for every* $\mathbf{y} \in \mathbb{B}_d(\mathbf{z}, \epsilon/\kappa)$ *it holds that* $\nabla f(\mathbf{x})^\top (\mathbf{y} - \mathbf{x}) \leq 0$.

Hence, we propose to relax the convexity (or pseudoconvexity) assumption and instead rely on local-quasiconvexity when minimizing $-l_t^\infty$.

3 Online maximum likelihood estimation

3.1 Quasiconvex optimization

Let f be the quasiconvex objective function we wish to minimize w.r.t parameter $\mathbf{x} \in \mathbb{R}^d$. It is well known that quasiconvex problems can be solved through a series of convex feasibility problems (Boyd and Vandenberghe, 2010). However, solving such feasibility problems can be very costly and involves finding a family of convex functions $\phi_t : \mathbb{R}^d \to \mathbb{R}$, $t \in \mathbb{R}$, that satisfy

$$f(\mathbf{x}) \leq t \iff \phi_t(\mathbf{x}) \leq 0,$$

and $\phi_s(\mathbf{x}) \leq \phi_t(\mathbf{x})$ whenever $s \geq t$. In the batch setup, a pioneering paper by Nesterov (1984) was the first to propose an efficient algorithm, Normalized Gradient Descent (NGD), and to prove that this algorithm converges to an ϵ-optimal solution within $O(1/\epsilon^2)$ iterations given a differentiable quasiconvex objective function. Gradient Descent (GD) with fixed step sizes is known to perform poorly when the gradients are too small in a plateau area of the function or explode in cliff areas. Among the deep learning community, there have been several attempts to tackle plateaus and cliffs. However, those works do not provide a theoretical analysis showing better convergence guarantees than NGD, which is similar to GD, except one normalizes the gradient. Having introduced SLQC functions, Hazan et al. (2015) prove that NGD also finds an ϵ-optimal minimum for such functions in $O(1/\epsilon^2)$ iterations.

3.2 Online Normalized Gradient Descent

NGD can be used to minimize $-l_t^{\infty}$ w.r.t θ. However, this implies to run NGD every time we want to update the parameter vector θ. Taking advantage of the SLQC assumption, Hazan et al. (2015) present SNGD, which is similar to Stochastic Gradient Descent (SGD) except they normalize the gradients, and prove the convergence of SNGD within $O(1/\epsilon^2)$ iterations to an ϵ-optimal minimum. From the observation that online learning and stochastic optimization are closely related and interchangeable (see, e.g., Cesa-Bianchi et al. (2004) and Duchi et al. (2011)), we use the Stochastic Normalized Gradient Descent (SNGD) introduced by Hazan et al. (2015) and derive the corresponding Online Normalized Gradient Descent (ONGD) for online learning. ONGD is presented in Algorithm 1. To the best of our knowledge, this is the first time SNGD is used in an online learning fashion for Online Quasiconvex Optimization. Following the framework introduced by Cesa-Bianchi

Algorithm 1 Online Normalized Gradient Descent (ONGD)

Input: convex set $\mathcal{K}, T, \theta_m \in \mathcal{K}$, step size η, minibatch size m

for $t = m, \ldots, T$ **do**

Play θ_t and observe cost $f_t(\theta_t) = \dfrac{1}{m} \displaystyle\sum_{j=t-m+1}^{t} f_j(\theta_t)$.

Update and project:

$$\tilde{\theta}_{t+1} = \theta_t - \eta \hat{g}_t \text{ where } g_t = \nabla f_t(\theta_t),\ \hat{g}_t = \frac{g_t}{\|g_t\|}$$

$$\theta_{t+1} = \Pi_{\mathcal{K}}(\tilde{\theta}_{t+1})$$

end for

and Lugosi (2006), we define ONGD in terms of a repeated game played between the online

player and the "environment" generating the outcome sequence. At each time t, we play a parameter vector $\mathbf{x}_t$. After we have committed to this choice, a cost function f_t is revealed and the cost we incur is $f_t(\mathbf{x}_t)$, the value of the cost function for the choice $\mathbf{x}_t$. Similarly to Online Gradient Descent (OGD), which is based on standard gradient descent from offline optimization and was introduced in its online form by Zinkevich (2003), we have included in ONGD a projection step $\Pi_{\mathcal{K}}(.)$. Indeed in each iteration, the algorithm takes a step from the previous point in the direction of the normalized gradient of the previous cost. This step may result in a point outside of the underlying convex set $\mathcal{K}$. In such cases the algorithm projects the point back to the convex set $\mathcal{K}$, i.e. finds its closest point in $\mathcal{K}$. SNGD requires the gradient to be estimated using a minibatch of minimal size m. Indeed, Hazan et al. (2015) provide a negative result showing that if the minibatch size is too small then the algorithm might diverge. This is where SNGD differs from SGD as in the latter and for the case of convex functions even a minibatch of size 1 is enough to ensure convergence.

GD methods update the current value of a parameter θ by taking a step in a descent direction, according to the gradient of the cost function f over a complete set of observations:

$$\hat{\theta}_{i+1} = \hat{\theta}_i - \eta_i \frac{1}{T} \sum_{j=1}^{T} \nabla_\theta f(x_j|\hat{\theta}_i), \tag{6}$$

where i is the current iteration and η_i the *step size* at iteration i. Online/stochastic GD is obtained by dropping the averaging operation in (6) and updating the parameter θ according to

$$\hat{\theta}_{i+1} = \hat{\theta}_i - \eta_i \nabla_\theta f(x_i|\hat{\theta}_i), \tag{7}$$

where x_i has been chosen randomly for SGD and x_i are successive observations for OGD.

The simplification relies on the hope that the random noise introduced by this procedure will not compromise the average behavior of the algorithm. Let $\alpha = 1$ and $-l_t^{\infty}(\theta) = -\frac{1}{t-p}\left[\sum_{j\in C_t(\theta)} \log p_j(\theta) + \sum_{j\in \overline{C_t}(\theta)} \log s_j(b)\right]$. Hence, ONGD can be applied to our framework by taking

$$f_j(\theta_t) := \begin{cases} -\log p_j(\hat{\theta}_t) & \text{if } j \in C_t(\hat{\theta}_t), \\ -\log s_j(\hat{b}_t) & \text{if } j \in \overline{C_t}(\hat{\theta}_t). \end{cases}$$

When minimizing w.r.t $\theta = (\Lambda, \sigma^2, \nu, b)$, we shall recover positive estimates of the scale parameter σ^2 and the shape parameter ν. This is constrained optimization which can be easily overcome by replacing σ^2 with $\omega = \log \sigma^2$ and ν with $\tau = \log \nu$. Such a change of variable allows us to avoid the projection step in Algorithm 1.

3.3 Recursive maximum likelihood estimation

A straightforward competitor to ONGD is a (classic) recursive MLE procedure. While sub-optimal for non convex problems, such a recursive procedure is a quasi-Newton approach which approximates the Hessian with a positive definite matrix thanks to first-order information. Hence, the algorithm is ensured to run although its performance depends on how close is the approximated Hessian to the true Hessian. Recall that $n_\alpha = \frac{1}{1-\alpha}$. We can rewrite (5) as

$$-l_t^{\infty}(\theta) = \begin{cases} -\alpha l_{t-1}^{\infty}(\theta) - (1-\alpha)\log p_t(\theta) & \text{if } t \in C_t(\theta), \\ -\alpha l_{t-1}^{\infty}(\theta) - (1-\alpha)\log s_t(b) & \text{if } t \in \overline{C_t}(\theta). \end{cases} \tag{8}$$

Let now $\hat{\theta}_t$ be the estimate of the parameter vector at time t. The recursive MLE procedure relies on a Newton step for obtaining $\hat{\theta}_t$ as a function of $\hat{\theta}_{t-1}$, see, e.g., Madsen (2007) and

Pinson and Madsen (2012). Let $\mathbf{h}_t = \nabla_\theta \log p_t(\hat{\theta}_{t-1})$ if $t \in C_t(\hat{\theta}_{t-1})$, $\mathbf{h}_t = \nabla_\theta \log s_t(\hat{b}_{t-1})$ if $t \in \overline{C_t}(\hat{\theta}_{t-1})$ and $\hat{\mathbf{R}}_t = -\nabla^2_\theta l^\infty_t(\hat{\theta}_t)$. Our two-step recursive scheme at time t is

$$\hat{\mathbf{R}}_t = \alpha \hat{\mathbf{R}}_{t-1} + (1-\alpha)\mathbf{h}_t\mathbf{h}_t^\top,$$

$$\hat{\theta}_t = \hat{\theta}_{t-1} + (1-\alpha)\hat{\mathbf{R}}_t^{-1}\mathbf{h}_t.$$

An algorithm based upon such a scheme might face computational issues as it requires inverting a matrix, the information matrix $\hat{\mathbf{R}}_t$, at each iteration. This can be prevented by directly working with the matrix inverse, the covariance matrix $\hat{\mathbf{P}}_t$. The resulting algorithm (rMLE.b) is described in Algorithm 2.

Algorithm 2 Recursive Maximum Likelihood Estimation (rMLE.b)

Input: $T, \theta_p \in \mathbb{R}^{p+3}$, forgetting factor $\alpha \in (0,1)$, $\hat{\mathbf{P}}_p = 10^6 \mathbf{I}_{p+3}$

for $t = p+1, \ldots, T$ **do**

Set $\mathbf{h}_t = \nabla_\theta \log p_t(\hat{\theta}_{t-1})$ if $t \in C_t(\hat{\theta}_{t-1})$ or set $\mathbf{h}_t = \nabla_\theta \log s_t(\hat{\theta}_{t-1})$ if $t \in \overline{C_t}(\hat{\theta}_{t-1})$.

Update:

$$\hat{\mathbf{P}}_t = \frac{1}{\alpha}\left[\mathbf{I}_{p+3} - \frac{\hat{\mathbf{P}}_{t-1}\mathbf{h}_t\mathbf{h}_t^\top}{\frac{\alpha}{1-\alpha} + \mathbf{h}_t^\top \hat{\mathbf{P}}_{t-1}\mathbf{h}_t}\right]\hat{\mathbf{P}}_{t-1}$$

$$\hat{\theta}_t = \hat{\theta}_{t-1} + (1-\alpha)\hat{\mathbf{P}}_t\mathbf{h}_t \qquad \text{if } t > T_0 + p$$

end for

The technical derivations and detailed computations required for Algorithms 1 and 2 are available in the supplementary material.

4 Simulation study

We perform an empirical study on synthetic data to check on the behaviour of ONGD, in particular in comparison with a classic recursive MLE procedure, i.e., rMLE.b. We run 100 Monte Carlo (MC) simulations with $T = 12000$, $\lambda = 0.9$, $\sigma^2 = 1$, $\nu = 1.5$. Only the bound parameter b is simulated so that it varies over time. Hence, in Section 4.1, while we test the tracking ability of our algorithms regarding b, we also control their ability to retrieve constant parameters, i.e., λ, σ^2 and ν. Then, in Section 4.2, we introduce the task of issuing predictive distributions using the tracked parameters and evaluate the algorithms depending on the sharpness and calibration of these forecasts.

4.1 Tracking the parameter vector

The lag $p = 1$ of the AR process is assumed to be known and the initial values of the parameter vector $\theta = (\lambda, \sigma^2, \nu, b)$ are set to $(0, 1, 1, 1)$. From a theoretical point of view, these initial values only need to be feasible, i.e., $\sigma_0^2 > 0$, $\nu_0 > 0$ and $b_0 = 1$. We (a priori) choose $\sigma_0^2 = 1$ and $\nu_0 = 1$ because they are standard values. The values of the hyperparameters m and η for ONGD, α for rMLE.b, are empirically selected. We test a grid of values for each hyperparameter and select the values which achieve the best tracking/accuracy trade-off on the first MC simulation. The same values are then used on the 99 remaining MC simulations. These values are listed in Table 1. ONGD runs as soon as m observations are available, while rMLE.b runs from p observations on but needs a warm-up period $T_0 = 150$ to update the covariance matrix $\hat{\mathbf{P}}_t$ before starting updating the parameters. Note that the computational complexity is $\mathcal{O}(Tmp)$ for ONGD and $\mathcal{O}(Tp^2)$

for rMLE.b. Since $p = 1$ and $m = 100$ here, rMLE.b is slightly faster. However, this is not always the case, as we will see when applying the algorithms to wind power forecasting in Section 5 with $p > m$. Overall, because $T >> m$ and $T >> p$, the computational cost is similar for both algorithms.

Table 1: Hyperparameter values for each algorithm: step size η and minibatch size m (Algorithm 1) and forgetting factor α (Algorithm 2)

	η	m	α
Algorithm 1: ONGD	0.001	100	-
Algorithm 2: rMLE.b	-	-	0.975

We provide in Figure 3 the plots of $-l_t^\infty$ for the first MC simulation, $t = 3000$ and $\alpha = 0.975$ w.r.t. each parameter, the other parameters being fixed to their true value. From the function being not convex in b, it is clear that the optimization problem cannot be assumed to be convex when introducing an upper bound parameter b. The tracked parameters averaged over the simulations are presented in Figure 4 along with corresponding MC intervals. ONGD manages to accurately estimate both time-varying and constant parameters while rMLE.b performs well in tracking a decreasing b, but fails to properly track an increasing b. Moreover, this comes with a cost in accuracy which is very high for constant parameters. The estimation of σ^2 and ν is also rather unstable. Note that σ^2 is already at its true value when the algorithms get started and allows us to control that there is no divergence from the optimal value. As a matter of fact and as stated by Hazan

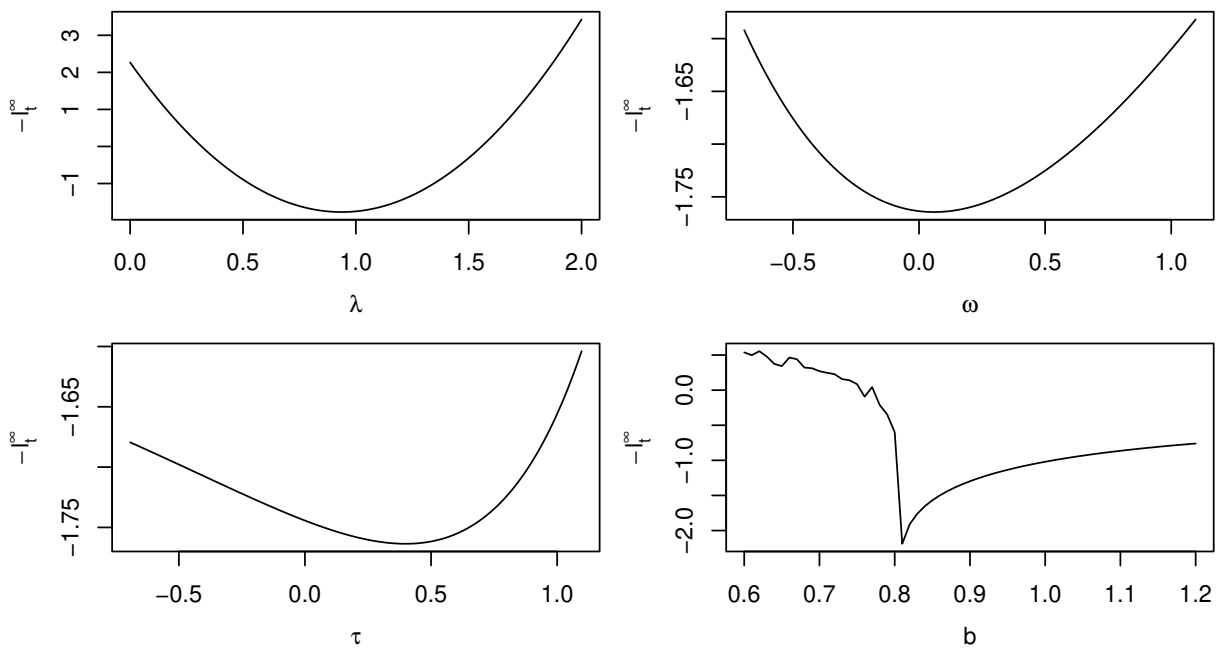

Figure 3: The extended time-dependent negative log-likelihood for the first MC simulation, $t = 3000$, $\alpha = 0.975$, w.r.t. λ (top left), ω (top right), τ (bottom left) and b (bottom right). For each plot, the remaining parameters are set to their true value.

et al. (2015), for ONGD we have observed σ^2 to diverge when the minibatch size m was too small, i.e., for $m < 10$.

4.2 Forecasting the distribution

Because many applications which might benefit from this framework involve forecasting, we are now interested in the performance of the algorithms when forecasting at time t the distribution of the bounded variable X_{t+1}. To be able to track the bound parameter b over time, we have introduced in Section 2 the extended time-dependent negative log-likelihood $-l_t^\infty$ which makes sense from an inference point of view. Because we are

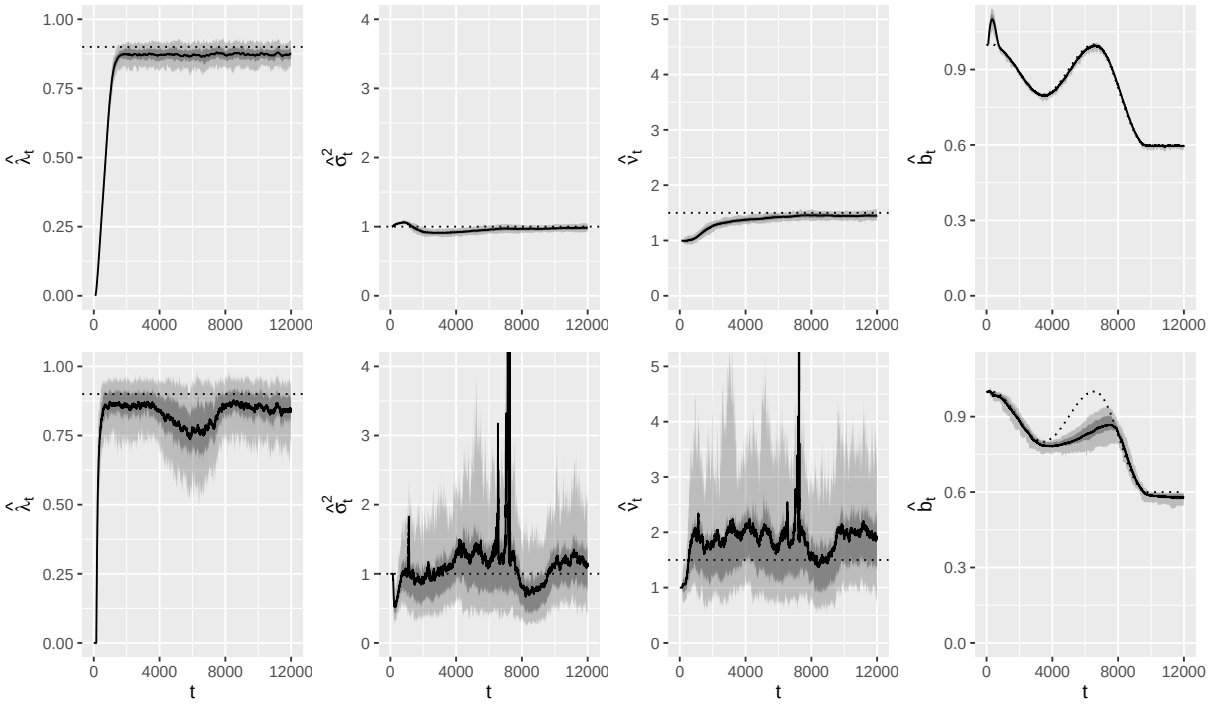

Figure 4: Confidence intervals of the tracked parameters for ONGD (top) and rMLE.b (bottom) with coverage probabilities 0.9 and 0.5, along with the average estimates (solid lines) and the true parameters (dotted lines).

working with series of dependent observations, when moving to forecasting the distribution of X_{t+1}, we need the current value b_t of b to be greater than all the p former observed values of $X_t, \ldots, X_{t-p+1}$ for the expected value of X_{t+1} to exist. Therefore, we introduce a projection step as described in Section 1 for ONGD: we project $\hat{\theta}_t$ on the convex set $\mathcal{K} = \mathbb{R}^{p+2} \times (\max(x_t, \ldots, x_{t-p+1}), +\infty)$ and we get the projected parameter $\tilde{\theta}_t = \Pi_{\mathcal{K}}(\hat{\theta}_t) = (\hat{\Lambda}_t, \hat{\omega}_t, \hat{\tau}_t, \tilde{b}_t)$ where $\tilde{b}_t = \max(x_t, \ldots, x_{t-p+1}) + \delta$ if $\max(x_t, \ldots, x_{t-p+1}) > \hat{b}_t$, $\tilde{b}_t = \hat{b}_t$ if $\max(x_t, \ldots, x_{t-p+1}) < \hat{b}_t$. Note that we need to introduce a small $\delta > 0$ as we project $\hat{b}_t$ on an open convex set. When looking at the observation x_t as a coarsened version of X_t, δ can be seen as a coarsening parameter. This coarsened data framework has been

formalized by Heitjan and Rubin (1991) and Heitjan (1993). We use $\delta = 0.001$.

We evaluate the predictive distributions through proper scoring rules. Scoring rules are attractive measures of predictive performance as they evaluate calibration and sharpness simultaneously. The paradigm of maximizing the sharpness subject to calibration was proposed in Gneiting et al. (2007) under the conjecture that ideal forecasts and the maximization of sharpness subject to calibration are equivalent. Not only scoring rules are measures of both calibration and sharpness. When they are proper for the class of predictive distributions at hand, they also incentivize truthful calibrated and informative forecasts (Holzmann and Eulert, 2014). We use the Continuous Ranked Probability Score (CRPS) which is a proper scoring rule relative to the class $\mathcal{P}$ of the Borel probability measures on $\mathbb{R}$ (Gneiting and Raftery, 2007). Proper scoring rules are often used in negative orientation, e.g., the lower the better. As we work with predictive densities, one could think of using the logarithmic score which is strictly proper relative to all measures that are absolutely continuous. However, in our framework it can always happen that x_{t+1} falls out of the support of the predictive density $\hat{p}_{t+1}$ we have issued at time t, when the current estimate $\hat{b}_t$ of the upper bound is too low. In such a case, the logarithmic score is equal to $-\log \hat{p}_{t+1}(x_{t+1}) = -\log 0 = +\infty$, which is not suitable. In contrast the CRPS is defined on $\mathbb{R}$ as

$$\mathrm{CRPS}(F, x) = \int_{-\infty}^{\infty} \left(F(y) - \mathbb{1}_{y \geq x}\right)^2 \mathrm{d}y, \tag{9}$$

where F is the cumulative distribution function (cdf) of the probabilistic forecast and y is the evaluation point. Let $F := \hat{F}_{t+1}$ and $x := x_{t+1}$. If x_{t+1} happens to be greater than $\hat{b}_t$

we get

$$\begin{aligned}\mathrm{CRPS}(\hat{F}_{t+1}, x_{t+1}) &= \int_{-\infty}^{\hat{b}_t} \left(\hat{F}_{t+1}(y) - \mathbb{1}_{y \geq x_{t+1}}\right)^2 \mathrm{d}y + \int_{\hat{b}_t}^{\infty} \left(1 - \mathbb{1}_{y \geq x_{t+1}}\right)^2 \mathrm{d}y, \\ &= \int_{-\infty}^{\hat{b}_t} \hat{F}_{t+1}(y)^2 \mathrm{d}y + \int_{\hat{b}_t}^{x_{t+1}} 1 \, \mathrm{d}y + \int_{x_{t+1}}^{\infty} 0 \, \mathrm{d}y, \\ &= \int_{-\infty}^{\hat{b}_t} \hat{F}_{t+1}(y)^2 \mathrm{d}y + x_{t+1} - \hat{b}_t.\end{aligned}$$

Hence, the CRPS is increased by an observation falling out of the support of the predictive distribution but to a higher finite value contrary to the logarithmic score which becomes infinite. Moreover, the CRPS allows us to compare discrete and continuous distributions, that is to compare our density-based algorithms to usual benchmarks such as climatology and probabilistic persistence, since if the predictive distribution takes the form of a sample of size N, then the right side of (9) can be evaluated in $\mathcal{O}(N \log N)$ operations (Hersbach, 2000). Climatology is based on all data available at the time of forecasting and probabilistic persistence is the last observed value, which we dress with the past persistence errors. We also evaluate the predictive distributions issued by rMLE.1, i.e., by the usual recursive MLE when the bound is assumed to be fixed and equal to 1 (Pierrot and Pinson, 2021).

We start computing predictive distributions after 2,000 observations. Recall that the hyperparameters were chosen in Section 4.1 upon looking at only the first MC simulation, but the whole simulated time series, i.e., looking at the data we are now computing probabilistic forecasts for. This may be optimistic, even if we only looked at the data from the first MC simulation. In order for our algorithms to not be more optimistic than the benchmarks, we use the hyperparameters for probabilistic persistence and rMLE.1 that give the best CRPS on the first MC simulation. The CRPS are available in Table 2. The

Table 2: 1-step-ahead CRPS and respective improvement over persistence and rMLE.1. The CRPS is averaged over the MC sample, and the standard deviation is also provided.

	mean (sd)	Imp./persist.	Imp./rMLE.1
ideal forecaster	5.78% (0.10)	-	-
climatology	15.26% (0.24)	-	-
probabilistic persistence	6.28% (0.10)	-	-
rMLE.1	6.04% (0.09)	3.77%	-
Algorithm 1: ONGD	5.81% (0.10)	7.52%	3.89%
Algorithm 2: rMLE.b	6.05% (0.17)	3.61%	-0.17%

average CRPS obtained by the ideal forecaster, i.e., the GLN distribution with the true values λ, σ^2, ν, $b_{t-p+1}, \ldots, b_{t+1}$, is 5.78%. Probabilistic persistence performs very well on our synthetic dataset with a CRPS which is already much closer to the ideal forecaster's than climatology's. ONGD, which properly handles the varying upper bound, shows a CRPS which is very close to the ideal forecaster's. Moving from rMLE.1 to ONGD provides a significant improvement, which is similar to moving from probabilistic persistence to rMLE.1. As for ONGD's natural competitor rMLE.b, it does not manage to improve the results of rMLE.1.

To specifically check on probabilistic and marginal calibration, Probability Integral Transform (PIT) histograms and marginal calibration plots are available in the supplementary material.

5 Application to wind power forecasting

Accurately forecasting wind power generation is highly important for the integration of wind energy into power systems. We are interested in very short-term forecasting, that is in lead times of a few minutes, which are not only crucial for transmission system operators to keep the system in balance but also very difficult to improve the forecasts for, especially compared to the simple but very efficient persistence.

5.1 Data description

We have historical data from a large offshore wind farm, Anholt in Denmark, from July 1, 2013 to August 31, 2014. The active power is available for 110 wind turbines at a temporal resolution of every 10 minutes. We scale each time series individually according to the nominal power of the wind turbine, and compute the average generation over the wind farm depending on the number of wind turbines which are available at each time t in order to handle missing values. The response random variable we wish to forecast at time t is $X_{t+1} \in (0, 1)$, the average active power generated by the wind farm at time $t+1$. As stated in Section 4.2, we choose to look at the observation x_t as a coarsened version of X_t with $\delta = 0.001$. Hence, an observation x_t is set to δ if $x_t < \delta$ and to $1 - \delta$ if $x_t > 1 - \delta$ and $x_t \in [\delta, 1 - \delta]$ whereas $X_t \in (0, 1)$.

5.2 Validation setup

We split our dataset into two datasets that we keep separate: a training/cross-validation dataset from July 1, 2013 to March 31, 2014, resulting in 39,450 observations; a test dataset

from April 1 to August 31, 2014, resulting in 22,029 observations. As in Section 4.2, we compare Algorithms 1 and 2 to climatology, probabilistic persistence and rMLE.1. For methods involving hyperparameters, that is all of them but climatology, we choose the hyperparameters upon CRPS-based cross-validation: we run each competing algorithm from the beginning of the training set, issue the corresponding probabilistic forecasts, and select the hyperparameters which give the lowest CRPS on the cross-validation dataset, i.e., data from November 1, 2013 to March 31, 2014. The subsequent hyperparameter values are available in Table 3.

Table 3: Hyperparameter values for each algorithm: order p of the AR process, step size η, minibatch size m and forgetting factor α.

	p	η	m	α
Algorithm 1: ONGD	4	0.03	1	-
Algorithm 2: rMLE.b	5	-	-	0.9982

5.3 Assessment of the probabilistic forecasts

The probabilistic forecasts associated with each algorithm are assessed on the test set. The corresponding CRPS are presented in Table 4. Again, probabilistic persistence improves the CRPS of climatology by a large percentage already and both rMLE algorithms perform similarly. As for ONGD, the improvement over both persistence and rMLE.1 is very significant and much larger than the one observed during the simulation study. The parameters

Table 4: 10-minute-ahead CRPS and respective improvements over probabilistic persistence and rMLE.1.

	CRPS	Imp./persist.	Imp./rMLE.1
climatology	22.03%	-	-
probabilistic persistence	1.35%	-	-
rMLE.1	1.08%	19.89%	-
Algorithm 1: **ONGD**	**0.89%**	**34.22%**	**17.89%**
Algorithm 2: rMLE.b	1.06%	21.83%	2.43%

**Best forecast bolded.*

tracked by rMLE.1 and Algorithms 1and 2 are plotted for some sub-sample of the test set in Figure 5. We plot the projection $\tilde{b}_t$ of $\hat{b}_t$, see Section 4.2, to display the upper bound which is actually used for prediction. The estimates of the parameter vector Λ are consistent between ONGD and rMLE.b, while more noisy for ONGD. The parameter estimates from rMLE.1 and rMLE.b are in general very close to one another and show similar patterns. Indeed, rMLE.b does not manage to track a varying upper bound since the estimated $\hat{b}_t$ does not vary significantly away from 1. Only ONGD captures variations of the bound parameter b below 1. When selecting the hyperparameters, we observed a significant improvement on the cross-validation set for a minibatch size $m = 1$ only, the values tested for being $m \in \{1, 5, 10, 20, 50, 100, 150\}$). By looking at the parameter estimates, we noticed that for $m = 5$ the bound estimate was also significantly varying below 1, but more slowly

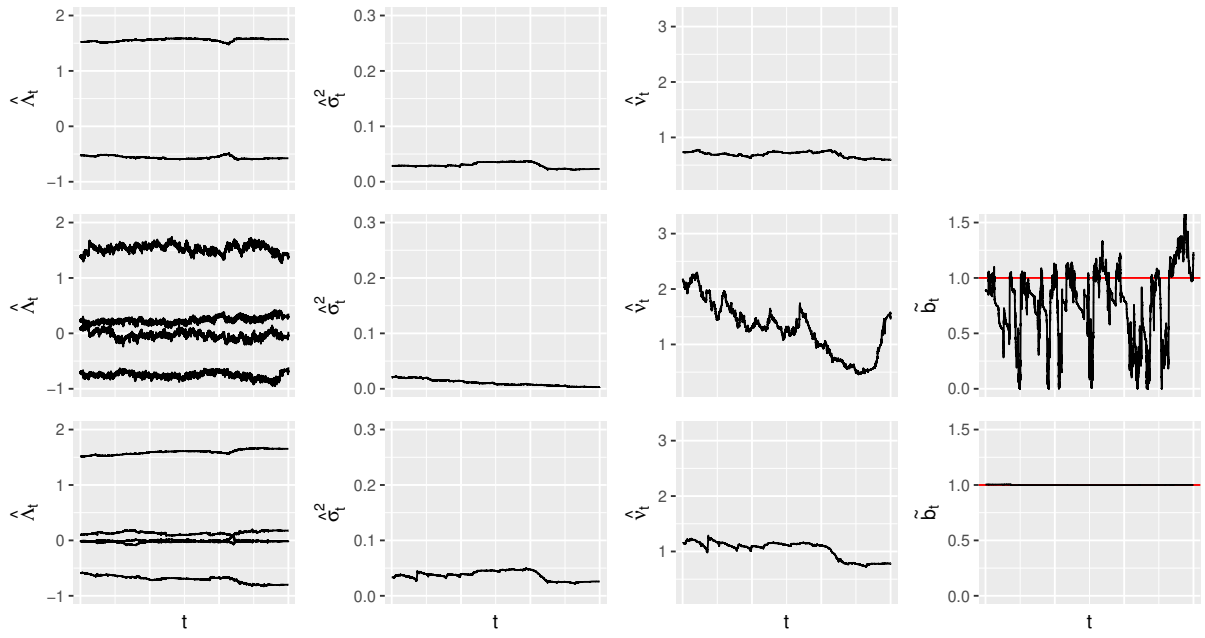

Figure 5: Estimates $\hat{\Lambda}_t$, $\hat{\sigma}^2_t$, $\hat{\nu}_t$ and projected estimate $\tilde{b}_t$ on a sub-sample of the test set for rMLE.1 (top), ONGD (center) and rMLE.b (bottom).

and closer to 1. When looking at the power generation itself, the bound tracked by ONGD for $m = 1$ does make sense. This result and the generalization performance of the model are confirmed on the test set, since the CRPS we get is very close to the one we got on the cross-validation set. Therefore, it seems that these data call for a very aggressive choice of m, so that the algorithm can track the bound. In return some noise is introduced into the other parameter estimates.

In Figure 6, we provide the probabilistic forecasts of ONGD over a 36-hour period of time on the test set. One can note that the prediction intervals are tight. When looking at probabilistic calibration, the PIT histogram shows a too large number of very low and very high PIT values, which suggests that the predictive distributions issued by ONGD

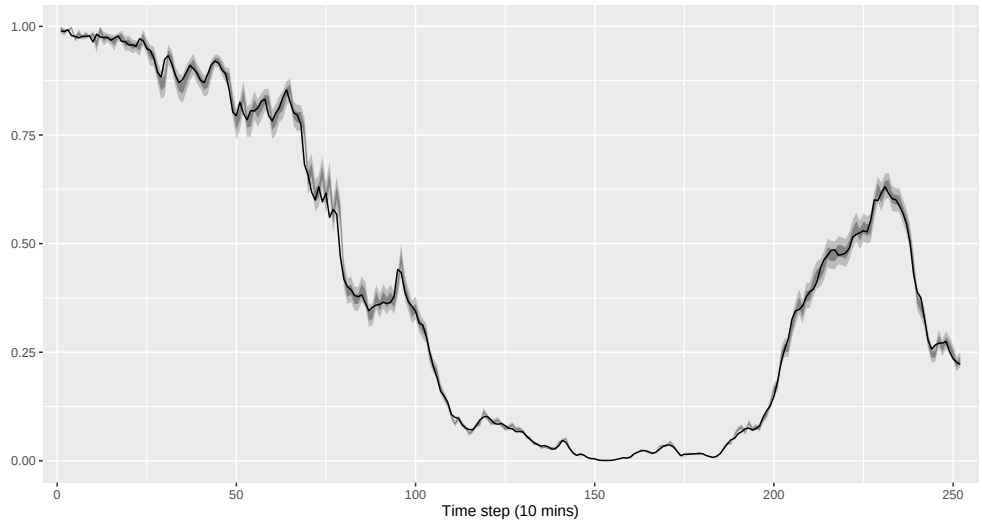

Figure 6: Probabilistic forecasts from ONGD, based on prediction intervals with nominal coverage rates of 95 and 75%, along with power measurements (solid black line).

are underdispersed. However, when an observation falls outside of the prediction interval, it does not fall far away, which explains the very good CRPS achieved by ONGD overall. PIT histograms and marginal calibration plots for all algorithms are again available in the supplementary material.

6 Discussion

We have introduced a framework where where we aim to track varying bounds for bounded time series as well as an extended time-dependent negative log-likelihood to deal with this new framework. As the objective functions now at hand are not convex anymore, we have proposed to use the broader local-quasiconvexity assumption through an online algorithm. To compete with this quasiconvexity framework and online algorithm, we have also proposed a more usual recursive MLE procedure. We have run these algorithms on

both a synthetic and a real dataset to track the parameters of a time series distribution over time, including the upper bound of the support of the distribution. Then, we have presented how to use such a new framework for forecasting.

The algorithm we propose in the new framework we have introduced is an online algorithm which is directly derived from SNGD and so, similarly to "ordinary" OGD, only relies on the negative log-likelihood we observe at time t for our current set of parameters. It does not longer require any kind of forgetting action and only asks for the usual step size when updating the parameter vector through the gradient at time t. A new hyper-parameter which is related to a specificity of SNGD is the size m of the minibatch, since SNGD is not guaranteed to converge for $m = 1$, unlike SGD. This algorithm performs very well on our simulated examples, with performances very close to the ideal forecaster. When moving to wind power forecasting, it improved the CRPS of probabilistic persistence by more than 30% on the test set. However, the predictive distributions do not achieve probabilistic calibration, as the prediction intervals appear to be too narrow in general. It is worth noting that ONGD required to set the minibatch size m to 1 on the wind power generation dataset in order to be able to track the upper bound. This is quite aggressive and suggests that this kind of data may call for methods which can handle big jumps in the bound values.

We challenged ONGD with two competitors, that both rely on recursive MLE: rMLE.1 and rMLE.b. The former operates in the usual framework when the upper bound is assumed to be fixed to 1, while the latter is as straightforward adaptation of rMLE.1 in the context of an upper varying bound. These algorithms showed similar performances on both synthetic

and real data, rMLE.b failing to properly track an upper varying bound. Since this is, to the best of our knowledge, the first time such a framework with varying bounds has been studied, further research could explore different assumptions, e.g., on the stochastic process, on the distribution of the bounded variable, or a lower bound.

SUPPLEMENTARY MATERIAL

Calculation details: Technical derivations and detailes computations required for Algorithms 1 and 2. (.pdf file)

Additional plots: PIT histograms and marginal calibration plots for all competing algorithms in the simulation study and the application to wind power forecasting. (.pdf file)

R project: R project with the R scripts for the simulation study in section 4, along with the corresponding synthetic data. All outputs necessary for the study can be reproduced with the corresponding scripts and are also provided. The structure and content of the R project is described in a README file. (.zip file)

References

Bacher, P., Madsen, H., and Nielsen, H. A. (2009), "Online Short-Term Solar Power Forecasting," *Solar Energy*, 83, 1772–1783.

Boyd, S. P. and Vandenberghe, L. (2010), *Convex Optimization*, Cambridge University Press.

Cesa-Bianchi, N., Conconi, A., and Gentile, C. (2004), "On the Generalization Ability of On-Line Learning Algorithms," *IEEE Transactions on Information Theory*, 50, 2050–2057.

Cesa-Bianchi, N. and Lugosi, G. (2006), *Prediction, Learning, and Games*, Cambridge University Press.

Cotgreave, P. and Clayton, D. H. (1994), "Comparative Analysis of Time Spent Grooming by Birds in Relation to Parasite Load," *Behaviour*, 131, 171–187.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977), "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, 39, 1–38.

Duchi, J., Hazan, E., and Singer, Y. (2011), "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization," *Journal of Machine Learning Research*, 12, 2121–2159.

Frederic, P. and Lad, F. (2008), "Two Moments of the Logitnormal Distribution," *Communications in Statistics - Simulation and Computation®*, 37, 1263–1269.

Gneiting, T., Balabdaoui, F., and Raftery, A. E. (2007), "Probabilistic Forecasts, Calibration and Sharpness," *Journal of the Royal Statistical Society. Series B (Methodological)*, 69, 243–268.

Gneiting, T. and Raftery, A. E. (2007), "Strictly Proper Scoring Rules, Prediction and Estimation," *Journal of the American Statistical Association*, 102, 359–378.

Guolo, A. and Varin, C. (2014), "Beta Regression for Time Series Analysis of Bounded Data, with Application to Canada Google® Flu Trends," *The Annals of Applied Statistics*, 8, 74–88.

Hazan, E., Levy, K. Y., and Shalev-Shwartz, S. (2015), "Beyond Convexity: Stochastic Quasi-Convex Optimization," in *Advances in Neural Information Processing Systems*, volume 28.

Heitjan, D. F. (1993), "Ignorability and Coarse Data: Some Biomedical Examples," *Biometrics*, 49, 1099–1109.

Heitjan, D. F. and Rubin, D. B. (1991), "Ignorability and Coarse Data," *The Annals of Statistics*, 19, 2244–2253.

Hersbach, H. (2000), "Decomposition of the Continuous Ranked Probability Score for Ensemble Prediction Systems," *Weather and Forecasting*, 15, 559–570.

Holzmann, H. and Eulert, M. (2014), "The role of the information set for forecasting—with applications to risk management," *The Annals of Applied Statistics*, 8, 595 – 621.

Johnson, N. L. (1949), "Systems of Frequency Curves Generated by Methods of Translation," *Biometrika*, 36, 149–176.

Laderman, J. and Littauer, S. B. (1953), "The Inventory Problem," *Journal of the American Statistical Association*, 48, 717–732.

Madsen, H. (2007), *Time Series Analysis*, Chapman & Hall.

Mead, R. (1965), "A Generalised Logit-Normal Distribution," *Biometrics*, 21, 721–732.

Nesterov, Y. E. (1984), "Minimization Methods for Nonsmooth Convex and Quasiconvex Functions." *Matekon*, 29, 519–531.

Pierrot, A. and Pinson, P. (2021), "Adaptive Generalized Logit-Normal Distributions for Wind Power Short-Term Forecasting," in *2021 IEEE Madrid PowerTech*, Institute of Electrical and Electronics Engineers Inc.

Pinson, P. (2012), "Very-Short-Term Probabilistic Forecasting of Wind Power with Generalized Logit-Normal Distributions," *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 61, 555–576.

Pinson, P. and Madsen, H. (2012), "Adaptive Modelling and Forecasting of Offshore Wind Power Fluctuations with Markov-Switching Autoregressive Models," *Journal of Forecasting*, 31, 281–313.

Wallis, K. F. (1987), "Time Series Analysis of Bounded Economic Variables," *Journal of Time Series Analysis*, 8, 115–123.

Warton, D. I. and Hui, F. K. C. (2011), "The arcsine is asinine: the analysis of proportions in ecology," *Ecology*, 92, 3–10.

Zinkevich, M. (2003), "Online Convex Programming and Generalized Infinitesimal Gradient Ascent," in *Proceedings of the 20th International Conference on Machine Learning*.

Additional plots

1 Introduction

To empirically assess probabilistic calibration, we provide Probability Integral Transform (PIT) histograms in Figures 1 and 3. To empirically assess marginal calibration, we provide marginal calibration plots which show the difference between the predictive and the empirical cumulative distribution functions in Figures 2 and 4. Probabilistic calibration is reflected through a uniform histogram and marginal calibration through the proximity between the predictive and the empirical cumulative distribution functions. Note that we have removed climatology from Figure 4 since the corresponding predictive cumulative distribution function is too far on average from the empirical one for the difference to be plotted on the same graph as the other methods'.

2 Simulation study

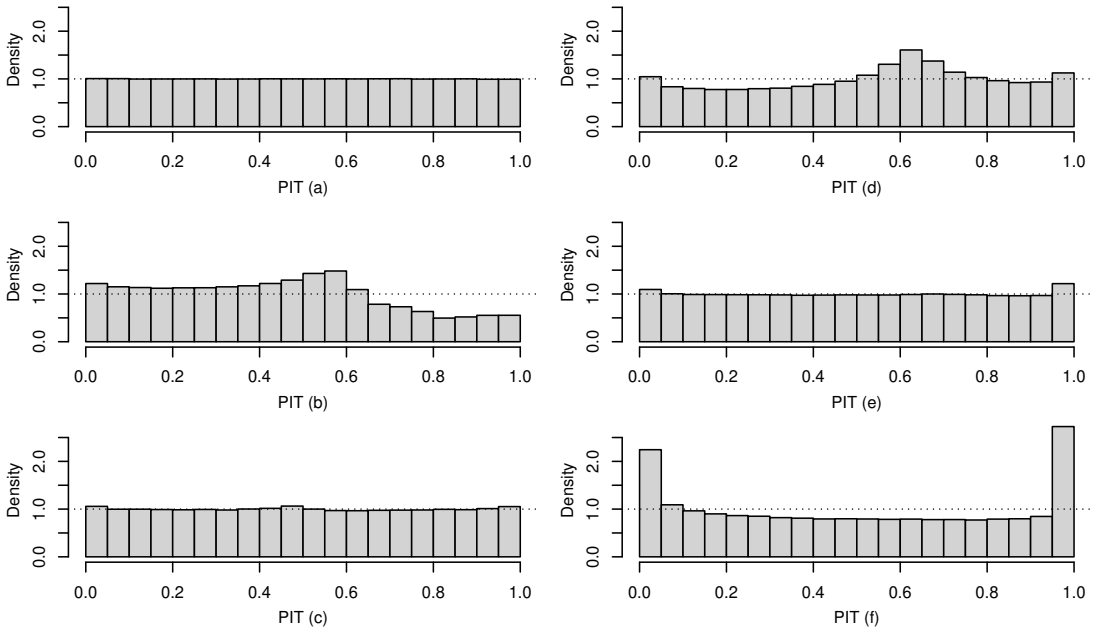

Figure 1: PIT histograms for all benchmarks and Algorithms 1 and 2: (a) ideal forecaster, (b) climatology, (c) probabilistic persistence, (d) rMLE.1, (e) ONGD, (f) rMLE.b.

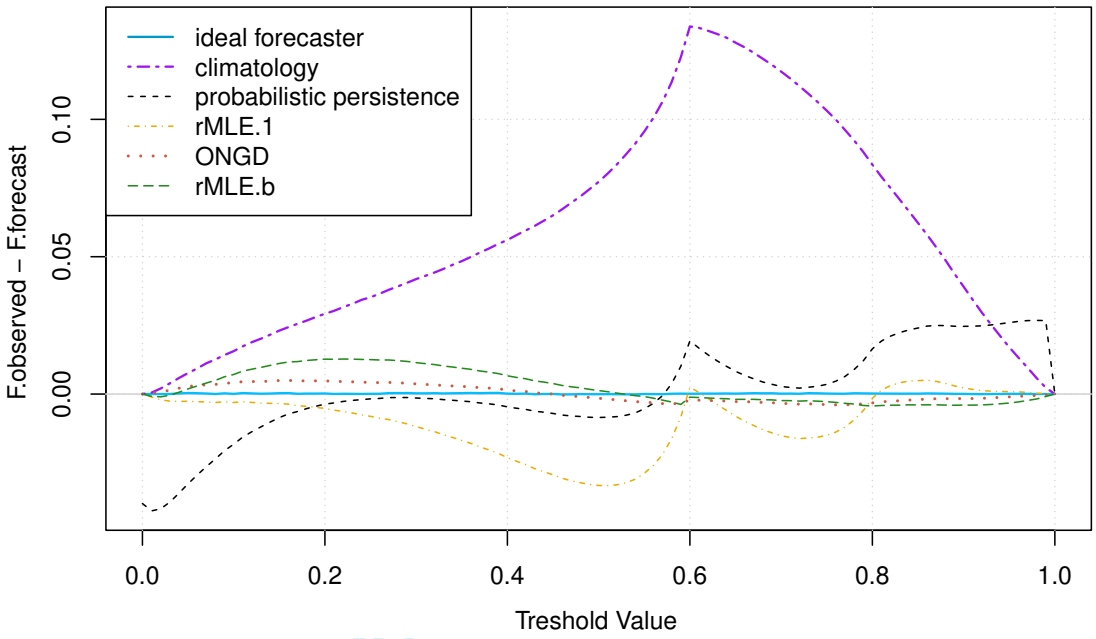

Figure 2: Marginal calibration plot for all benchmarks and Algorithms 1 and 2.

3 Application to wind power forecasting

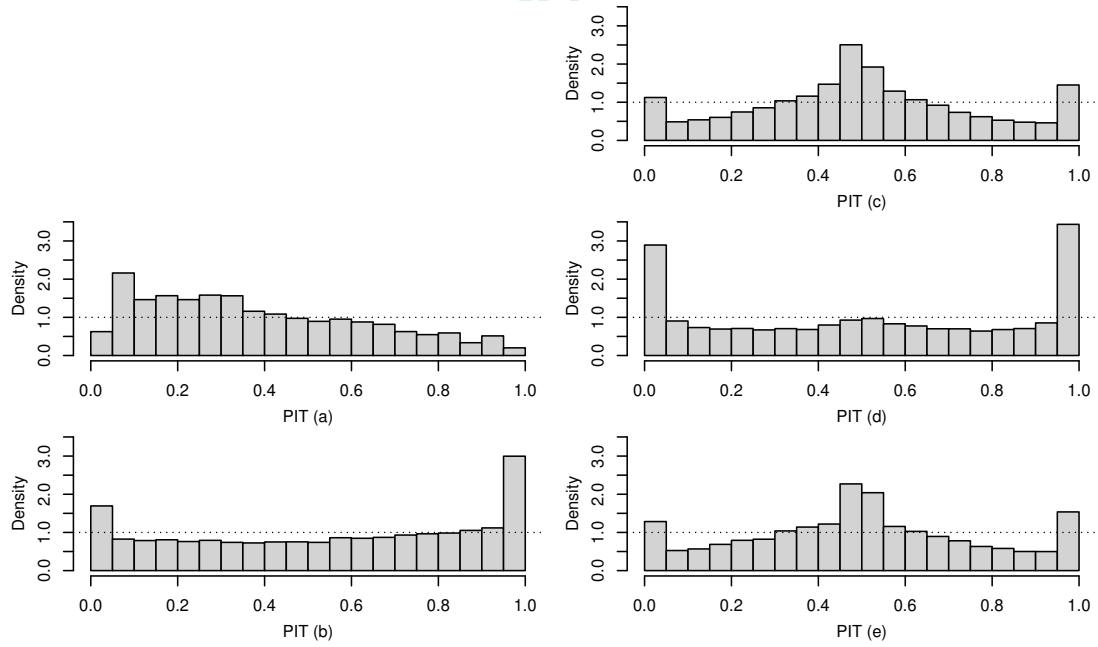

Figure 3: PIT histograms for all benchmarks and Algorithms 1 and 2: (a) climatology, (b) probabilistic persistence, (c) rMLE.1, (d) ONGD, (e) rMLE.b.

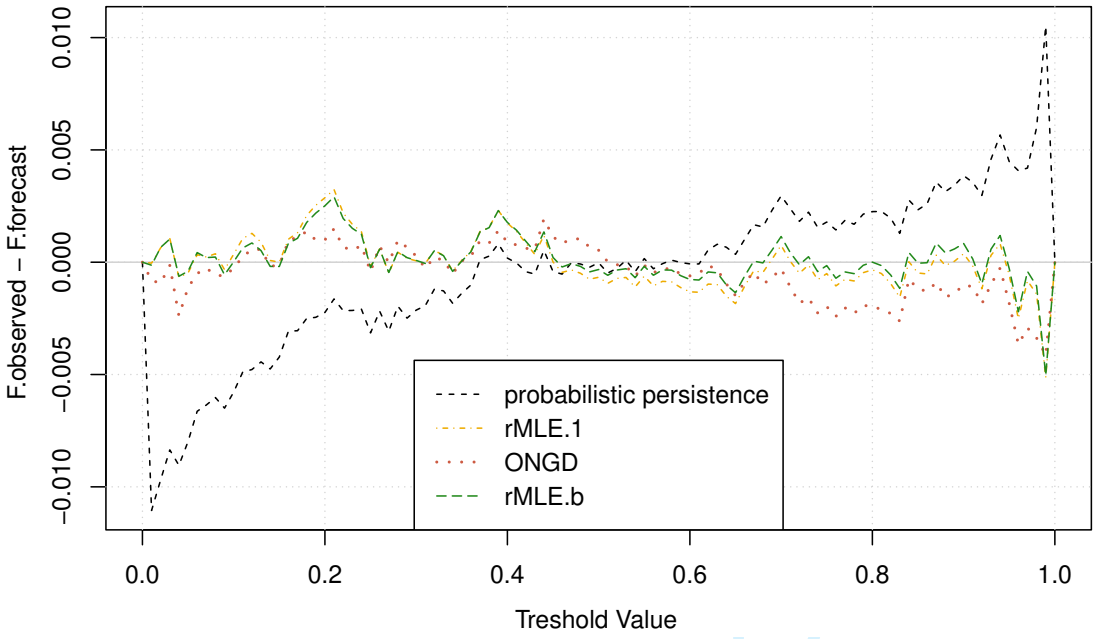

Figure 4: Marginal calibration plot for probabilistic persistence, rMLE.1 and Algorithms 1 and 2.

Calculation details

1 Algorithm 1: ONGD

In this section,
$\theta = (\Lambda, \omega, \tau, b) = (\Lambda, \log\sigma^2, \log\nu, b) \in \mathbb{R}^{p+3}$,
$U_t^m = \{t-m+1, \ldots, t\}$,
$C_t^m(\theta) = \{j \in U_t^m \mid x_{j-k} < b, k = 0, \ldots, p\}$, $\overline{C_t^m}(\theta) = \{j \in U_t^m | j \notin C_t^m(\theta)\}$,
$\mathbf{y} = (y_j) \in \mathbb{R}^{|C_t^m(\theta)|}$, where $y_j = \gamma(x_j/b; \nu)$ and $j \in C_t^m(\theta)$,
$\mathbf{Y}$ is a matrix with columns $B\mathbf{y}, B^2\mathbf{y}, \ldots, B^p\mathbf{y} \in \mathbb{R}^{|C_t^m(\theta)| \times p}$, where B is the backshift operator,
C is a constant which does not depend on θ.

1.1 Objective function

$$
\begin{aligned}
f_t(\theta) &= \frac{1}{m}\left[\sum_{j\in C_t^m(\theta)} \log p_j(\theta) + \sum_{j\in\overline{C_t^m}(\theta)} \log s_j(b)\right], \\
&= \frac{1}{m}\left(\frac{\omega}{2} - \tau\right)|C_t^m(\theta)| - \frac{1}{m}\sum_{j\in C_t^m(\theta)} \log\left(1 - (x_j/b)^{\exp\tau}\right) \\
&+ \frac{1}{m}\frac{1}{2\exp\omega}(\mathbf{y} - \mathbf{Y}\Lambda)^\top(\mathbf{y} - \mathbf{Y}\Lambda) + \frac{1}{m}\sum_{j\in\overline{C_t^m}(\theta)} \log(1 + \exp(-b + x_j)) \\
&+ C.
\end{aligned}
$$

1.2 Gradient

First derivative w.r.t. Λ

$$\frac{\partial f_t}{\partial \Lambda} = -\frac{1}{m\exp\omega}\,\mathbf{Y}^\top(\mathbf{y} - \mathbf{Y}\Lambda).$$

First derivative w.r.t. ω

$$\frac{\partial f_t}{\partial \omega} = \frac{1}{2m}|C_t^m(\theta)| - \frac{1}{2m\exp\omega}(\mathbf{y} - \mathbf{Y}\Lambda)^\top(\mathbf{y} - \mathbf{Y}\Lambda).$$

First derivative w.r.t. τ

$$\frac{\partial f_t}{\partial \tau} = -\frac{1}{m}|C_t^m(\theta)| - \frac{\exp\tau}{m}\sum_{j\in C_t^m(\theta)} \exp y_j \log(x_j/b) + \frac{1}{m\exp\omega}\,(\mathbf{u} - \mathbf{U}\Lambda)^\top(\mathbf{y} - \mathbf{Y}\Lambda),$$

where $\mathbf{u} = \dfrac{\partial \mathbf{y}}{\partial \tau}$, $u_j = \exp\tau \dfrac{\log(x_j/b)}{1 - (x_j/b)^{\exp\tau}}$.

First derivative w.r.t. b

$$\frac{\partial f_t}{\partial b} = \frac{1}{m}\frac{\exp\tau}{b}\sum_{j\in C_t^m(\theta)}\exp y_j + \frac{1}{m\exp\omega}\,(\mathbf{z}-\mathbf{Z}\Lambda)^\top(\mathbf{y}-\mathbf{Y}\Lambda) - \frac{1}{m}\sum_{j\in\overline{C_t^m}(\theta)}\frac{\exp(-b+x_j)}{1-\exp(-b+x_j)},$$

where $\mathbf{z} = \dfrac{\partial \mathbf{y}}{\partial b}$, $z_j = -\dfrac{\exp\tau}{b\,(1-(x_j/b)^{\exp\tau})}$.

2 Algorithm 2: rMLE.b

In this section,
$\theta = (\Lambda, \omega, \tau, b) = (\Lambda, \log\sigma^2, \log\nu, b) \in \mathbb{R}^{p+3}$,
$\mathbf{y} = (y_{t-1}, \ldots, y_{t-p}) \in \mathbb{R}^p$, $y_j = \gamma(x_j/b;\nu)$.

2.1 Gradient

$$\mathbf{h}_t = \begin{cases} \nabla_\theta \log p_t(\hat\theta_{t-1}) & \text{if } t \in C_t(\theta), \\ \nabla_\theta \log s_t(\hat b_{t-1}) & \text{if } t \in \overline{C_t}(\theta). \end{cases}$$

First derivatives w.r.t. Λ

$$\frac{\partial \log p_t}{\partial \Lambda} = \frac{1}{\exp\omega}\,(y_t - \Lambda^\top\mathbf{y})\mathbf{y},$$
$$\frac{\partial \log s_t}{\partial \Lambda} = 0.$$

First derivatives w.r.t. ω

$$\frac{\partial \log p_t}{\partial \omega} = -\frac{1}{2} + \frac{1}{2\exp\omega}(y_t - \Lambda^\top\mathbf{y})^2,$$
$$\frac{\partial \log s_t}{\partial \omega} = 0.$$

First derivatives w.r.t. τ

$$\frac{\partial \log p_t}{\partial \tau} = 1 + \exp\tau\exp y_t\log(x_t/b) - \frac{1}{\exp\omega}(u_t - \Lambda^\top\mathbf{u})(y_t - \Lambda^\top\mathbf{y}),$$
$$\frac{\partial \log s_t}{\partial \tau} = 0,$$

where $\mathbf{u} = \dfrac{\partial \mathbf{y}}{\partial \tau}$, $u_t = \exp\tau\dfrac{\log(x_t/b)}{1-(x_t/b)^{\exp\tau}}$.

First derivatives w.r.t. b

$$\frac{\partial \log p_t}{\partial b} = -\frac{\exp\tau}{b}\exp y_t + \frac{1}{\exp\omega}(z_t - \Lambda^\top\mathbf{z})(y_t - \Lambda^\top\mathbf{y}),$$
$$\frac{\partial \log s_t}{\partial b} = \frac{\exp(-b+x_t)}{1+\exp(-b+x_t)},$$

where $\mathbf{z} = \dfrac{\partial \mathbf{y}}{\partial b}$, $z_t = \dfrac{\exp\tau}{b\,(1-(x_j/b)^{\exp\tau})}$.

2.2 Detailed computation of the matrix $\hat{\mathbf{P}}_t$

We use the matrix inversion rule

$$[\mathbf{A}+\mathbf{BCD}]^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}\left[\mathbf{DA}^{-1}\mathbf{B}+\mathbf{C}^{-1}\right]^{-1}\mathbf{DA}^{-1},$$

with $\mathbf{A} = \alpha\hat{\mathbf{R}}_{t-1}$, $\mathbf{B} = \mathbf{D}^\top = \mathbf{h}_t$, $\mathbf{C} = (1-\alpha)\mathbf{I}$ and we get

$$\begin{aligned}
\hat{\mathbf{P}}_t &= \hat{\mathbf{R}}_t^{-1}, \\
&= \left[\alpha\hat{\mathbf{R}}_{t-1} + (1-\alpha)\mathbf{h}_t\mathbf{h}_t^\top\right]^{-1}, \\
&= \left(\alpha\hat{\mathbf{R}}_{t-1}\right)^{-1} - \left(\alpha\hat{\mathbf{R}}_{t-1}\right)^{-1}\mathbf{h}_t\left[\mathbf{h}_t^\top\left(\alpha\hat{\mathbf{R}}_{t-1}\right)^{-1}\mathbf{h}_t + ((1-\alpha)\mathbf{I})^{-1}\right]^{-1}\mathbf{h}_t^\top\left(\alpha\hat{\mathbf{R}}_{t-1}\right)^{-1}, \\
&= \frac{1}{\alpha}\hat{\mathbf{P}}_{t-1} - \frac{1}{\alpha^2}\hat{\mathbf{P}}_{t-1}\mathbf{h}_t\left[\frac{1}{\alpha}\mathbf{h}_t^\top\hat{\mathbf{P}}_{t-1}\mathbf{h}_t + \frac{1}{1-\alpha}\right]^{-1}\mathbf{h}_t^\top\hat{\mathbf{P}}_{t-1}, \\
&= \frac{1}{\alpha}\hat{\mathbf{P}}_{t-1} - \frac{\hat{\mathbf{P}}_{t-1}\mathbf{h}_t\mathbf{h}_t^\top\hat{\mathbf{P}}_{t-1}}{\alpha^2\left[\frac{1}{\alpha}\mathbf{h}_t^\top\hat{\mathbf{P}}_{t-1}\mathbf{h}_t + \frac{1}{1-\alpha}\right]}, \\
&= \frac{1}{\alpha}\left[\mathbf{I} - \frac{\hat{\mathbf{P}}_{t-1}\mathbf{h}_t\mathbf{h}_t^\top}{\frac{\alpha}{1-\alpha} + \mathbf{h}_t^\top\hat{\mathbf{P}}_{t-1}\mathbf{h}_t}\right]\hat{\mathbf{P}}_{t-1}.
\end{aligned}$$