Recency-Weighted Temporally-Segmented Ensemble for Time Series Modeling

PÅL V. B. JOHNSEN*, EIVIND BØHN, SØLVE EIDNES, FILIPPO REMONATO, and SIGNE RIEMER-SØRENSEN, SINTEF Digital, Norway

Time series modelling in process industries faces the challenge of dealing with complex, multi-faceted and evolving data characteristics. Conventional single-model approaches often struggle to capture the interplay of diverse dynamics, resulting in suboptimal forecasts. Addressing this, we introduce the Recency-Weighted Temporally-Segmented (ReWTS, pronounced 'roots') ensemble model, a novel chunk-based approach for multi-step forecasting. The key characteristics of the ReWTS method are twofold: 1) It facilitates specialization of models into different dynamics by segmenting the training data into 'chunks' of data and training one model per chunk; 2) During forecasting, an optimization procedure assesses each model on the recent past and selects the active models, such that the appropriate mixture of previously learned dynamics can be recalled to forecast the future. This method not only captures the nuances of each period, but also adapts more effectively to changes over time compared to conventional 'global' models trained on all data in one go. We present a comparative analysis, using two years of data from a wastewater treatment plant and a drinking water treatment plant in Norway, demonstrating the ReWTS ensemble's superiority. It consistently outperforms the global model in terms of mean squared forecasting error across various model architectures by 10-70% on both datasets, notably exhibiting greater resilience to outliers. We further explore the generalizability of ReWTS by applying it to four publicly available datasets. The results indicate that ReWTS is particularly valuable in non-stationary, heterogeneous environments with frequent concept drift, as exemplified by the process industry datasets. This approach shows promise in developing automatic, adaptable forecasting models for decision-making and control systems in process industries and other complex systems.

JAIR Associate Editor: Kuldeep Meel

JAIR Reference Format:
Pål V. B. Johnsen, Eivind Bøhn, Sølve Eidnes, Filippo Remonato, and Signe Riemer-Sørensen. 2025. Recency-Weighted Temporally-Segmented Ensemble for Time Series Modeling. *Journal of Artificial Intelligence Research* 84, Article 12 (October 2025), 41 pages. DOI: 10.1613/jair.1.17572

1 Introduction

Process industries are concerned with the processing of raw materials into products. To achieve the desired quality, this processing must be controlled based on the conditions in which it takes place. For example, for a chemical process to run optimally, the chemical reactions can be controlled through chemical dosing.

The processing system is often complex and consists of several interconnected stages. The system is also often regulated by closed-loop controllers that, given feedback from the system, make decisions in order to control the state of the system (Bequette 2010). An example of a closed-loop controller is the proportional-integral-derivative (PID) controller (Åström and Hägglund 2001). However, this type of control is solely reacting based on

*Corresponding Author.

Authors' Contact Information: Pål V. B. Johnsen, ORCID: 0000-0002-2599-7914, pal.johnsen@sintef.no; Eivind Bøhn, ORCID: 0000-0002-0676-3688; Sølve Eidnes, ORCID: 0000-0002-1002-3543, solve.eidnes@sintef.no; Filippo Remonato, ORCID: 0000-0002-1988-6230; Signe Riemer-Sørensen, ORCID: 0000-0002-5308-7651, signe.riemer-sorensen@sintef.no, SINTEF Digital, Oslo, Norway.

DOI: 10.1613/jair.1.17572

the present situation, and does not support incorporating future information such as forecasts. An alternative control technique called model predictive control (MPC) is applicable if one can account sufficiently for future consequences of different actions. To employ MPC, a requirement is to have a *forecasting model* that can predict the state of the system in the near future as a function of the control inputs with sufficient accuracy (Schwenzer et al. 2021). The forecasting model can be entirely based on known physical or chemical system properties, which entails that certain assumptions need to be made. It can also be entirely data-driven or model-free based on historical measurements, with no inherited information about known causal, physical or chemical properties in the system (Nauman et al. 2022). A hybrid model is a mix of the two types, combining both data-driven methods, as well as scientific and causal knowledge about the process.

Causal knowledge can come in many forms. It can be the awareness of intentional changes such as maintenance of equipment, often carried out at regular time intervals. One example of this is the flushing of filters in the drinking water treatment plant considered in Section 5.4.2. The process can also be affected by external factors such as weather conditions, and other external factors such as seasons may introduce periodic patterns in the process. At the same time, the process relies on situation-specific occurrences that vary stochastically, or can be intractable to determine in advance. This can for instance be clogging of pipes or sudden weather changes.

The corresponding stochastic processes of the variables governing the industrial process do not typically exhibit *stationarity* (constant mean and covariance over time). Trends and seasonalities are often seen in the time series, but not necessarily in a consistent, deterministic way, rendering transformations to stationarity non-trivial. Another complicating factor is that many variables in the process are often statistically correlated. Consequently, we deal with a *multivariate*, *correlated*, *non-stationary*, and partly *seasonal* time series. In this context, it is crucial that the forecasting model can adapt to new patterns, while also being able to recognize previously seen patterns. In this work, we present a strategy for reaching this goal, in particular for data-driven forecasting models. The idea is to construct models for disjoint historical regions and combine them in such a way that previously seen patterns in the data, implicitly learned by the models, can be captured efficiently at prediction time.

In Section 2 we formulate the problem, introduce notation and list some assumptions. In Section 3 we discuss proposed solutions from the literature and explain why they are insufficient. Our novel method is given in Section 4, before we test it on simulated and real data in Section 6. Section 7 provides a discussion of the strengths and weaknesses of the method and concluding remarks.

2 Problem Description, Assumptions and Notation

Given multivariate time series data describing a certain process, we consider the aim of forecasting the future behaviour of the target variable y. For each time point, along with a measurement from y, there are also measurements from K variables, hereon denoted covariates. For simplicity, assume the target and the K covariate measurements are consistently available or otherwise estimated at equidistributed time points. Let $t_0 = 0$ be the first available recording time of all measurements, while t_n denotes the present time point. The time interval between two consecutive time points is denoted Δt. We let $\mathbf{y}_{a:b}$ be the target vector of size $b - a + 1$, and $X_{a:b}$ the covariate matrix of dimension $(b - a + 1) \times K$ that include all the measurements between $t = t_a$ and $t = t_b$. At any prediction time, t_n, we want to construct a *forecasting model* that can predict the target measurements between $t = t_{n+1}$ and $t = t_{n+h}$, where h denotes the *forecast horizon*.

Let the vector $\hat{\mathbf{y}}_{(n+1):(n+h)}$ denote the h predicted future measurements at time t_n of the target variable from a particular forecasting model as a function of historical data of both the target and the covariates. In some cases, the future values of the K covariates up to time t_{n+h}, included in the matrix $X_{(n+1):(n+h)}$, are known. An example relevant in process industries is where a covariate represents planned operations or control signals such as chemical dosing, or when filters are flushed. In this case, the forecasts of the target can also be a function of these already-known future covariate values.

Assume we have historical data available including known values of the target and a corresponding trained forecasting model. We want to evaluate the performance of the forecasting model with respect to the historical data. At prediction time, $t = t_n$, the performance of the forecasting model given particular forecast values $\hat{\mathbf{y}}_{(n+1):(n+h)}$ can be quantified by a similarity measure, also called a *loss function*, $L(\mathbf{y}, \hat{\mathbf{y}})$. For simplicity, we will in this work focus on the mean squared error (MSE) as the loss function:

$$L\left(\mathbf{y}_{(n+1):(n+h)}, \hat{\mathbf{y}}_{(n+1):(n+h)}\right) = \frac{1}{h}\sum_{i=1}^{h}(y_{n+i} - \hat{y}_{n+i})^2. \tag{1}$$

Given all historical data from t_f up to time t_e, denoted $X_{f:e}$ and $\mathbf{y}_{f:e}$, we want to evaluate the h-step forecasts from the forecasting model not only at a single prediction time point, but by considering its predictions on the entire span of predictions between $t = t_f$ to $t = t_e$. This can be achieved by averaging all forecast loss values for individual time stamps from $t = t_f$ to $t = t_e$ with a *stride* s (the number of time steps between successive forecasts).

$$L_s(\mathbf{y}_{f:e}, \hat{\mathbf{y}}_{(f+1):(f+h)}, \ldots, \hat{\mathbf{y}}_{(f+\psi s+1):(f+\psi s+h)}) = \tfrac{1}{\psi}\textstyle\sum_{k=0}^{\psi} L\left(\mathbf{y}_{(f+ks+1):(f+ks+h)}, \hat{\mathbf{y}}_{(f+ks+1):(f+ks+h)}\right), \tag{2}$$

with $\psi = \lfloor (e - h - f)/s \rfloor$.

3 Related Work

The challenge of continuously training a machine learning model with new incoming data streams is described in McCloskey and Cohen (1989) where the term *catastrophic forgetting* is introduced. This describes the situation where a model by learning relationships from new data, undesirably forgets previously learned relationships. The consequence is that even though it would be intuitive and practical to update a machine learning model trained on historical data with a batch of the most recent data, it may cause the model to forget previously observed relationships. The alternative is to retrain the model from scratch, including both old and new data.

An alternative to a single model that needs tuning or complete retraining for new incoming data, is to have several models trained from different time intervals, and appropriately combine them to give the final prediction. Hence, this can be seen as a particular type of *ensemble model*. The idea is that the different models are trained on different dynamics, so that each model is specialized on a certain subset of the global dynamics. At forecasting time, the present dynamics might be a combination of several dynamics previously seen by the models, such that a specific weighted combination or *ensemble* of the models may give a reasonable forecast.

In Opitz and Maclin (1999), it is empirically shown that ensemble models most often, but not always, outperform a single model for a large number of tabular datasets when dealing with classification problems. However, in this case every base model in the ensemble model is trained on the same underlying data. A comprehensive survey of ensemble learning with data streams is provided in Krawczyk et al. (2017). In this survey, the authors distinguish between *online learning* and *chunk-based learning*. Online learning approaches process each new incoming sample sequentially, for instance by updating the model at every incoming sample. In chunk-based learning, incoming data is processed in chunks, for instance by fitting a separate model on each chunk. It is further distinguished between *stationary* and *non-stationary* data streams. In the second case, the data-generating process is not stable in terms of constant mean and covariance, but actually changing over time. We say the data exhibits *concept drift* in the form of gradual changes in the data-generating process. This is a typical example for process industries.

A typical framework for chunk-based models is to first define a chunk length, adaptively train a model for every chunk of data, and finally combine all chunk models into an ensemble model to predict for new data.

Exactly how to combine all chunk models into one ensemble model does not have a definitive answer, and there exist several proposals for this. One proposal described in Wang et al. (2003) for classification models, is to assign weights to each chunk model based on its performance on the newest training chunk. The assumption is that the last training chunk is the most representative of the new incoming data. Here, the weight for each chunk model is computed based on a classification-based mean squared error on the most recent training chunk.

There are also several approaches dealing with how to discard previous chunk models if they are underperforming. One approach is presented in Wang et al. (2003) where classification chunk models are discarded if they are performing worse than a random classification model. However, as observed in McCloskey and Cohen (1989), this pruning strategy might be too strict in the situation of sudden concept drifts, as is typical for process industry data.

In the realm of non-stationary data stream analysis, numerous ensemble techniques have been developed to address classification tasks (Brzezinski and Stefanowski 2014; Elwell and Polikar 2011; Street and Kim 2001). The exploration of ensemble methods for multi-step forecasting has been less explored at the time of writing. This is in particular true for non-stationary data streams. In Galicia et al. (2019), a heterogeneous ensemble model for multi-step forecasting is introduced, however not presented explicitly in the context of continuous data streams, but with evaluation on static future test data. Moreover, this is not a chunk-based method, but rather only the most recent data of a fixed size is used to construct the ensemble model at prediction time. The ensemble model is a weighted sum of a decision tree model, a gradient boosting model and a random forest model trained on the most recent data, denoted as the training set. The weights are calculated such as to minimize the mean squared error of the forecasts on the last portion of the training set, denoted the validation set. The authors further introduce an updating period for when to recalculate the weights based on the most recent historical data at prediction time.

4 The Recency-Weighted Temporally-Segmented Ensemble

Here we consider a chunk-based ensemble model to be a model consisting of multiple models (be it domain-based, data-driven, or a combination) that are combined in some way that involves weighting of the models, to produce a single forecast. The models in the ensemble may be of the same type (homogeneous ensembles), or of different types (heterogeneous ensembles). An example of a homogeneous ensemble is a random forest, where all the models are decision trees. For the chunk-based ensemble model we propose here in the context of time series forecasting, each single model is of the same type, but trained on different time intervals. In the following, we consider only the most naive approach for choosing these intervals, which is to split the timeline into disjoint equidistant sets. However, we note that varying length or overlapping intervals could be other alternatives.

There are several ways the models in an ensemble can be weighted, but here we suggest a linear combination scheme designed to overcome practical challenges in the process industry. Let the matrix Z_n characterize the dynamics of the data at time t_n, given by recent historical data from a specified historical time point up to the present time point t_n. Assume we have the set $\mathcal{M}_n$ of already trained forecasting models at time t_n, and let the matrix $M_h(X_{:n}, \mathbf{y}_{:n})$ of dimension $h \times |\mathcal{M}_n|$ include the forecasts for the h following time steps for each model. Then, the h-step ahead expected forecast can be written as

$$E_{\mathcal{M}}[\hat{\mathbf{y}}_{(n+1):(n+h)}] = M_h(X_{:n}, \mathbf{y}_{:n})\mathbf{p}_{\mathcal{M}_n}(\mathbf{Z}_n), \tag{3}$$

where the vector $\mathbf{p}_{\mathcal{M}_n}(\mathbf{Z}_n)$ of size $|\mathcal{M}_n|$ can be interpreted as containing the probabilities that each model $m_j \in \mathcal{M}_n$ accounts for the true dynamics in the data described by $\mathbf{Z}_n$. From hereon we will denote $\mathbf{p}_{\mathcal{M}_n}(\mathbf{Z}_n)$ as the *weights* $\mathbf{w}(t_n)$ at time t_n. Interpreting the weights as probabilities implies the constraints

$$w_j(t_n) \geq 0 \quad \forall\, j, n,$$
$$\sum_{j=1}^{\mathcal{M}_{t_n}} w_j(t_n) = 1.$$

The question then becomes in what way the weights should be estimated at every time point t_n, to support the goal of providing robust forecasts. A particular characterization of the dynamics in the data, $\mathbf{Z}_n$, is required for this purpose and will set the stage for how the weights are calculated. Below, we suggest one weight assignment procedure by letting $\mathbf{Z}_n$ include the most recent historical data from a predefined historical time point and up to time t_n. We call this the look-back data.

4.1 The Near Past as an Indication of the Near Future

Based on an idea from Wang et al. (2003) for classification problems, the weights of the models at prediction time t_n can be estimated by looking at the dynamics of the data in the near past of time t_n. Intuitively, the best-performing weight assignment of the models on recent data is likely also applicable for data in the near future. From the present time point t_n, let $\mathbf{Z}_n = (X_{(n-l_b):n}, \mathbf{y}_{(n-l_b):n})$ denote the look-back data going l_b time steps back in time. We seek the linear combination of previously trained models that minimizes the mean squared error of the h step ahead forecasts across the look-back data. Translating this to a mathematical optimization problem, we search for the weights $w_j(t_n)$, for $j = 1, \ldots, |\mathcal{M}_n|$, at time t_n for the ensemble of the models in $\mathcal{M}_n$ that solve the following optimization problem:

$$\begin{aligned}
&\arg\min_{\mathbf{w}(t_n)} \sum_{k=n-l_b}^{n-h} \left(\mathbf{y}_{(k+1):(k+h)} - M_h\left(X_{:k}, \mathbf{y}_{:k}\right)\mathbf{w}(t_n)\right)^T \left(\mathbf{y}_{(k+1):(k+h)} - M_h\left(X_{:k}, \mathbf{y}_{:k}\right)\mathbf{w}(t_n)\right) \\
&= \arg\min_{\mathbf{w}(t_n)} \left[\frac{1}{2}\mathbf{w}(t_n)^T \left(\sum_{k=n-l_b}^{n-h} M_h(X_{:k}, \mathbf{y}_{:k})^T M_h(X_{:k}, \mathbf{y}_{:k})\right)\mathbf{w}(t_n)\right. \\
&\quad \left. - \left(\sum_{k=n-l_b}^{n-h} M_h(X_{:k}, \mathbf{y}_{:k})^T \mathbf{y}_{(k+1):(k+h)}\right)^T \mathbf{w}(t_n)\right] \qquad (4) \\
&\text{s.t.} \quad \mathbf{w}(t_n) \geq 0 \\
&\qquad \mathbf{1}^T \cdot \mathbf{w}(t_n) = 1.
\end{aligned}$$

In fact, one can show that this is a so-called *convex quadratic programming* optimization problem, for which efficient solvers exist (Nocedal and Wright 2006). In particular, we apply the quadratic programming solver provided by the CVXOPT Python package (Andersen et al. 2013). See Appendix A for more details. Notice that the mean squared error is computed across the look-back dataset with a stride equal to one ($s = 1$). The weights that solve (4) are denoted by $\hat{\mathbf{w}}(t_n)$. The predicted h-step-ahead forecasts at time t_n is given by $M_h(X_{:n}, \mathbf{y}_{:n})\hat{\mathbf{w}}(t_n)$.

Since the ensemble is based on models trained on different time segments, and the weighting is based on the most recent dynamics, we call the method the *Recency-Weighted Temporally-Segmented* (ReWTS) ensemble method. As opposed to what is done in Wang et al. (2003), the weights are not computed separately for each chunk model. Instead, by the construction of the optimization problem in 4, the best-performing chunk models on the look-back data will be assigned the largest weights implicitly due to the constraint that the weights need

to sum to one. The ReWTS ensemble is arguably more similar to the ensemble model described in Galicia et al. (2019). However, here, all the models are trained on the same data, specifically restricted to the most recent data. Hence, in the context of data streams, previous relationships are disregarded, and only the most recent data is considered relevant. This approach risks forgetting or never learning about recurring events and cyclic patterns, which are often observed in industrial processes.

Like the Mixture of Experts (MoE) model (Masoudnia and Ebrahimpour 2014), our model facilitates model specialization and employs a weighting strategy during inference to determine the most relevant subset of models for a given input. However, in our approach, specialization is achieved through explicit segmentation of training data, whereas MoE employs a learned gating network to direct inputs to specific experts. At inference, MoE uses this gating network to activate relevant experts based on current input, whereas ReWTS employs a structured optimization process informed by recent historical data to select operative models. Unlike MoE, which primarily aims to enhance computational efficiency, ReWTS is designed to improve forecasting accuracy in complex time series data.

4.2 The Entire Forecasting Process

Consider a defined model architecture with fixed hyperparameters for each chunk model, chunk length l_c, look-back length l_b, and forecast horizon h. Let t_n be the present time point, and assume that chunk c has just finished (i.e. n/l_c is integer) with c trained models $\mathcal{M}_n = \{M_1, \ldots, M_c\}$. For ease of notation, and to avoid any further complexity, we will assume that the time it requires to train each model is less than the time interval between two consecutive time points, Δt. With a continuous flow of new data, the automatic forecasting process to be employed in practice can be described as given in Algorithm 1.

Algorithm 1 The h step forecasting procedure - ReWTS method

▷ Given forecast horizon h, look-back length l_b, chunk length l_c, stride s and present time point t_n, with n/l_c an integer
▷ Assign $C = n/l_c$, the number of available chunks
▷ Assign $v = n$
▷ Given all previously trained models, $\mathcal{M}_{t_v} = \{M_1, ..., M_C\}$, from the C previous chunks
repeat
  while $v \leq (C+1) \cdot l_c$ **do**
    ▷ Collect look-back data $(X_{(v-l_b):v}, \mathbf{y}_{(v-l_b):v})$
    ▷ Calculate the weights $\hat{\mathbf{w}}(t_v)$ according to optimization problem (4) given chunk models in $\mathcal{M}_v$
    ▷ Compute h-step forecasts from time t_v for all c chunk models, and save in matrix $M_h(X_{:v}, \mathbf{y}_{:v})$
    ▷ Compute final forecasts given by $M_h(X_{:v}, \mathbf{y}_{:v})\hat{\mathbf{w}}(t_v)$ at present time point t_v
    ▷ $v = v + s$
  end while
  ▷ Collect all data from new chunk number $C+1$ between time points $t_{C \cdot l_c}$ and $t_{(C+1) \cdot l_c}$
  ▷ Train new chunk model M_{C+1} from chunk $C+1$ and update $\mathcal{M}_v$
  ▷ $C = C + 1$
until Collection of new data stops

It is important to have in mind that the data-generating process can change over time, and thus be different from chunk to chunk. Before any chunk model is fitted, we scale the chunk using standardization of the chunk mean and standard deviation. When using these chunk models in the ensemble, we scale the inputs to each model independently by the standardization parameters of the corresponding chunk.

Figure 1 provides a schematic overview of the proposed ReWTS ensemble model approach using a look-back dataset.

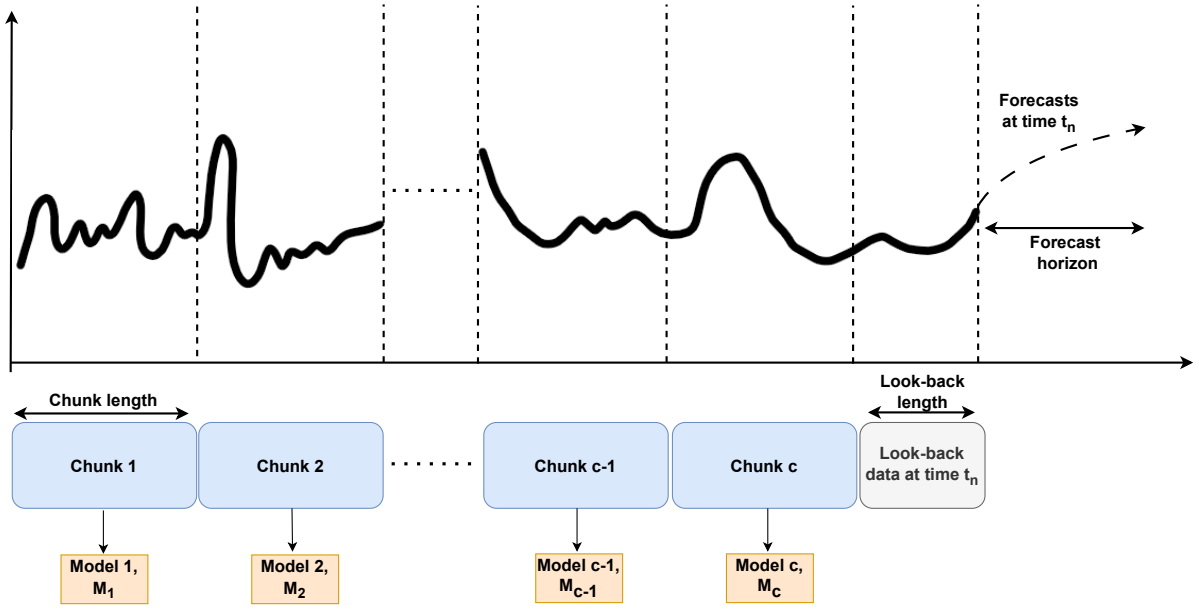


Fig. 1. Conceptual illustration of the look-back, chunk-based approach. The figure shows how the models trained on disjoint chunks of times series are used to provide forecasts of the target response at time t given look-back data.

5 Methodology

In this section we outline the procedure for testing and evaluating the ReWTS ensemble model. The source code for this work is openly available at https://github.com/SINTEF/rewts.

5.1 Architectures and Hyperparameters

The implementation of the ReWTS ensemble model is integrated within the Python package Darts (Herzen et al. 2022). In this way, one can easily experiment with several built-in model architectures. In this work, each model in the ReWTS ensemble model will be of the same machine learning architecture, and trained on separate disjoint time intervals of equal size as depicted in Figure 1. However, we test various architectures including neural networks, tree-based models, and linear regression.

There are several choices to be made regarding hyperparameters that must be in place before deploying the automatic forecasting process of the ReWTS ensemble model. This includes model-specific hyperparameters (such as learning rate, hidden layers, etc.) as well as chunk length and look-back length. Our software package leverages the Hydra framework (Yadan 2019), enhancing modularity and flexibility to streamline configuration management and enable rapid experimentation. It further features a pipeline using the Optuna package for hyperparameter optimization (Akiba et al. 2019). To investigate the ReWTS ensemble model, we employ it on generated simulation data, as well as based on real process industry data.

5.2 Baseline Model - The Global Model

For both the simulated data and the process industry data, we compare the ReWTS ensemble model with what we call a *global model*, which means that one single forecasting model is trained on all historical data at once.

The automatic forecasting process for the global model is described in Algorithm 2 given a model architecture and predefined hyperparameters. This means that the global model is retrained from scratch for every new chunk of data. The smaller the chunk length, the more often the global model is updated.

Algorithm 2 The h step forecasting procedure - Global method

▷ Given forecast horizon, h, chunk length, l_c, stride s and present time point t_n with n/l_c an integer
▷ Given previously trained global model, M_f using all historical data up to time t_n
▷ Assign $v = n$
▷ Assign $C = n/l_c$, the number of available chunks
repeat
  while $v \leq (C+1) \cdot l_c$ **do**
    ▷ Compute h step forecasts from global model given by $M_f(X_{:v}, \mathbf{y}_{:v})$ at present time point t_v
    ▷ $v = v + s$
  end while
  ▷ Collect all data from new chunk number $C+1$ between time points $t_{C \cdot l_c}$ and $t_{(C+1) \cdot l_c}$
  ▷ Update global model M_f from scratch using all historical data including new data from chunk $C+1$
  ▷ $C = C + 1$
until Collection of new data stops

The baseline model has the same overall architecture as the individual chunk models in the ReWTS ensemble. However, in the pursuit of fair evaluation, when possible we upscale the number of trainable parameters in the global model to be similar to that of the total ensemble model. See details in Section 5.4.4. The initial hyperparameter optimization (HPO) before deployment of the automatic forecasting process is based solely on the global model. The resulting fitted hyperparameter values from the global model are used directly to assign hyperparameter values for each chunk model in the ReWTS ensemble. More details on this follow in Section 5.3 and 5.4.

5.3 Simulation Data

To demonstrate the potential advantages of the ReWTS ensemble method and identify the conditions where it proves beneficial, we create a synthetic dataset. The synthetic dataset has distinct data-generating processes, in terms of sinusoidal functions, in non-overlapping consecutive regions or chunks. Hence, each chunk has its unique data pattern, creating well-defined transitions between them. This design choice allows us to assign one model per chunk, ensuring each model in the ReWTS ensemble is trained exclusively on data from one data-generating process.

The synthetic dataset is made up of 16 chunks each consisting of 500 data points sampled from a sine wave ($A\sin(\omega t)$), where the amplitudes and frequencies are given in Table 1, while making sure the time series is continuous between all chunks. We denote the first eight chunks to be in the train dataset, and the rest in the test dataset.

Our hypothesis posits that the ReWTS ensemble model in these circumstances should perform better in total than the corresponding global model. This is because each chunk model in the ReWTS ensemble model can specialize in one specific data-generating process, while the global model will focus its effort on learning the

Table 1. Parameters used to generate the various chunks of the synthetic dataset.

Dataset	Parameter	Chunk							
		#1	#2	#3	#4	#5	#6	#7	#8
Train	Amplitude A	0.5	2	20	2	2	0.5	2	5
	Frequency ω	10	2	5	1	0.5	8	3	1
Test	Amplitude A	0.75	10	3	0.5	5	1.25	3	4
	Frequency ω	8	0.75	7	11	0.65	4	2	5

chunks with dominating dynamics (i.e. in this case the largest amplitudes). This is advantageous in these specific chunks, yet it yields less focus to generalize for all chunks.

As an interesting feature: When forecasting with the ReWTS ensemble model, the weight assignment procedure should ideally allocate all weight to the model trained on the same data-generating process for which we are currently forecasting, assuming the occurrence of the data-generating process in question has previously taken place.

For each chunk in the train dataset, we train one forecasting model until convergence on a hold-out validation set accounting for the last 25% of the chunk. We then construct a ReWTS ensemble model of the resulting eight models. The look-back length is set to $l_b = 160$ time steps. The forecasts of the ReWTS ensemble model at any point will then be a linear combination of these eight models, where the corresponding weights are calculated as given in Algorithm 1. The ReWTS ensemble's performance on the train and test datasets is compared against the global model that is trained simultaneously on all eight chunks in the training set. For both the ReWTS ensemble model and the global model, a recurrent neural network (RNN) of type Long-short-term memory (LSTM) (Staudemeyer and Morris 2019) is applied with an input length of 80 time steps (ensuring at least one full period of the preceding sine wave), and equal hyperparameter values such as optimizer (Adam), batch size, early stopping and max epochs. The difference lies in the dimension space of the hidden LSTM layer. As first mentioned in Section 5.2, to compare the ReWTS ensemble with a global model with comparable learning capacities, we ensure the global model has as many learnable parameters as the total number of parameters in the ReWTS ensemble model. Therefore, each chunk model is set to have a single LSTM layer of dimension 10, yielding ≈ 500 learnable parameters. With eight chunk models, this implies the global model has around eight times more parameters than each chunk model, 4000 in total, resulting in an LSTM layer of dimension 32.

We deploy the automatic forecasting process for the chunk model and the global model as described in respectively Section 4 and 5 on the train data as well as the test data with unseen sinusoidal functions. The forecasts are evaluated based on their ability to forecast 30 time steps ahead for every 30th time step (stride of 30). We evaluate the performance of both methods in each chunk separately using the loss function given in (2), for all forecasts between $[t_{i \cdot l_c}, t_{(i+1) \cdot l_c}]$, with $t_{i \cdot l_c}$ denoting first time step of chunk i. To make the performances on the different chunks comparable, we further scale by the amplitude of the corresponding sine wave in the chunk. Separately from the forecasting evaluations explained above, we will also investigate the behaviour of the two approaches in the transitions between the chunks where there is a sudden shift in the data-generating process.

5.4 Process Industry Data

We investigate the performance of the ReWTS ensemble method on nearly two years of observed data from two water treatment plants in Norway.

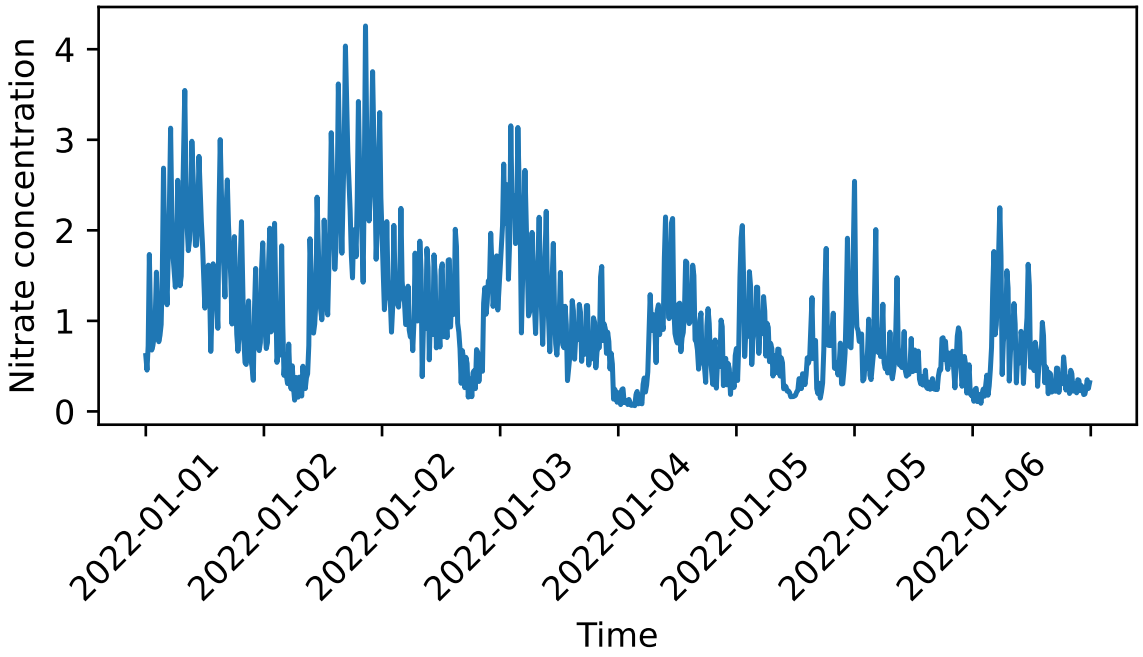


Fig. 2. Wastewater data: A small time segment of the first six days in January 2022, showing how the nitrate concentration varies in time. The time resolution is 10 minutes.

5.4.1 Wastewater Treatment Data. The Veas wastewater treatment plant in Slemmestad is responsible for processing more than half of the wastewater originating from the greater Oslo region. They have provided data for January 2022 to October 2023 for the denitrification process where facultative anaerobic bacteria naturally present in the wastewater convert nitrate into dinitrogen. These bacteria respire nitrate in low-oxygen conditions. The conversion is achieved using a carbon and electron source, which for the particular wastewater treatment plant is methanol. Removing nitrogen is one of the main objectives of the wastewater treatment plant, and accurate modelling of the nitrate concentration out of the process is important for the optimal dosage of methanol. We focus on modelling forecasts of the nitrate concentration in a single hall consisting of four filters in parallel. The historical data is resampled to have a frequency of 10 minutes between every measurement, which will also be the basis when developing the forecasting models which gives 94 988 data points in total. Figure 2 shows a typical trend of the nitrate concentration after passing the denitrification hall as a function of time. One can observe a large non-stationary fluctuation of the nitrate concentration. The measurements are naturally noisy, as the nitrate concentration is measured in a basin cycling water from the four filters, and these filters can be in different states. The use case for this dataset is to provide forecasts for the nitrate concentration after being processed in the denitrification hall at intervals of 10 minutes, spanning up to four hours into the future with a stride of four hours. Based on domain knowledge from the operators and the availability of data, the model input features are: previously recorded measurements of the nitrate concentration before entering the denitrification hall, the flow of methanol, and the nitrate concentration in the wastewater after being processed in the denitrification hall. Prior to the construction of the forecasting models, measurements of the nitrate concentration out of the

hall during a small time interval of around two days (288 data points) were removed due to severe measurement errors. These measurements were replaced by linearly interpolated values.

5.4.2 Drinking Water Treatment Data. Bergen Vann is the city of Bergen's drinking water producer. There we focus on one filter out of several that cleans drinking water from January 2022 to July 2023. The target is to model the turbidity of the drinking water, a measure of how cloudy the water is, which is measured in Formazin Nephelometric Units. The aim is to provide forecasts at intervals of 10 minutes spanning up to four hours into the future with a stride of four hours. Based on domain knowledge from the operators and availability of data, the input features are: previously recorded measurements of the drinking water volume produced per time unit, amount of raw water entering the treatment plant per time unit, pH, temperature in the water, conductivity, chemical dosing of ferric chloride, as well as measurements connected to regular flushing of the filter, among them flushing water amount, backwash air, maturation rate and maturation time. The raw data, which is sampled at a one-minute time resolution, is resampled to a frequency of 10 minutes which gives 78 768 data points in total.

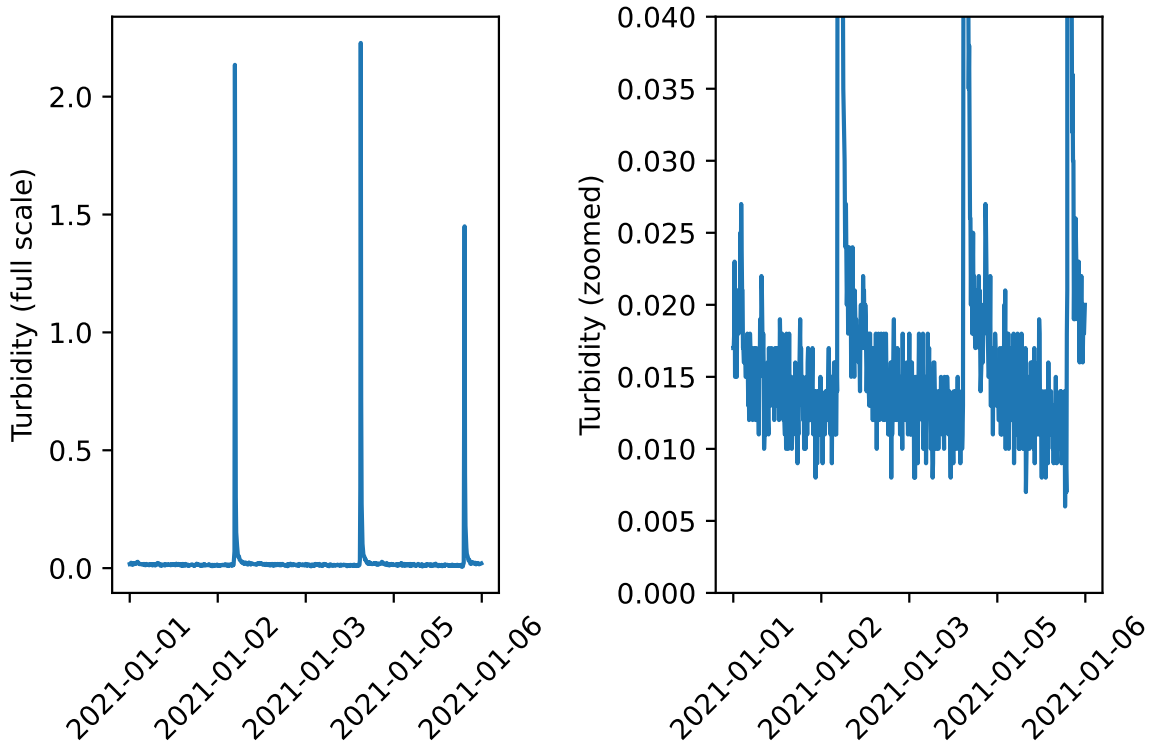


Fig. 3. Drinking water data. Left panel: A small time segment of the first six days in January 2021 showing how the turbidity varies in time. The peaks indicate when the filter is flushed. Drinking water is produced between the peaks. The time resolution is 10 minutes. Right panel: Measurements for the same time segment, but zoomed in to see the dynamics of the turbidity during drinking water production.

Between mid-September and mid-November in 2022, there is a period with measurement errors in the turbidity due to planned maintenance, in particular renewal of filtration media. This is a typical and inevitable scenario

in process industries when operating online. Replacing these measurements using interpolation, as is done for the wastewater treatment data, will be too inaccurate for such a long period. Simply removing this period from the dataset is a simple option that will make sure the models are not affected by them. However, in an online prediction scenario such as this, removal of incorrect measurements would need to be automated which is not a simple task in general. Instead, we leave the incorrect measurements in the data to explore to what degree the global method and the ReWTS ensemble method are affected by them, particularly in the period after the measurement errors are gone and the data-generating process is back to its normal state.

A visualization of a typical trend of the turbidity as a function of time is given in Figure 3, where one can observe regular spikes in the turbidity measurements, which corresponds to time intervals where the filter is cleaned. In the time intervals between the spikes, drinking water is produced at required turbidity levels.

5.4.3 Exploratory Data Analysis. To visualize the heterogeneity in the two datasets, we split each dataset in chunks each of 2016 data points (two weeks of data), and create a box-and-whisker plot of the target variable for each chunk. See the two upper plots in Figure 4.

For the wastewater treatment dataset (Veas) we see a large heterogeneity between chunks in terms of the variation in location (median) and spread (interquartile range). This is not the case for the drinking water treatment data (Bergen) with quite stable location and spread in the data across the chunks except for chunks 19-21 including the measurement errors. We also show how the relationship between the input features and the target variable varies between chunks in terms of the Pearson correlation coefficient (PCC). See the two lower plots in Figure 4. In particular, for the Veas dataset, we see that the PCC varies from chunk to chunk and even switches signs. The picture is different for the Bergen data where the PCC is stable across the chunks for each input feature, again with the exception of chunks 19-21 where there are measurement errors in the turbidity.

Altogether the plots in Figure 4 indicate that there is a varying data-generating process for the wastewater treatment data, while for the drinking water treatment data the data-generating process is considerably more stable across the dataset.

5.4.4 Online Prediction Framework and Initial Hyperparameter Selection. Unlike in the simulation data, we typically do not know precisely the time spans of the data-generating processes in real process industry data. Hence, in addition to model-specific hyperparameters, chunk length and look-back length need to be determined before the deployment of an automatic forecasting process. We evaluate these hyperparameters by exploring the effect on forecast performance for different values of one parameter while holding the other fixed and vice versa (see Figures 10,11). To decide on model-specific hyperparameters we reserve the first third (33 %) of the data (around half a year) for HPO, using the first 90% to train models and the last 10% to evaluate the hyperparameter choices. During HPO we train and evaluate hyperparameters in the global model fashion such that the baseline model becomes as optimal as possible. We then apply the hyperparameters from HPO of the global model for each chunk model in the ensemble, except hyperparameters deciding the number of tunable parameters (e.g. neurons per layer). In this case, we scale down the number of parameters per chunk model such that the corresponding ensemble has an equal number of tunable parameters to the global model, given the specified chunk length. More details regarding this are given in Section 5.4.5. The resulting hyperparameters are specified in the files configs/model/{veas,bergen}_{rnn,tcn,xgboost,elastic_net}.yaml in the project code-base available at https://github.com/SINTEF/rewts.

For the ReWTS ensemble model each chunk (2016 data points) is standardized separately. Each chunk is divided into a 75% training set and a 25% validation set, where the training set is standardized by the mean and standard deviation prior to training of the model. The global model data is scaled after the same principle, but using 90% of all data (i.e. all chunks concatenated) as training data and 10% as validation.

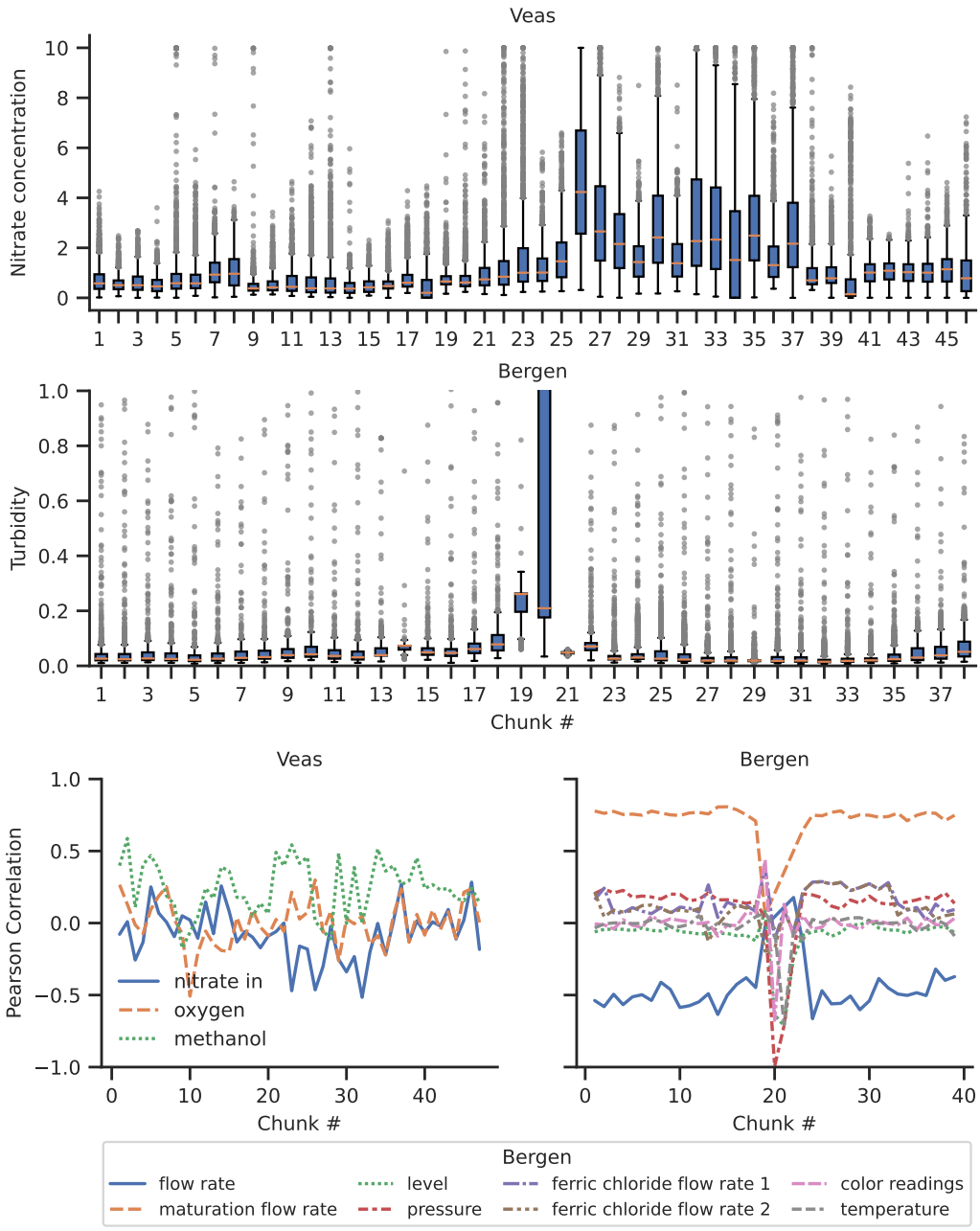


Fig. 4. Upper and middle plot: Boxplots demonstrating median, first and third quantiles, and outliers for each chunk in the Veas and Bergen datasets respectively. Lower plot: Pearson correlation coefficient between each input feature and the target feature separately for each chunk for Veas (left) and Bergen (right) respectively. Note how the Veas dataset exhibits significantly more heterogeneity across time than the Bergen dataset with the exception of the period with measurement errors intentionally included in the Bergen data.

We investigate four different model architectures: a linear model with elastic net regularization (Zou and Hastie 2005), the tree ensemble boosting model XGBoost (T. Chen and Guestrin 2016), a temporal convolutional network (TCN) (Bai et al. 2018) and a LSTM model (Staudemeyer and Morris 2019). All model architectures are directly available through the Darts Python package. These models have in common that the corresponding input data to the model includes both historical values of the target variable as well as other covariates that are thought to affect the target variable. The covariates used in the model are assumed known into the future when computing the forecasts. This is often the case in process industries where control signals can be fully determined, for example the future dosing of chemicals. All the chosen model architectures can be applied *autoregressively* to produce multi-step forecasts, however, note that the HPO procedure described above resulted in the linear, XGBoost, and LSTM model all outputting one-step predictions at a time, while the TCN model outputs several steps at once.

5.4.5 Forecasting on New Incoming Data. In this section, we outline the automatic forecasting procedure for the global and ReWTS ensemble models for a particular process industry dataset with measurements in the interval $[0, t_e]$. Given a model architecture, model-specific HPO of the global model is executed as explained in Section 5.4.4 using the data in the interval $[0, t_e/3]$. Thereafter, chunk length, l_c, and look-back length, l_b, for the ReWTS ensemble model is set, which gives $\lambda = \lfloor t_e/3l_c \rfloor$ chunk models within the interval $[0, t_e/3]$. If the model architecture is a neural network (TCN or LSTM), let p denote the fitted number of trainable parameters resulting from the chosen hyperparameters from HPO. To facilitate a fair comparison, we ensure that the global model at all times has at least as many trainable parameters as the sum of trainable parameters of the ReWTS ensemble model. This is achieved by downscaling the hyperparameters affecting the number of trainable parameters of each chunk model, such that the number of trainable parameters is approximately p/λ. For model architectures of type linear and XGBoost, the hyperparameter values of each chunk model will be equal to those hyperparameter values found during HPO of the global model. At this point, all hyperparameter values are set.

The automatic forecasting process for the ReWTS ensemble method is carried out as explained in Section 4.2 with $C = 2$. That is, we start the forecasting when we initially have trained two chunk models from the chunks $[0, t_{l_c}]$ and $[t_{l_c}, t_{2l_c}]$ respectively, with p/λ trainable parameters in each chunk.

The automatic forecasting process for the global method, with $C = 2$, is carried out as described in 5.2. Hence, the initial global model is trained on data from both chunks on the whole time interval $[0, t_{2l_c}]$. The global model is the same type of model as for each chunk model. For the linear model and XGBoost model architecture, the hyperparameter values are identical. For the LSTM and TCN model architectures, the number of trainable parameters of the global model is equal to $2p/\lambda$, such that it matches the total number of trainable parameters of the corresponding ReWTS ensemble. A global model is retrained from scratch for every new chunk of data of length l_c using all historical data. When retraining the TCN and LSTM model, the number of trainable parameters is adaptively upscaled from $(C-1)p/\lambda$ to Cp/λ when finishing chunk number C.

For comparison, we will also compare the ReWTS ensemble with static global models without upscaling trainable weights. Specifically, the constructed global model after HPO with p trainable parameters is directly applied in Algorithm 2 with $C = 2$. Hence, the retraining includes the data from the most recent chunk, but the number of trainable parameters, p, is kept constant.

We compare the performance of the automatic forecasting process for the global models and the ReWTS ensemble models respectively for each chunk using the loss function given in Equation (2). The total performance over the whole dataset is measured as the mean performance over all chunks. Further, we perform a one-sided Wilcoxon signed-rank test to infer the null hypothesis whether the difference in MSE between the ReWTS method and the global method follows a probability distribution function (PDF) which is symmetric about zero indicating no difference in performance between the methods. We reject the null hypothesis at a significance value of 0.05. The rejection of this null hypothesis indicates that the ReWTS method consistently outperforms the global

method in terms of the MSE by comparing their performances in each chunk separately. Finally, to see the effect of the choice of chunk length and look-back length, we repeat the process above for different values of chunk length, and look-back length. In particular, we investigate the chunk lengths 5, 7, 14, 21 and 28 days while keeping the look-back length fixed at 3 days. The look-back length is investigated for 2, 3, 5, 7 and 10 days while keeping the chunk length at 14 days.

5.5 Application on Publicly Available Time Series Datasets

We explore further the capability of the ReWTS ensemble model beyond noisy and complex process industry data by looking at a collection of four publicly available datasets of different sizes and from different domains.

The first dataset is the Air Quality (AQ) dataset (Vito 2008), which contains sensor measurements collected from March 2004 to February 2005 in a polluted area of an Italian city which gives 9357 data points in total. It includes hourly averaged ground-truth concentrations of carbon monoxide, non-methanic hydrocarbons, benzene, total nitrogen oxides, and nitrogen dioxide. We use the benzene concentration as the target variable and forecast its values 12 hours ahead.

The second dataset is the Beijing Multi-Site Air Quality (BMAQ) dataset (S. Chen 2017), which contains hourly recordings of air pollutants, particulate matter concentrations (PM2.5 and PM10), and meteorological variables from 12 air-quality monitoring sites in Beijing between March 2013 and February 2017. For simplicity, we analyze the recordings from a single site, the Changping district which gives 35 064 data points in total. The forecasting task is to predict PM2.5 concentrations 12 hours ahead.

The third dataset investigated is the ELEC2 dataset, which consists of electricity prices collected from the Australian New South Wales Electricity Market between May 1996 and December 1998 (Harries 1999). Each data point of the dataset refers to a period of 30 minutes which gives 45 312 data points in total. We use the New South Wales electricity price (nswprice) as the target variable, and forecast these prices 12 hours into the future.

The fourth dataset is the Individual Household Electric Power Consumption (IHEPC) dataset, which includes measurements of electric power consumption in one household located in Sceaux, France between December 2006 and November 2010. Specifically, minute-averaged global active power, global reactive power, voltage, global intensity, and energy consumed from three submeters are recorded for every minute. The one-minute time resolution is resampled to a frequency of 10 minutes, and we restrict to data collected until December 2008 which gives 100 800 data points in total. We use the global active power as target, and forecast the next 3 hours.

We follow the procedure as described in Section 5.4.5, however with some simplifications and extra explorations. 1) For simplicity, we do not explore different chunk lengths, look-back data lengths and static global models; instead, we keep the chunk length fixed corresponding to two weeks of data and the look-back length fixed corresponding to two days of data. We upscale the global model for the LSTM and TCN model. 2) For the TCN model we explore the situation, across all datasets, where future covariates are not assumed known, and where forecasts are produced in a direct multi-step manner instead of autoregressively. 3) For the BMAQ and IHEPC datasets, we will do a modification to reduce the inference time of the ReWTS ensemble model by setting a stride, $s = 6$, during the weight fitting procedure given in (4). For the IHEPC dataset we will moreover recalculate the ReWTS weights only after every 16 hours (96 data points) to reduce the computational complexity even further.

6 Results

We present all results following the methodology described in Section 5.

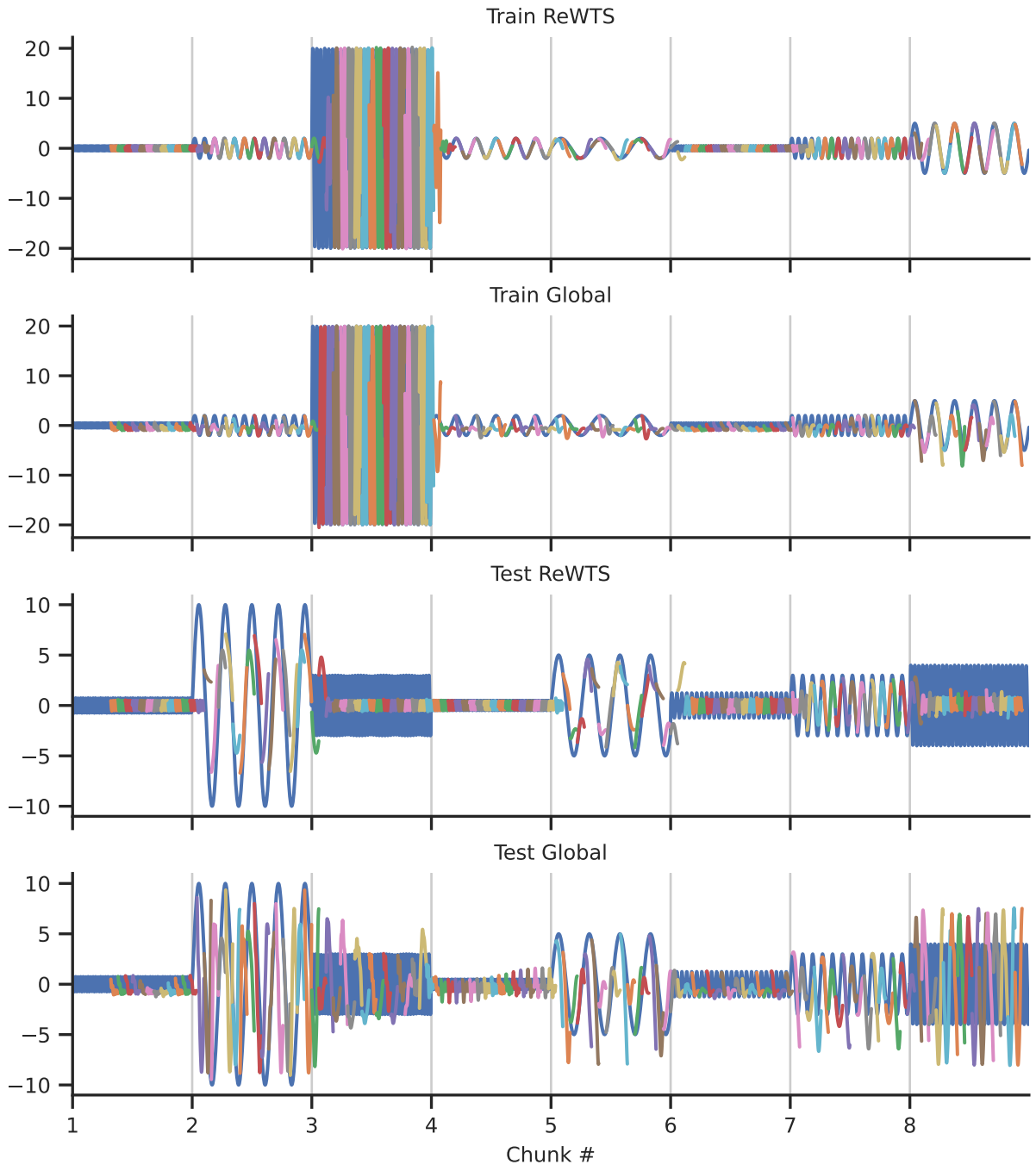


Fig. 5. Data (blue) and model predictions (colours) for all chunks for the training set and the test data from the sine experiment. Note how the global model has prioritized learning the dynamics of the dominant amplitude in chunk #3 of the training set.

6.1 Simulation Data

We evaluate our proposed ReWTS ensemble method by using the simulated data as described in Section 5.3. A summary of the results is given in Table 2 and Figure 5 showing the performance of both the ReWTS ensemble model and the global model according to the loss function given in 2 for each chunk individually as well for all chunks as a whole. The ReWTS ensemble model generally outperforms the global model on the training dataset as well as on the unseen test dataset.

Table 2. Relative (normalized by the maximum amplitude value of the chunk) mean squared error from applying the ReWTS and global methods on the simulation data.

Dataset	Model	Chunk								
		#1	#2	#3	#4	#5	#6	#7	#8	Mean
Train	ReWTS	**1.97E-05**	**6.04E-04**	4.86E-03	**9.41E-03**	**2.00E-02**	**4.15E-05**	**3.95E-04**	**2.37E-02**	**7.38E-03**
	Global	7.49E-01	6.13E-01	**1.47E-03**	1.18E+00	1.38E+00	1.33E+00	6.90E-01	1.20E+00	8.94E-01
Test	ReWTS	**5.90E-02**	**1.74E+00**	**1.20E+00**	**2.46E-01**	**7.57E-01**	**4.87E-01**	**2.02E-01**	**8.36E-01**	**6.90E-01**
	Global	1.08E+00	6.15E+00	4.55E+00	1.11E+00	2.05E+00	9.47E-01	1.87E+00	7.28E+00	3.13E+00

This holds true when evaluating over all concatenated chunks as one time series (Figure 5), and when assessing each chunk individually, except for Chunk #3 in the training dataset where the global model attains a smaller loss. See Appendix B to get a closer look at the predicted forecasts from the ReWTS ensemble model and the global model for each of the 16 chunks. This behaviour follows our hypothesis, where the global model does exceptionally well on the chunk with the highest amplitude that dominates the learning objective, at the cost of performance on the other data chunks. Furthermore, the ReWTS ensemble model demonstrates superior extrapolation abilities compared to the global model with respect to the results on the test data.

Note that when evaluating the ReWTS ensemble model over concatenated chunks, its adjustment to a new data-generating process is more gradual compared to the global model. This behaviour is due to the weight assignment procedure based on the look-back data preceding the current prediction point. As a result, the look-back data will include information from the previous chunk, slowing the ReWTS ensemble's adjustment, until we are further than the length of the look-back data into the new chunk of the data-generating process. These edge effects are highlighted in Figure 6, where the global model can be seen to be better at adapting to a sharp change in the dynamics.

The evaluation on the simulation data has been based on an autoregressive LSTM model. As explained in Section 4, the weights are calculated based on the performance of the h-step forecasts for each model on the look-back data. This is intuitive to do since the final task is to predict h-step forecasts at the present time point, and hence the evaluation of the chunk models in the look-back data should reflect what is the final task. An alternative procedure for autoregressive models is to find the weights that minimize the *one-step* ahead forecast in the look-back data, equivalent to setting $h = 1$ in the QP optimization problem (4). At prediction time, the h-step forecasts ($h > 1$) can in this case be computed by reapplying the computed weights h times recursively. See Algorithm 3 for more details. More generally speaking, the forecast horizon applied in Algorithm 1 does not need to be the same as the desired forecasting horizon, h, at prediction time. In fact, the forecast horizon used in Algorithm 1 can be smaller than h, and the corresponding weights can be reapplied recursively the number of times necessary to provide h-step forecasts.

At prediction time, we want the ReWTS ensemble model to recognize previously seen patterns that correspond to the dynamics in the present chunk. In the algorithm, this means that a particular chunk model should be assigned a weight equal to one at prediction time if the exact same dynamics as in the previously observed

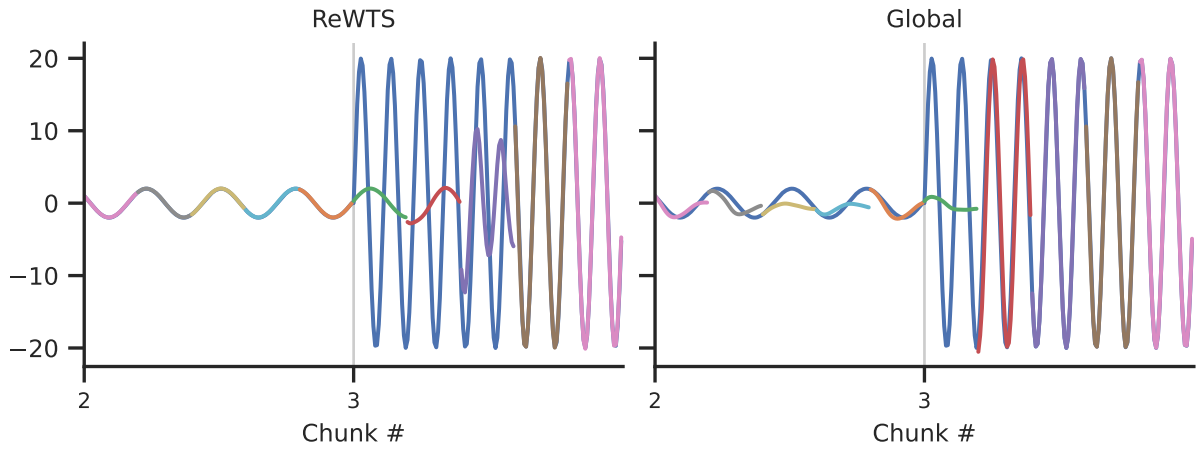
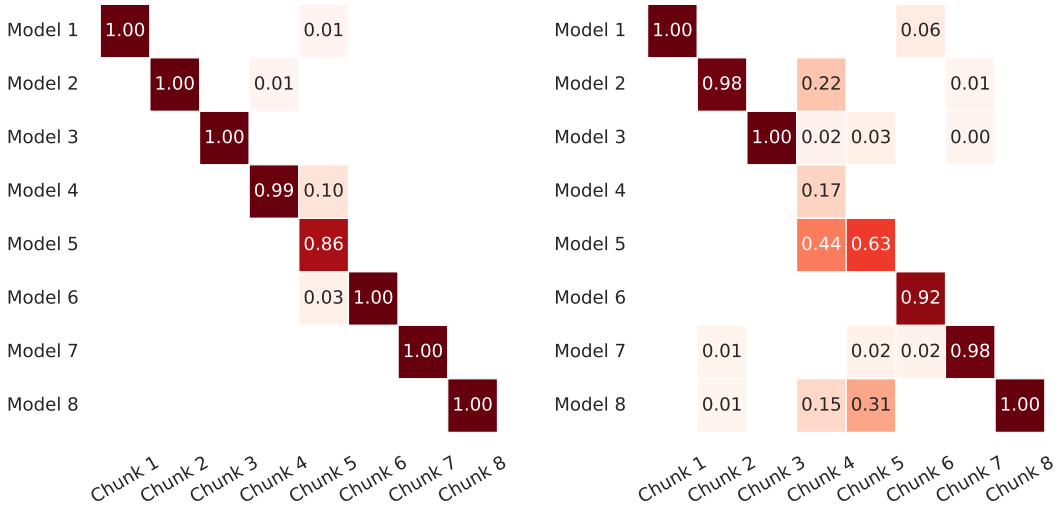


Fig. 6. With rapid change in the dynamics, the ReWTS ensemble model is slow to adapt (the true dynamics given by the blue line).



(a) Weights $h = 30$

(b) Weights $h = 1$

Fig. 7. Increasing the forecast horizon when fitting ensemble weights increases the accuracy with which it identifies the most suitable model.

chunk reappear in the future. In Figure 7, we investigate how different weight assignment strategies affect the distribution of the weights for each model for each chunk. The first weight assignment is performed by setting the forecast horizon in Algorithm 1 equal to the forecast horizon at prediction time, $h = 30$. The second weight assignment is that of the alternative Algorithm 3, with a forecast horizon equal to one. In this approach, we search for the weights that minimize the average MSE of the one-step-ahead predictions in the look-back dataset.

Algorithm 3 The one-step weight fitting with forecasting horizon h - ReWTS ensemble model

▷ Given forecast horizon, h, look-back length , l_b, chunk length, l_c, stride s and present time point t_n with n/l_c an integer.
▷ Assign $C = n/l_c$, the number of available chunks
▷ Assign $v = n$
▷ Given all previously trained models, $\mathcal{M}_v = \{M_1, ..., M_C\}$ from the C previous chunks
repeat
  while $v \leq (C+1) \cdot l_c$ **do**
    ▷ Collect look-back data $(X_{(v-l_b):v}, \mathbf{y}_{(v-l_b):v})$
    ▷ Calculate the weights $\hat{\mathbf{w}}(t_v)$ according to optimization problem (4) with $h = 1$ given chunk models in $\mathcal{M}_v$
    for $j = 1 : h$ **do**
      if $j = 1$ **then**
        ▷ Compute one-step ahead forecast at time t_{v+1} for each chunk model, and save in $1 \times |\mathcal{M}_v|$ matrix $M_1(X_{:v}, \mathbf{y}_{:v})$
        ▷ Compute final forecast at time point t_{v+1} by ReWTS ensemble given by $\hat{y}_{v+1} = M_1(X_{:v}, \mathbf{y}_{:v})\hat{\mathbf{w}}(t_v)$
      else if $j > 1$ **then**
        ▷ Compute one-step ahead forecast at time t_{v+j} for each chunk model by using the previous predictions $\hat{\mathbf{y}}_{v+k}$ for $k = 1, \ldots, j-1$, and save in $1 \times |\mathcal{M}_v|$ matrix $M_1(X_{:(v+j-1)}, \hat{\mathbf{y}}_{:(v+j-1)})$, with $\hat{\mathbf{y}}_{:(v+j-1)}$ including all known previous values of the target as well as the most recent predictions of the target
        ▷ Compute final forecast by ReWTS ensemble at time t_{v+j} given by $\hat{y}_{v+j} = M_1(X_{:(v+j-1)}, \hat{\mathbf{y}}_{:(v+j-1)})\hat{\mathbf{w}}(t_v)$
      end if
    end for
    ▷ $v = v + s$
  end while
  ▷ Collect all data from new chunk number $C+1$ between time points $t_{C \cdot l_c}$ and $t_{(C+1) \cdot l_c}$
  ▷ Train new chunk model M_{C+1} from chunk $C+1$ and update $\mathcal{M}_v$
  ▷ $C = C + 1$
until Collection of new data stops

The fitted weights can afterwards be used recursively to provide the desired forecast horizon at the prediction time. We see that the first weight assignment ($h = 30$, Figure 7a), outperforms the second weight assignment ($h = 1$, Figure 7b), in terms of allocating the correct chunk model with the largest weight.

By the construction of the ReWTS ensemble model, the purpose of the weight assignments of the chunk models goes beyond the goal of achieving better forecasts than a corresponding global model. In fact, the weights can be used for interpretation, such as identifying previous intervals that are considered relevant at prediction time. Figure 8 shows how the weights are distributed at different prediction times for the training set, when applying Algorithm 1 with $h = 30$. Inside the chunks, the correct chunk model is assigned a large weight close to or equal to one. To evaluate the trustworthiness of a particular forecast, one way to proceed is to investigate the properties of the data distributions of the chunk datasets corresponding to the chunk models with the largest weights. This can be a valuable property in the context where the detection of recurring events is relevant.

In Figure 8 we also observe the previously discussed edge effects between two neighbouring chunks. Another interesting observation is at chunk 5 where the ReWTS ensemble model is struggling the most to have a weight

distribution concentrated around the correct chunk model. By closer inspection, one can see that the chunk model for chunk 4 is assigned a weight of 0.1. This is reasonable given the close resemblance between the dynamics in chunks 4 and 5.

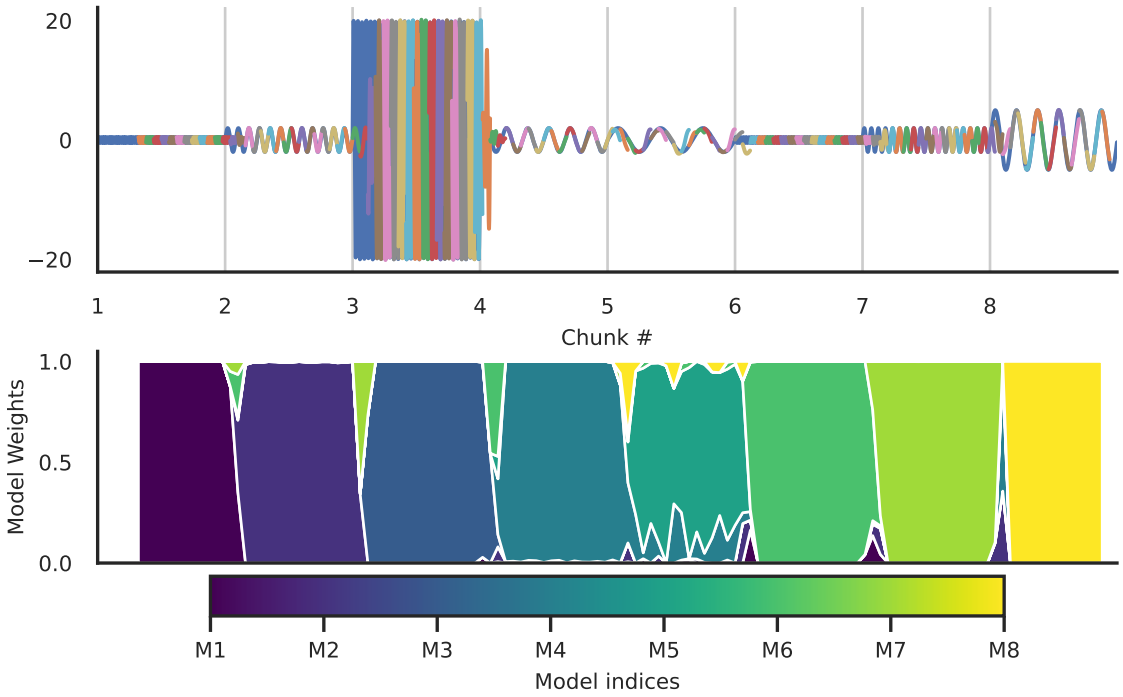


Fig. 8. Ensemble model predictions on the train set along with the weights assigned to the models.

6.2 Process Industry Data

We present the evaluation of the ReWTS ensemble method based on the real process industry data from both the Veas wastewater treatment plant and the drinking water treatment plant in Bergen. Based on the results from the simulation data, the ReTWS ensemble models are trained using Algorithm 1 with the parameter h equal to the desired forecasting horizon at prediction time.

6.2.1 Results on Wastewater Treatment Plant. Figure 9 shows the result when comparing the ReWTS ensemble method with the global method on the data from the wastewater treatment plant with a chunk length of 2016 time points (corresponding to two weeks of data), and a look-back data length of 300 time points (corresponding to 50 hours of data).

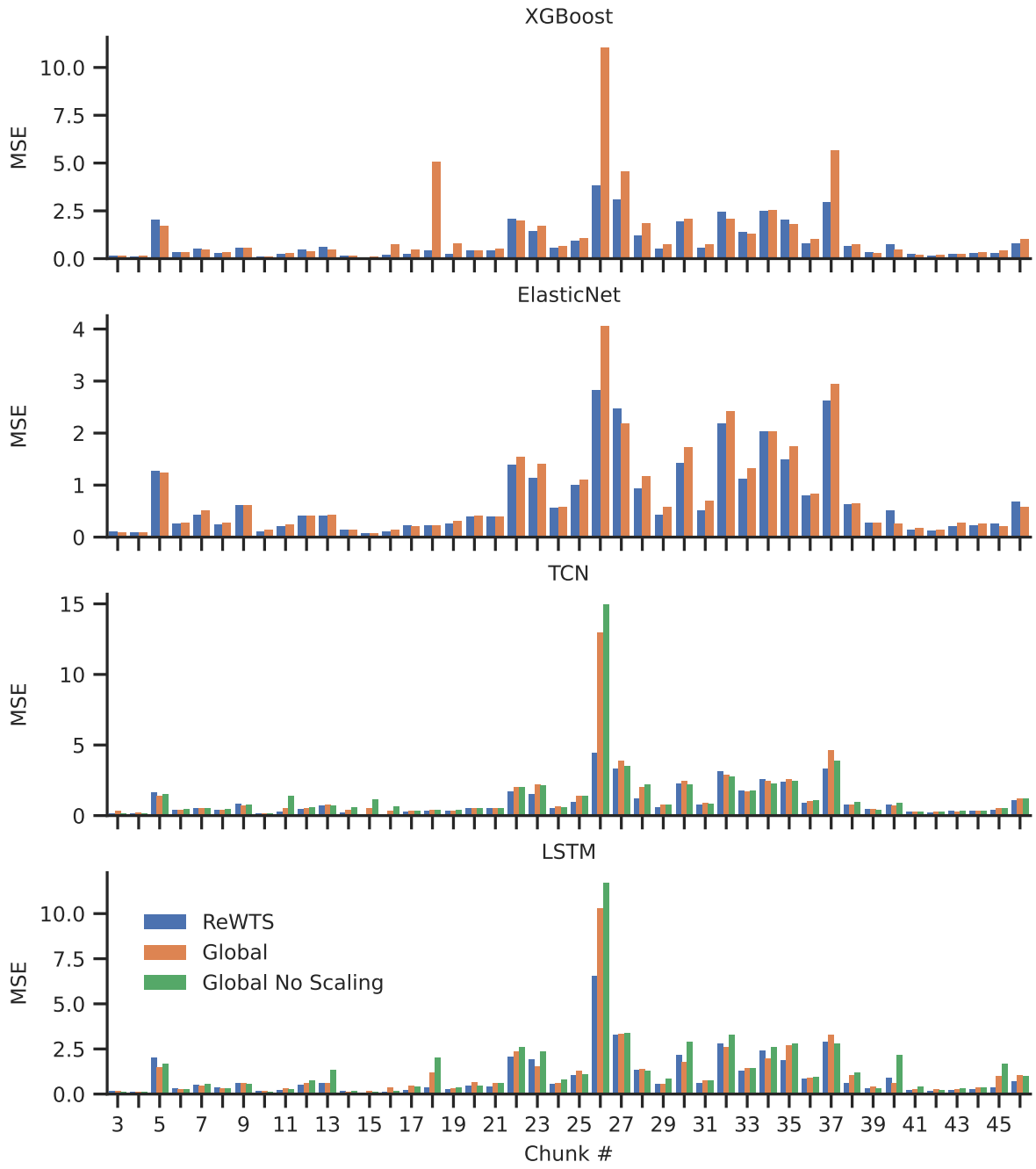


Fig. 9. Comparing the ReWTS ensemble with the corresponding global models for the wastewater treatment plant data. The TCN and LSTM models were upscaled for each new chunk to match the total number of trainable parameters in the ReWTS ensemble. For comparison, a global model without upscaling (Global No Scaling) is also presented.

Table 3. The average MSE over all chunks of the data from the wastewater and drinking water treatment plants for the global method and ReWTS ensemble method, along with the percentage difference, for different model architectures. Included is also the one-sided Wilcoxon signed-rank test which infers whether the ReWTS method outperforms the global method in terms of the MSE grouped by chunks.

		Architecture			
Dataset		Elastic net	XGBoost	TCN	LSTM
Veas (94 988 data points, 46 chunks)	ReWTS MSE	**0.73**	**0.92**	**1.01**	**0.99**
	Global MSE	0.81	1.15	1.33	1.16
	% Difference	10.1	22.5	28.0	15.9
	Wilcoxon p-value	0.00037	0.0044	0.000051	0.0023
Bergen (74 768 data points, 38 chunks)	ReWTS MSE	**0.04**	**0.05**	**0.12**	**0.09**
	Global MSE	0.09	0.11	0.19	0.12
	% Difference	74.6	65.9	48.4	33.9
	Wilcoxon p-value	0.70	0.065	0.55	0.68

In order to assess the performance of each of the two procedures, we compute the loss function according to (2) for *each chunk*, a total of 44 chunks (the first two chunks are omitted, as the ReWTS ensemble must necessarily have at least two models). We then compute the average loss, the average MSE, across all chunks for the global and the ReWTS ensemble methods, separately for each model architecture. In addition to assessing the mean performance over chunks, we test if one method is consistently outperforming the other. To this end, we rank the methods on each chunk based on the MSE and perform a one-sided Wilcoxon signed-rank test using each chunk as a sample to test if ReWTS consistently outperforms the global method. The numbers are given in Table 3.

Consistently, for all model architectures, the global model is trailing the ReWTS ensemble model in terms of average MSE over the chunks. The reduction in average MSE is significant, and in the range from 10 % to 28 %, with the smallest reduction for the elastic net model, and the largest reduction for the TCN model. Interestingly, the linear model with regularization, elastic net, yields the best average MSE, both for the global model and the ReWTS ensemble model. This may be explained by the noisy measurements of the nitrate concentration (see Section 5.4.1). For complex non-linear models, this noise combined with the high heterogeneity of this dataset (Section 5.4.3) may lead to overfitting and poorer generalizability between heterogeneous chunks. The superior performance of ReWTs over the global method indicates that increased model complexity is less useful than the ability to adapt between chunks, when there is unpredictable variation in time.

To further investigate the effect of the chunk length, we repeat the process above by keeping the look-back length constant at 300 data points, but varying the chunk length over 5, 7, 14, 21 and 28 days. The results are given in Figure 10 with chunk length on the x-axis and the average MSE over the chunks on the y-axis for the four model architectures.

Within this range of chunk lengths, the ReWTS ensemble model consistently outperforms the corresponding global model. The trend, while not consistent across all model architecture, is that the global model improves the more often it is updated (i.e. with smaller chunk lengths). There is also a trend that the ReWTS ensemble model improves the smaller the chunk length is.

We do the same exercise to explore the sensitivity of the look-back length for the ReWTS ensemble model. We keep the chunk length constant to two weeks of data, and vary the look-back length over 2, 3, 5, 7 and 10 days of data. The results are given in Figure 11. The trend is clear, and consistent across all model architectures, that the

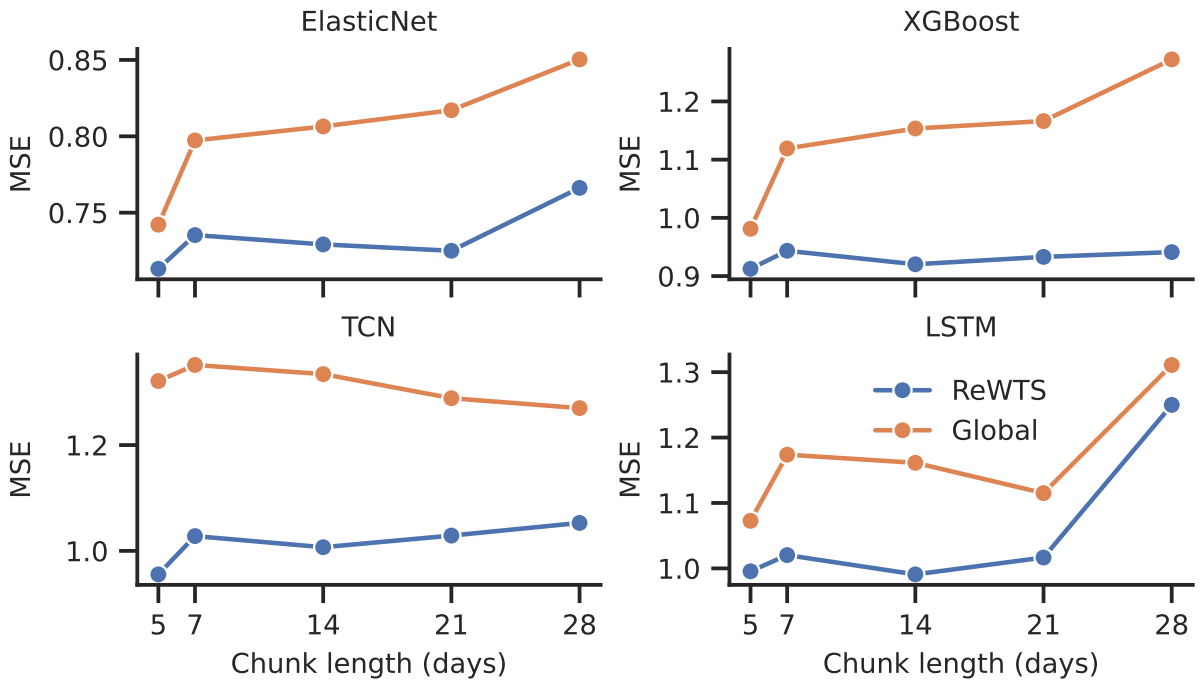


Fig. 10. The average MSE, for different model architectures, of the ReWTS ensemble model for varying chunk length (with a constant look-back length of 300 data points). Results when applied on the wastewater treatment data. The results are compared with the corresponding global model.

smaller the look-back length, the better performance with respect to the average MSE across the chunks. In total, it is apparent that it is beneficial to keep both the chunk length, and the look-back length sufficiently small.

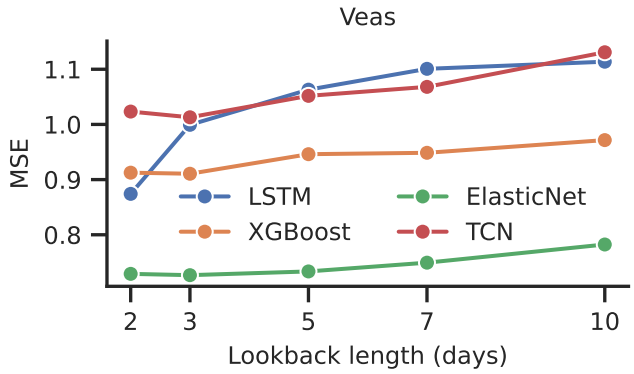


Fig. 11. The average MSE, for different model architectures, of the ReWTS ensemble model for varying look-back length (with constant chunk length equal to two weeks of data). Results are given for the wastewater treatment data.

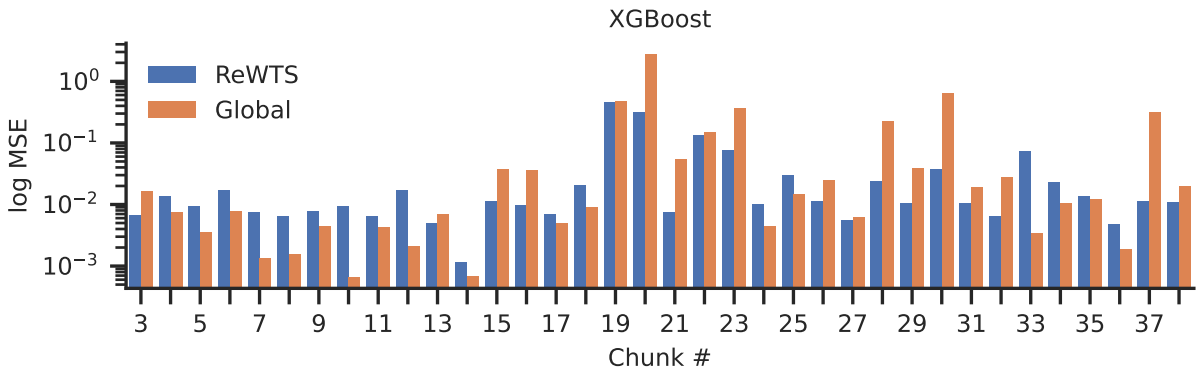


Fig. 12. Comparing ReWTS ensemble models with the corresponding global models for the drinking water treatment plant data using the XGBoost model architecture. Note that the y-axis is in log scale.

6.2.2 Results on Drinking Water Treatment Plant. Figure 12 shows the result when comparing the ReWTS ensemble with the global method using the XGBoost architecture on the data from the water treatment plant with chunk length of 2016 time points (corresponding to two weeks of data), and a look-back data length of 300 time points (corresponding to two days of data). The corresponding plots for elastic net, TCN and LSTM can be seen in Appendix C.

Table 3 shows the average MSE for the different model architectures. The reduction in average MSE is significant, and in the range from 34 to 75 %, with the smallest reduction for the LSTM model, and the largest reduction for the linear model. Most of this percentage difference is due to the measurement errors in the time interval between chunks 19 and 21, as can be seen in Figure 12. However, from the Wilcoxon signed-rank test in Table 3 we see, across all model architectures investigated, that we cannot conclude that the ReWTS method consistently outperforms the global method on this dataset.

The erroneous measurements described in Section 5.4.2 can be found between chunks 19 and 22. That is the reason why we see a sudden jump in MSE as a result of unseen, anomalous data. What is of particular interest, however, is what happens with the behaviour of the ReWTS and global models afterwards, and to what extent the two methods are affected by the measurement errors with respect to future forecasts. From Figure 12, we see that ReWTS maintains a small MSE in future chunks, while the global method can have sudden spikes in the MSE (such as for chunks 30 and 37), which indicates that measurement errors, if not treated correctly or not removed from the training data, may be more damaging to the future performance of the global model than the ReWTS model. Nonetheless, we cannot conclude based on the Wilcoxon signed-rank test that the ReWTS method consistently outperforms the global method for this specific dataset. When comparing the results with the wastewater treatment plant use case, we see that the MSE is in general smaller across all chunks. This is reasonable as the drinking water plant data is less heterogenous, and has less noise in its measurements. As for the wastewater treatment data, the average MSE of the ReWTS ensemble model is computed for varying chunk lengths and look-back lengths. The trends are mostly the same as for the wastewater treatment data, and for completeness, the corresponding plots are given in Appendix C.

6.2.3 Results on Publicly Available Data. We do the same exercise of comparing ReWTS with the corresponding global model for the publicly available datasets AQ, BMAQ, ELEC2 and IHEPC with some modifications: 1) A fixed chunk length (two weeks) and look-back data length (two days) for all datasets, 2) The TCN model produces the forecast in one go (without access to future covariates), unlike the autoregressive setup used for the other

Table 4. The average MSE over all chunks of the publicly available datasets, along with the percentage difference, for different model architectures. Included is also the one-sided Wilcoxon signed-rank test which infers whether the ReWTS method outperforms the global method in terms of the MSE grouped by chunks.

		Architecture			
Dataset		Elastic net	XGBoost	TCN	LSTM
Air Quality (9357 data points, 27 chunks)	ReWTS MSE	1.92E+00	6.90E-01	4.61E+01	4.09E+00
	Global MSE	**1.76E+00**	**3.03E-01**	**4.19E+01**	**1.23E-01**
	% Difference	-9.1	-78.1	-9.5	-188.3
	Wilcoxon p-value	5.94E-01	1.00E+00	0.99E+00	1.00E+00
Beijing Multi-site Air Quality (35 064 data points, 104 chunks)	ReWTS MSE	6.62E+02	8.18E+02	4.49E+03	9.86E+02
	Global MSE	**5.84E+02**	**5.74E+02**	**3.46E+03**	**4.90E+02**
	% Difference	-12.5	-35.1	-25.9	-67.3
	Wilcoxon p-value	0.99E+00	1.00E+00	1.00E+00	1.00E+00
ELEC2 (45 312 data points, 67 chunks)	ReWTS MSE	1.50E-03	**8.48E-04**	**1.98E-03**	1.39E-03
	Global MSE	**1.44E-03**	1.05E-03	2.02E-03	**1.00E-03**
	% Difference	-4.2	20.8	2.2	-32.3
	Wilcoxon p-value	4.10E-01	3.59E-05	4.30E-02	8.63E-01
House Electric Power (100 800 data points, 50 chunks)	ReWTS MSE	**7.08E-04**	3.50E-03	**9.30E-01**	2.17E-03
	Global MSE	7.15E-04	**1.61E-03**	9.34E-01	**3.42E-04**
	% Difference	1.0	-74.0	0.4	-145.7
	Wilcoxon p-value	3.20E-02	9.95E-01	4.10E-01	1.00E+00

models, and 3) For the BMAQ and IHEPC, the weight fitting in ReWTS is carried out with a stride larger than one ($s = 6$) across the look-back data, and additionally for the IHEPC dataset the weights are refitted only after every 16 hours (96 data points). See the results in Table 4 which show that in most cases the global model is outperforming the ReWTS ensemble model. This is especially clear for the AQ and BMAQ datasets both in terms of the percentage difference and the result from the Wilcoxon signed-rank test. The main exception is the ELEC2 dataset with XGBoost, where the ReWTS ensemble is significantly outperforming the global model, and it is also the model giving the lowest average MSE across all model architectures. For the IHEPC dataset, the ReWTS ensemble is competitive with the global model for the Elastic net and TCN model architectures. As expected, the TCN model has the highest average MSE overall, reflecting the difficulty of producing the full forecast in one step without future information of the covariates. Nonetheless, the general pattern remains: global models dominate on AQ and BMAQ, while ReWTS performs more competitively on ELEC2 and IHEPC.

6.2.4 Computational Complexity of the ReWTS Ensemble Model. We analyse the computational complexity of the ReWTS ensemble model and the global model. The computations can be separated into two separate tasks. The first task is the training of each model in its respective chunk. The second task is with respect to the forecasting. Note that for the ReWTS ensemble model, according to the procedures in 1, forecasts need to be evaluated on the look-back data to compute the weights, before providing forecasts of the future. Conversely, the global model can at once provide future forecasts.

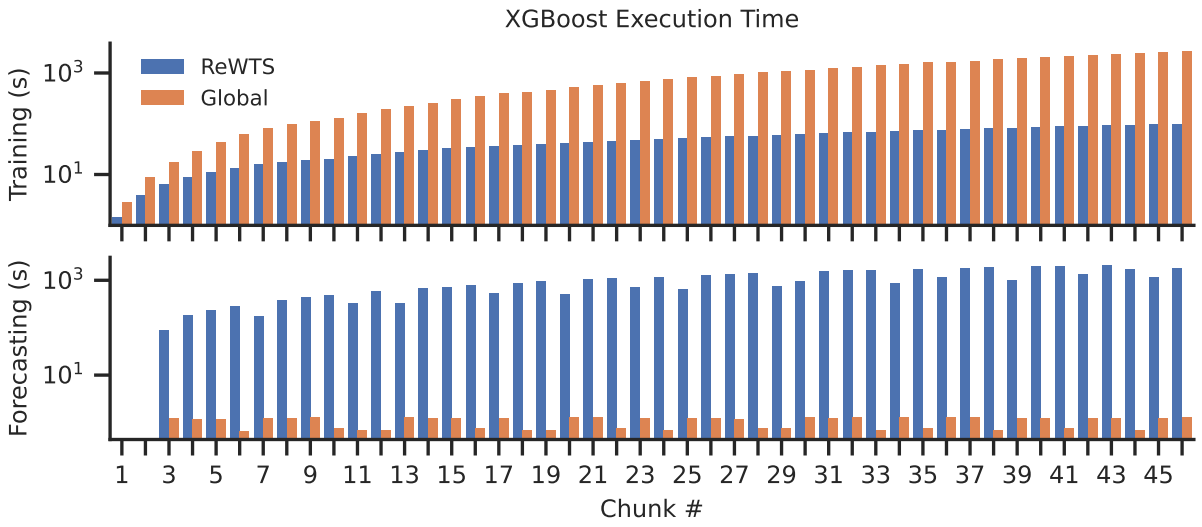


Fig. 13. Comparison of training and forecasting execution times for the global and ReWTS ensemble methods on wastewater treatment plant data using the XGBoost model (logarithmic scale).

Figure 13 illustrates the computational time required for both the training and forecasting procedures when employing the ReWTS ensemble model and the global model on wastewater treatment data, using the XGBoost architecture. The upper plot of Figure 13 depicts the total training time after completing each chunk, comparing the ReWTS ensemble model and the global model with a logarithmic scale on the y-axis. The ReWTS ensemble model updates by training a new chunk model with a fixed number of data points, defined by the chunk length. In contrast, the global model updates by retraining on an increased amount of data points equal to the chunk length. Consequently, the ReWTS ensemble model requires less computational time in this regard, and the difference in training time between the two methods amplifies over time. On the other hand, for the forecasting procedure, the computational time is significantly larger for the ReWTS method than the global method, and the difference increases as more chunk models are added to the ReWTS ensemble. This is because the number of forecasts produced on the look-back data, as part of the weight fitting procedure, increases linearly in the number of chunk models. The global model, on the other hand, does not rely on any look-back data, and can provide the forecasts immediately at prediction time.

7 Discussion

Figure 14 depicts the variation in the performance of the chunk models through a boxplot of the MSEs they produce when forecasting individually, compared to the ReWTS ensemble. We observe that the ensemble has a much clearer advantage on the drinking water dataset. This is due to the lower heterogeneity of this dataset, leading to a more optimal past-conditioned weight-selection scheme. Conversely, with high heterogeneity, the look-back data is often not representative of the future. Therefore, with the wastewater treatment data some individual models outperform the ensemble at different times (note that it is not the same models consistently performing well on all chunks), although the ensemble is always among the top percentiles of MSE. Yet, it is with high heterogeneity data that we see the ReWTS method excel. This suggests that there might be other weight selection schemes that synergize better than using the recent past, an interesting direction for future research.

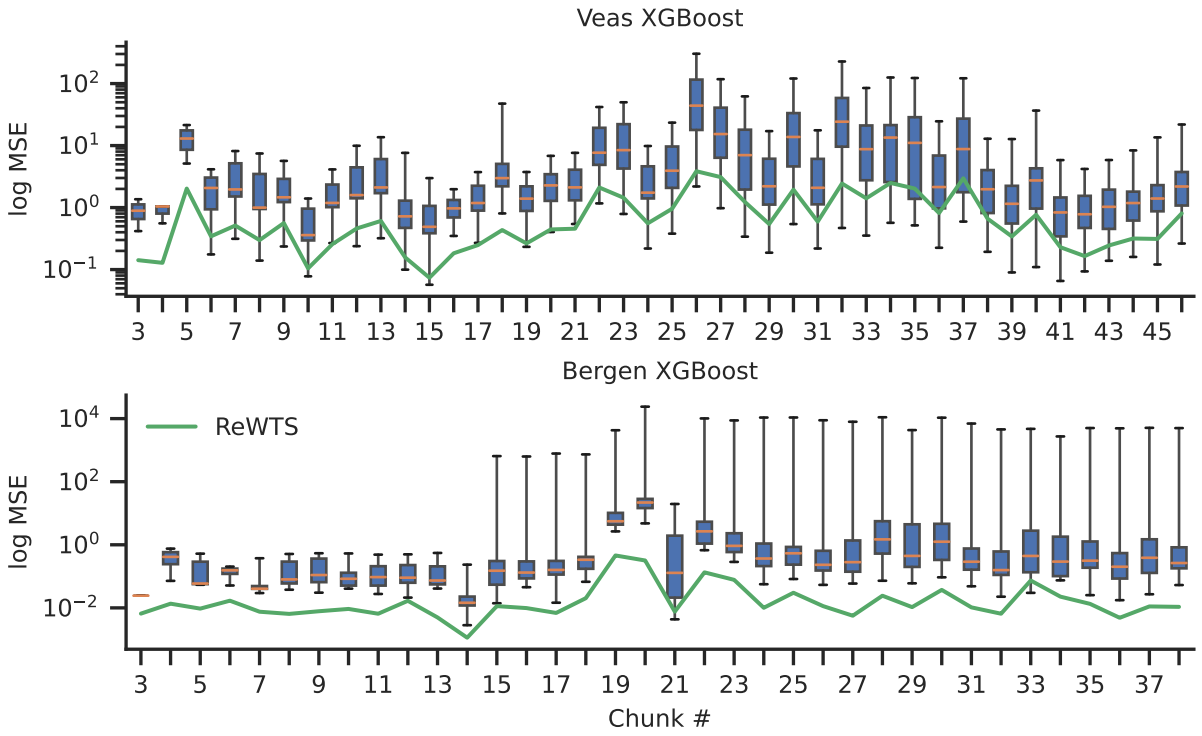


Fig. 14. Boxplot over the forecasts produced individually by each model in the ensemble, demonstrating their heterogeneity. The whiskers illustrate the highest and lowest MSE of any model, while the line depicts the MSE obtained by the ensemble as a whole. Note that sometimes individual models outperform the ensemble. This is to be expected as the ensemble selects models based on the past, while the depicted MSE measures forecasts for future data.

For both process industry datasets, each data chunk does not represent a static data-generating process. However, a chunk may include characteristics that are unique, or present only in a few other chunks. Hence, each chunk model can specialize within a distinct time interval with restricted heterogeneity in the dynamics. We deem this the most likely reason for the ReWTS method outperforming the global method in this case. When applied to the publicly available datasets (Table 4), we observe multiple cases where the global model outperforms the ReWTS ensemble when the chunk length and look-back length are fixed and not optimized. The performance of the ReWTS ensemble compared to the global model will depend on the choice of chunk length and look-back length. In Appendix D, we show for the ELEC2 dataset how ReWTS outperforms the global model for all model architectures in terms of average MSE when the chunk length is changed from two weeks to four weeks. Nonetheless, ReWTS is not a one-size-fits-all approach, and there are cases where the global model is a more performant approach. Taken together, these results suggest that the ReWTS ensemble may outperform a corresponding global model, depending on chunk length and look-back data length, in non-stationary datasets of sufficient size that include high heterogeneity and concept drift.

Another factor that might play a role is when important variables in the system are unobserved, as is often the case in complex process industries; in such cases, each chunk model in the ReWTS ensemble can capture distinct

dynamics induced by the varying influence of these hidden factors. Future research could more systematically characterize the conditions under which the ReWTS ensemble outperforms global models.

By varying the chunk length and look-back length for the process industry datasets Veas and Bergen, we conclude that both should be kept sufficiently small. Regarding the chunk length, this is intuitive in the case where a smaller chunk length provides more distinct, and less diverse characteristics in each chunk. The look-back length determines how much data in the past will be used for the weighting procedure. The shorter the length, the more recent data will be used. This can be an advantage in cases where the dynamics change rapidly. Domain knowledge about characteristic timescales in the system should guide these choices. As the ReWTS method is built for a continuous flow of data, HPO of the chunk length and look-back lengths cannot be performed at an early stage, and given the importance of these hyperparameters, the ReWTS ensemble is most relevant once there is sufficient amount of data available. To find the specific optimal values of these hyperparameters, one can run several ReWTS models in parallel and inspect which values yield historically the best forecasts at any given time point. We leave the exploration of disjoint, overlapping or different-length chunks to future work.

A benefit of the ReWTS ensemble, as opposed to the global models, lies in its enhanced capacity for interpretability and reliability analyses, namely by examining the weights of the different chunk models. One can assess the credibility of the forecasts by analysing the dynamics within the chunks that correspond to the chunk models that were assigned the largest weights.

When comparing the ReWTS ensemble with a corresponding global model, the number of trainable parameters was upscaled for the neural network architectures when retraining the global model, while not for the linear and XGBoost models. A procedure for upscaling the global XGBoost model was implemented by making sure the maximum number of regression trees from the global model would match the maximum number of regression trees from the corresponding ReWTS ensemble model, with and without early stopping. This did not lead to any significant changes in the performance of the XGBoost model.

In this work, the comparison between the ReWTS ensemble and the global model is based on four diverse model architectures: Elastic net (linear model), XGBoost (tree-based gradient boosting model), TCN (convolutional neural network) and LSTM (recurrent neural network). The results have shown that the benefit of ReWTS depends on the choice of model architecture. Interesting future research will be to investigate other model architectures including transformer models (Lim et al. 2021).

As seen from Figure 13, while the ReWTS ensemble is less time-consuming for training compared to the global model, the ReWTS ensemble is significantly more time-consuming during forecasting, due to the large number of predictions needed across the look-back data. This may pose challenges when it is important to get forecasts quickly in real time. For most datasets, the predictions across the look-back data during the weight fitting procedure of the ReWTS ensemble have been achieved with stride $s = 1$, and the weights of the ReWTS ensemble have been recomputed for every new prediction point. For the BMAQ and IHEPC datasets, we demonstrate an easy way to reduce the computational complexity by increasing the stride $s > 1$ in the look-back data which effectively reduces the number of computations proportionally to $1/s$ yielding a time complexity of $O(l_b/s)$ with l_b the look-back length. Moreover, by recomputing the weights with a stride $\tilde{s}$, as was done with the IHEPC dataset, the computational cost is further reduced with complexity inversely proportional to $\tilde{s}$. Investigating the trade-off between computational efficiency and forecast accuracy when tuning these stride parameters is left for future work.

As discussed in Section 3, several methods to discard underperforming chunk models have been proposed. By the construction of the weight assignment in ReWTS, the weights assigned to each model can give an indication of how important it is. For instance, a model that is often assigned a small weight, and with small variation, can be considered irrelevant in most cases. In this case, one could consider discarding it in future weight updates, although conversely, it could be that this chunk represents a rare but recurring event and as such would be important to keep. We leave this as future research.

In this work, we have focused on the mean squared error both for assigning the weights in the ReWTS ensemble model, and to evaluate the performance of the forecasting models. In principle, any other similarity measure can be applied to decide the weights in the ReWTS ensemble model. Note that the metric can be different between the model training and the weight assignment. This allows for the reuse of existing models of particular phenomena in the system to be combined with purely data-driven models.

An interesting modification to the ReWTS ensemble method would be to group sequences of data into chunks based on their similarities, as opposed to creating chunks by splitting the dataset in time. With respect to the wastewater treatment data, this means for instance to have one model trained to forecast future nitrate concentrations based on input sequences of small and stable (low variance) nitrate concentrations. Yet another model can be trained with input sequences where the nitrate concentration is rapidly increasing. Grouping the input sequences into different classes or clusters will require some assumptions or decisions. With this alternative grouping of data, at prediction time, the models are weighted based on how much a given input sequence resembles their respective group of data, therefore eliminating the need for look-back data. Such a cluster-based approach amplifies the specialization aspect of the models in the ReWTS method. However, for this procedure to adapt to new patterns in the data, it is necessary to be able to assign new clusters of sequences online and fit corresponding forecasting models. How and when to assign new clusters is left as future research.

8 Conclusion

The ReWTS ensemble model for multi-step forecasting is introduced in this paper. For simulated data, where the data-generating processes are known and recurrent, we show that the ReWTS ensemble model outperforms a global model trained with all data in one go. The same applies for noisy, complex and heterogeneous process industry data, as observed on the wastewater treatment dataset (Veas). Here, we achieved a noteworthy reduction in the mean squared forecasting error, ranging from 10% to 30%, across diverse model architectures. For the much less heterogeneous drinking water dataset (Bergen), the ReWTS method cannot be said to outperform the global method as a whole, as judged by the Wilcoxon signed-rank test. The reduction in the mean squared forecasting error nonetheless ranges from 30 % to 75%, where much of the difference is due to the measurement errors occurring in this dataset around chunks 19-21. This suggest that the ReWTS method demonstrates increased resilience to outliers, an important characteristic when operating online. The results on the publicly available datasets highlight that ReWTS is not universally superior: Global models dominate on AQ and BMAQ, while ReWTS proves competitive or superior on ELEC2 and IHEPC. The overall pattern suggests that ReWTS is particularly valuable in non-stationary environments with heterogeneous dynamics and concept drift, but less advantageous in more homogeneous datasets. Beyond performance, the weighting procedure of ReWTS provides interpretability by revealing which historical dynamics are most relevant to a given forecast. Future work should investigate alternative or adaptive chunking strategies, more systematic analysis to characterize the conditions under which the ReWTS ensemble outperforms global models, and computational trade-offs to further establish the role of ReWTS in the broader forecasting landscape and in real-time settings with data streams.

9 Compliance with Ethical Standards

This submission includes research using data from a wastewater treatment plant and a drinking water treatment plant. Access and data collection adhered to the procedures required by the data providers to ensure secure and safe data handling.

10 Author Contribution Statement

Pål V. Johnsen: Conceptualization, methodology, coding, data curation, writing, revision. Eivind Bøhn: Coding, analysis, methodology, writing, revision. Sølve Eidnes: Methodology, analysis, writing, revision. Filippo Remonato: Methodology, analysis, writing, revision. Signe Riemer-Sørensen: Methodology, analysis, writing, revision.

11 Data Availability and Access

The source code used for providing the results in this work is openly available at https://github.com/SINTEF/rewts. The results of the simulation data described in Section 5.3 can be reproduced as detailed in the README file. Also described in the README file is how to deploy the ReWTS ensemble model on the ELEC2 dataset.

The drinking water treatment data used in this project require handling with utmost sensitivity, given their direct implications for public health, safety, and privacy. Access to the data platforms from which this project has sourced its information may be granted upon formal request from qualified individuals or organizations, specifically for research purposes or with expediency justified by the data providers. Requests for access should adhere to established ethical and legal considerations. Interested parties are invited to contact Bergen Vann for further details, and to initiate the request process.

Comprehensive understanding and appropriate utilization of wastewater treatment data necessitate additional knowledge about Veas. Veas ensures both this knowledge and adherence to ethical and legal guidelines by sharing data through agreement-based methods. If you wish to access data for development or research purposes, feel free to reach out to Veas.

Acknowledgments

This research has been funded by The Research Council of Norway, grants 309517 (INVAPRO), 294544 (TAPI), as well as by the EU HORIZON-IA program, grant 101135932 (FAITH).

References

T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama. 2019. "Optuna: A Next-generation Hyperparameter Optimization Framework." In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (KDD '19). Association for Computing Machinery, New York, NY, USA, 2623–2631. ISBN: 9781450362016. doi:10.1145/3292500.3330701.

M. S. Andersen, J. Dahl, and L. Vandenberghe. 2013. *CVXOPT: A Python package for convex optimization, version 1.1.6.* http://cvxopt.org. (2013).

K. Åström and T. Hägglund. 2001. "The future of PID control." *Control Engineering Practice*, 9, 11, 1163–1175. PID Control. doi:https://doi.org/10.1016/S0967-0661(01)00062-4.

S. Bai, J. Z. Kolter, and V. Koltun. 2018. *An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling.* (2018). arXiv: 1803.01271.

B. W. Bequette. 2010. *Process control: modeling, design, and simulation.* (9th print ed.). Prentice-Hall international series in the physical and chemical engineering sciences. Prentice Hall PTR, Upper Saddle River, NJ. ISBN: 0-13-353640-8.

D. Brzezinski and J. Stefanowski. Jan. 2014. "Reacting to Different Types of Concept Drift: The Accuracy Updated Ensemble Algorithm." *IEEE Transactions on Neural Networks and Learning Systems*, 25, 1, (Jan. 2014), 81–94. doi:10.1109/TNNLS.2013.2251352.

S. Chen. 2017. *Beijing Multi-Site Air Quality.* UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5RK5G. (2017).

T. Chen and C. Guestrin. Aug. 2016. "XGBoost: A Scalable Tree Boosting System." In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (KDD '16). Association for Computing Machinery, New York, NY, USA, (Aug. 2016), 785–794. doi:10.1145/2939672.2939785.

R. Elwell and R. Polikar. Oct. 2011. "Incremental Learning of Concept Drift in Nonstationary Environments." *IEEE Transactions on Neural Networks*, 22, 10, (Oct. 2011), 1517–1531. doi:10.1109/TNN.2011.2160459.

A. Galicia, R. Talavera-Llames, A. Troncoso, I. Koprinska, and F. Martínez-Álvarez. Jan. 2019. "Multi-step forecasting for big data time series based on ensemble learning." *Knowledge-Based Systems*, 163, (Jan. 2019), 830–841. doi:10.1016/j.knosys.2018.10.009.

M. Harries. 1999. *Splice-2 Comparative Evaluation: Electricity Pricing.* Tech. rep. Electronic resource. University of New South Wales, School of Computer Science and Engineering, Sydney.

J. Herzen et al.. 2022. "Darts: User-Friendly Modern Machine Learning for Time Series." *Journal of Machine Learning Research*, 23, 124.

B. Krawczyk, L. L. Minku, J. Gama, J. Stefanowski, and M. Woźniak. Sept. 2017. "Ensemble learning for data stream analysis: A survey." *Information Fusion*, 37, (Sept. 2017), 132–156. doi:10.1016/j.inffus.2017.02.004.

B. Lim, S. Ö. Arık, N. Loeff, and T. Pfister. Oct. 2021. "Temporal Fusion Transformers for interpretable multi-horizon time series forecasting." *International Journal of Forecasting*, 37, 4, (Oct. 2021), 1748–1764. doi:10.1016/j.ijforecast.2021.03.012.

S. Masoudnia and R. Ebrahimpour. 2014. "Mixture of experts: a literature survey." *Artificial Intelligence Review*, 42, 275–293. doi:10.1007/s10462-012-9338-y.

M. McCloskey and N. J. Cohen. 1989. "Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem." In: *Psychology of Learning and Motivation*. Vol. 24. Elsevier, 109–165. doi:10.1016/S0079-7421(08)60536-8.

M. Nauman, W. Shireen, and A. Hussain. Jan. 2022. "Model-Free Predictive Control and Its Applications." *Energies*, 15, 14, (Jan. 2022), 5131. doi:10.3390/en15145131.

J. Nocedal and S. J. Wright. 2006. *Numerical Optimization*. (2nd ed.). Springer, New York, NY. Chap. 16. ISBN: 0-387-30303-0.

D. Opitz and R. Maclin. Aug. 1999. "Popular Ensemble Methods: An Empirical Study." *Journal of Artificial Intelligence Research*, 11, (Aug. 1999), 169–198. doi:10.1613/jair.614.

M. Schwenzer, M. Ay, T. Bergs, and D. Abel. Nov. 2021. "Review on model predictive control: an engineering perspective." *The International Journal of Advanced Manufacturing Technology*, 117, 5, (Nov. 2021), 1327–1349. doi:10.1007/s00170-021-07682-3.

R. C. Staudemeyer and E. R. Morris. Sept. 2019. *Understanding LSTM – a tutorial into Long Short-Term Memory Recurrent Neural Networks*. (Sept. 2019). arXiv: 1909.09586.

W. N. Street and Y. Kim. Aug. 2001. "A streaming ensemble algorithm (SEA) for large-scale classification." In: *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining* (KDD '01). Association for Computing Machinery, New York, NY, USA, (Aug. 2001), 377–382. doi:10.1145/502512.502568.

S. Vito. 2008. *Air Quality*. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C59K5F. (2008).

H. Wang, W. Fan, P. S. Yu, and J. Han. Aug. 2003. "Mining concept-drifting data streams using ensemble classifiers." In: *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, (Aug. 2003), 226–235. doi:10.1145/956750.956778.

O. Yadan. 2019. *Hydra - A framework for elegantly configuring complex applications*. Github. (2019). https://github.com/facebookresearch/hydra.

H. Zou and T. Hastie. Apr. 2005. "Regularization and Variable Selection Via the Elastic Net." *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 67, 2, (Apr. 2005), 301–320. doi:10.1111/j.1467-9868.2005.00503.x.

A ReWTS Weight Fitting Derivation

In this section we will provide the ReWTS weight fitting procedure, the optimization problem and the corresponding closed-form solution.

We let $\mathcal{M}_k$ denote the set of fitted forecasting models at time t_k. Further, the vector $\mathbf{y}_{(k+1):(k+h)}$ yields the target values from time t_{k+1} to t_{k+h}, and the matrix $M_h(X_{:k}, \mathbf{y}_{:k})$ of dimension $h\times \mid \mathcal{M}_k \mid$ includes the next h predictions for each model at time t_k.

Assume we are at the present time point t_n with a look-back data of length l_b given the most recent data points going back l_b time steps back in time. Then, we want to find the weights, $\mathbf{w}(t_n)$, that satisfies

$$\arg\min_{\mathbf{w}(t_n)} \sum_{k=n-l_b}^{n-h} \left(\mathbf{y}_{(k+1):(k+h)} - M_h\left(X_{:k}, \mathbf{y}_{:k}\right)\mathbf{w}(t_n)\right)^T \left(\mathbf{y}_{(k+1):(k+h)} - M_h\left(X_{:k}, \mathbf{y}_{:k}\right)\mathbf{w}(t_n)\right) \tag{5}$$

$$\text{s.t.} \quad \mathbf{w}(t_n) \geq 0$$
$$\mathbf{1}^T \cdot \mathbf{w}(t_n) = 1.$$

We can rewrite the optimization problem:

$$
\begin{aligned}
&\arg\min_{\mathbf{w}(t_n)} \sum_{k=n-l_b}^{n-h} \left(\mathbf{y}_{(k+1):(k+h)} - M_h\left(X_{:k}, \mathbf{y}_{:k}\right)\mathbf{w}(t_n)\right)^T \left(\mathbf{y}_{(k+1):(k+h)} - M_h\left(X_{:k}, \mathbf{y}_{:k}\right)\mathbf{w}(t_n)\right) \\
&= \arg\min_{\mathbf{w}(t_n)} \sum_{k=n-l_b}^{n-h} \left(\mathbf{y}_{(k+1):(k+h)}\right)^T \left(\mathbf{y}_{(k+1):(k+h)}\right) + \mathbf{w}(t_n)^T M_h(X_{:k}, \mathbf{y}_{:k})^T M_h(X_{:k}, \mathbf{y}_{:k})\mathbf{w}(t_n) \\
&- 2\mathbf{y}_{(k+1):(k+h)}^T M_h(X_{:k}, \mathbf{y}_{:k})\mathbf{w}(t_n) \\
&= \arg\min_{\mathbf{w}(t_n)} \frac{1}{2}\mathbf{w}(t_n)^T \left(\sum_{k=n-l_b}^{n-h} M_h(X_{:k}, \mathbf{y}_{:k})^T M_h(X_{:k}, \mathbf{y}_{:k})\right)\mathbf{w}(t_n) \\
&- \left(\sum_{k=n-l_b}^{n-h} M_h(X_{:k}, \mathbf{y}_{:k})^T \mathbf{y}_{(k+1):(k+h)}\right)^T \mathbf{w}(t_n)
\end{aligned} \tag{6}
$$

$$
\begin{aligned}
&\text{s.t.} \quad \mathbf{w}(t_n) \geq 0 \\
&\quad \mathbf{1}^T \cdot \mathbf{w}(t_n) = 1.
\end{aligned}
$$

Let

$$Q = \sum_{k=n-l_b}^{n-h} M_h(X_{:k}, \mathbf{y}_{:k})^T M_h(X_{:k}, \mathbf{y}_{:k}),$$

a matrix of dimension $\mid \mathcal{M}_k \mid \times \mid \mathcal{M}_k \mid$, and moreover let

$$p = \sum_{k=t-L}^{t-h} M_h(X_{:k}, \mathbf{y}_{:k})^T \mathbf{y}_{(k+1):(k+h)},$$

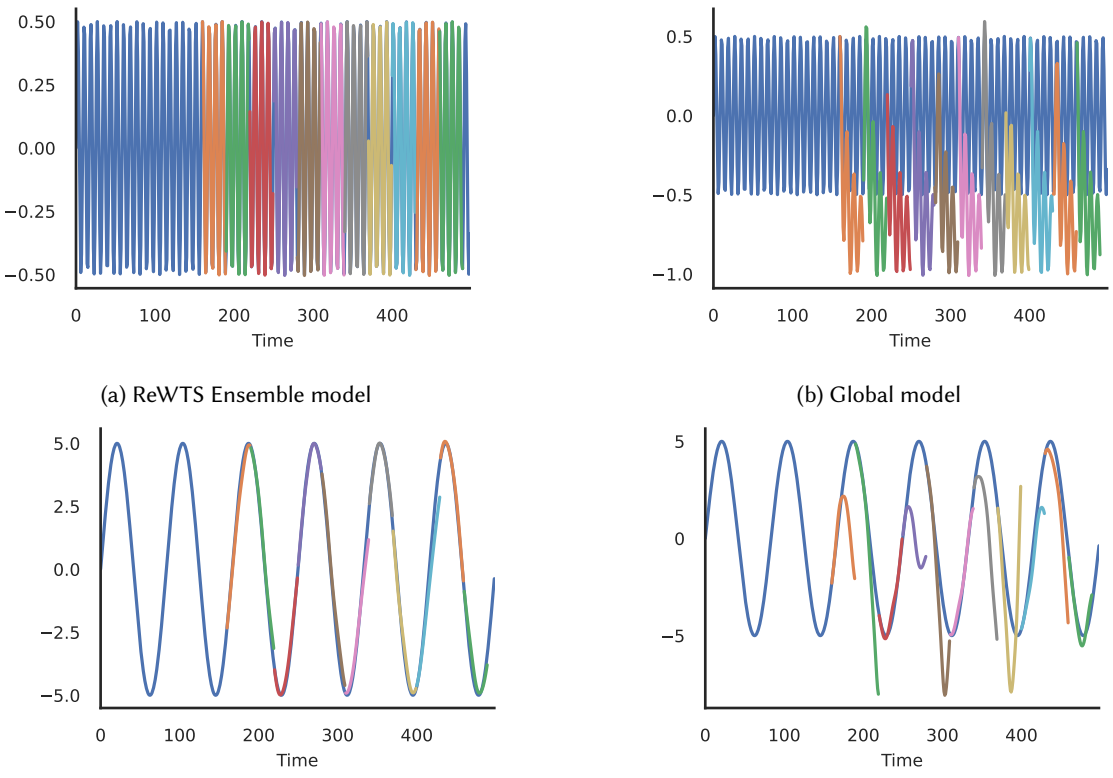
a vector of size $\mid \mathcal{M}_k \mid$, and plug in these variables into the optimization problem (6). The matrix Q is symmetric as it is a sum of symmetric matrices. This structure of the optimization problem, with the fact that Q is symmetric, makes it a *quadratic programming* (QP) optimization problem. Additionally, the matrix Q is positive semidefinite, because it is a sum of positive semidefinite matrices ($M^T M$ positive semidefinite for any matrix M of dimension $m \times n$). In this case, we say that we have a *convex* QP problem for which there exist efficient solvers (Nocedal and Wright 2006). The software used in this work is the CVXOPT package in Python (Andersen et al. 2013).

Both the matrix Q and the vector p are a sum of multiple matrices or vectors respectively, and the number of summands increases proportionally with the look-back length. Depending on the scaling of the target y, some elements in the matrix Q and vector p might therefore become very big in magnitude. This may yield numerical instabilities in the QP-solver, and occasionally non-convergence and ultimately errors. These instabilities can also happen for elements that are very small in magnitude. One simple way to circumvent these potential issues is to rescale the matrix Q and vector p in cases where the element with the maximum value is very small, $\max(Q_{ij}, p_j) < \epsilon$, given by some predefined threshold ϵ, or the element with the maximum value is very large, $\max(Q_{ij}, p_j) > L$, given by some predefined threshold L. In these cases, Q and p in (6) can be rescaled to $Q' = Q/\epsilon, p' = p/\epsilon$ or $Q' = Q/L, p' = p/L$ respectively. Swapping Q, p with Q', p' in the optimization problem (6) will not alter the solution, $\mathbf{w}(t_n)$, of the original QP-problem.

B Forecast on the Simulation Data

Below are the forecasts for all chunks, from the ReWTS ensemble model and the global model for the simulation data explained in Section 6.1 in the main article.

B.1 Training Set



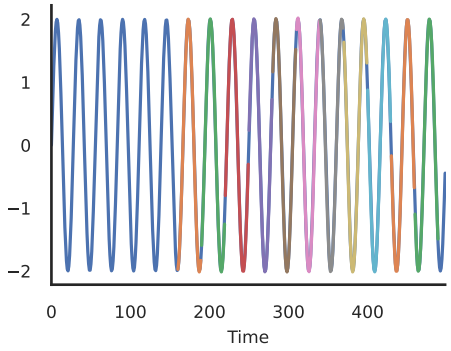
(a) ReWTS Ensemble model

(b) Global model

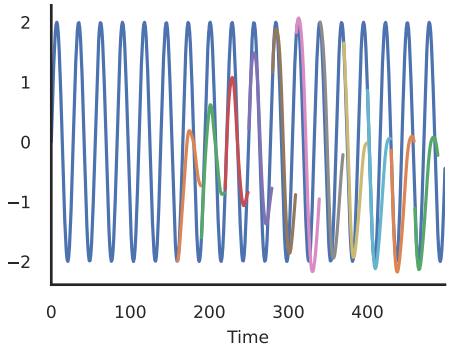
(c) ReWTS Ensemble model

(d) Global model

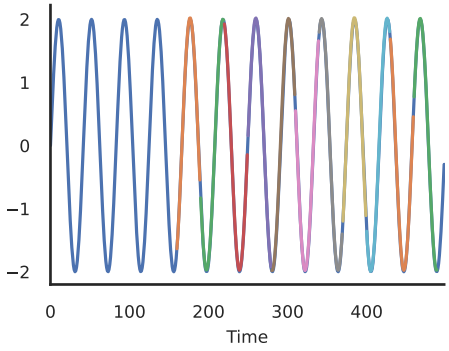
Fig. 15. Training set, chunks 1, 2



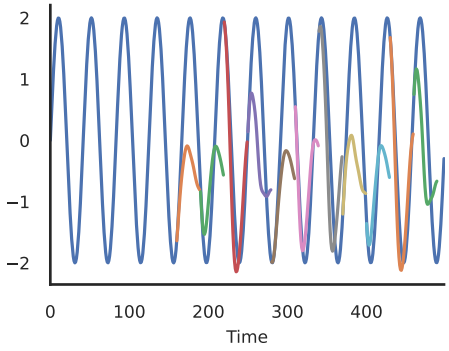
(a) ReWTS Ensemble model



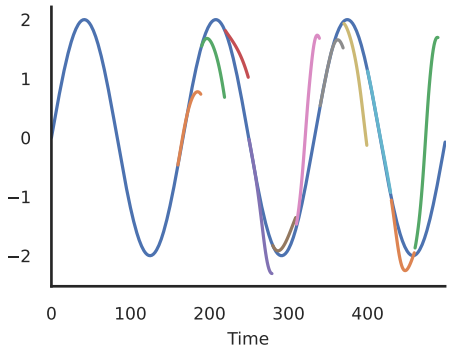
(b) Global model



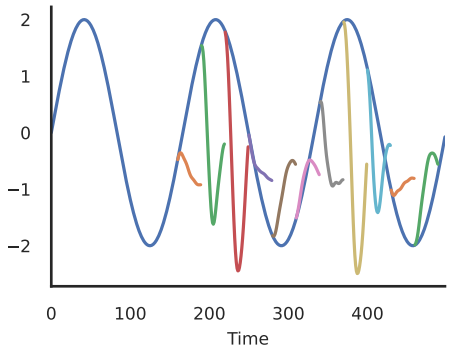
(c) ReWTS Ensemble model



(d) Global model

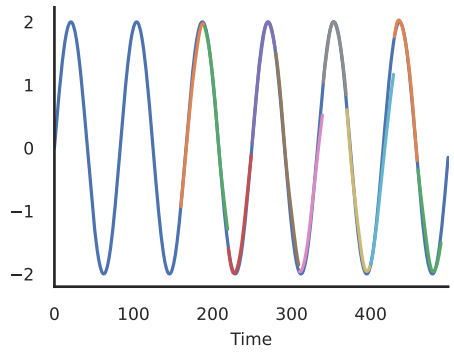


(e) ReWTS Ensemble model

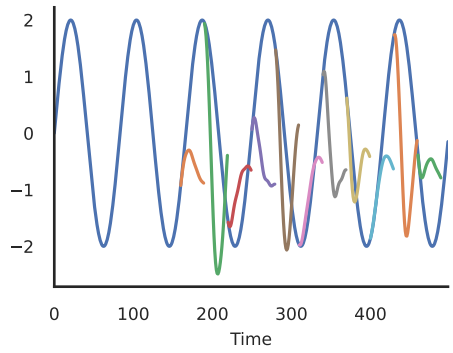


(f) Global model

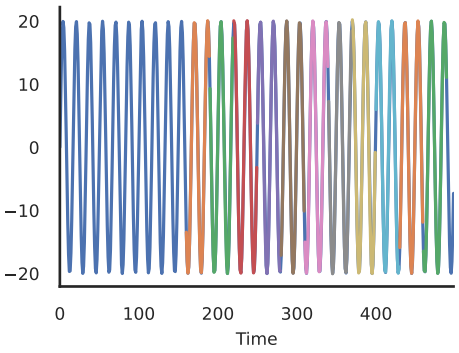
Fig. 16. Training set, chunks 3, 4 and 5



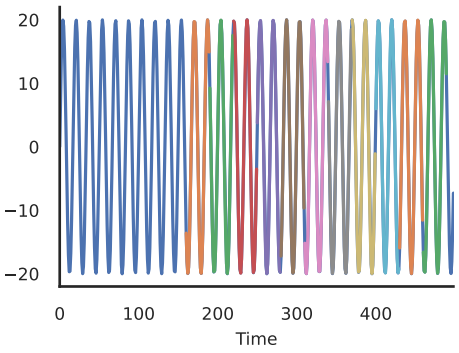
(a) ReWTS Ensemble model



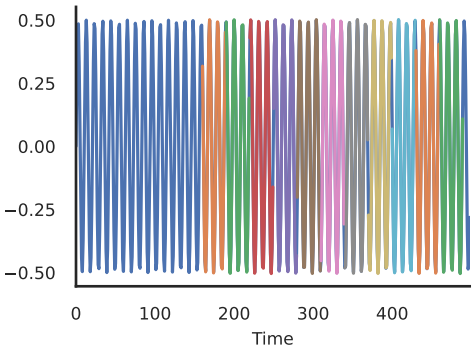
(b) Global model



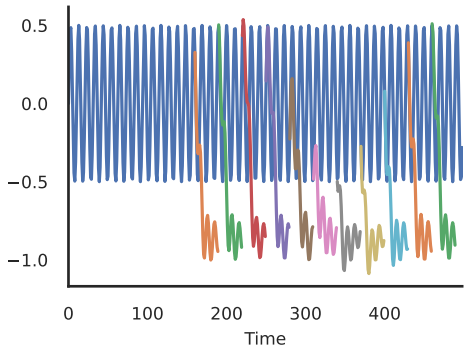
(c) ReWTS Ensemble model



(d) Global model



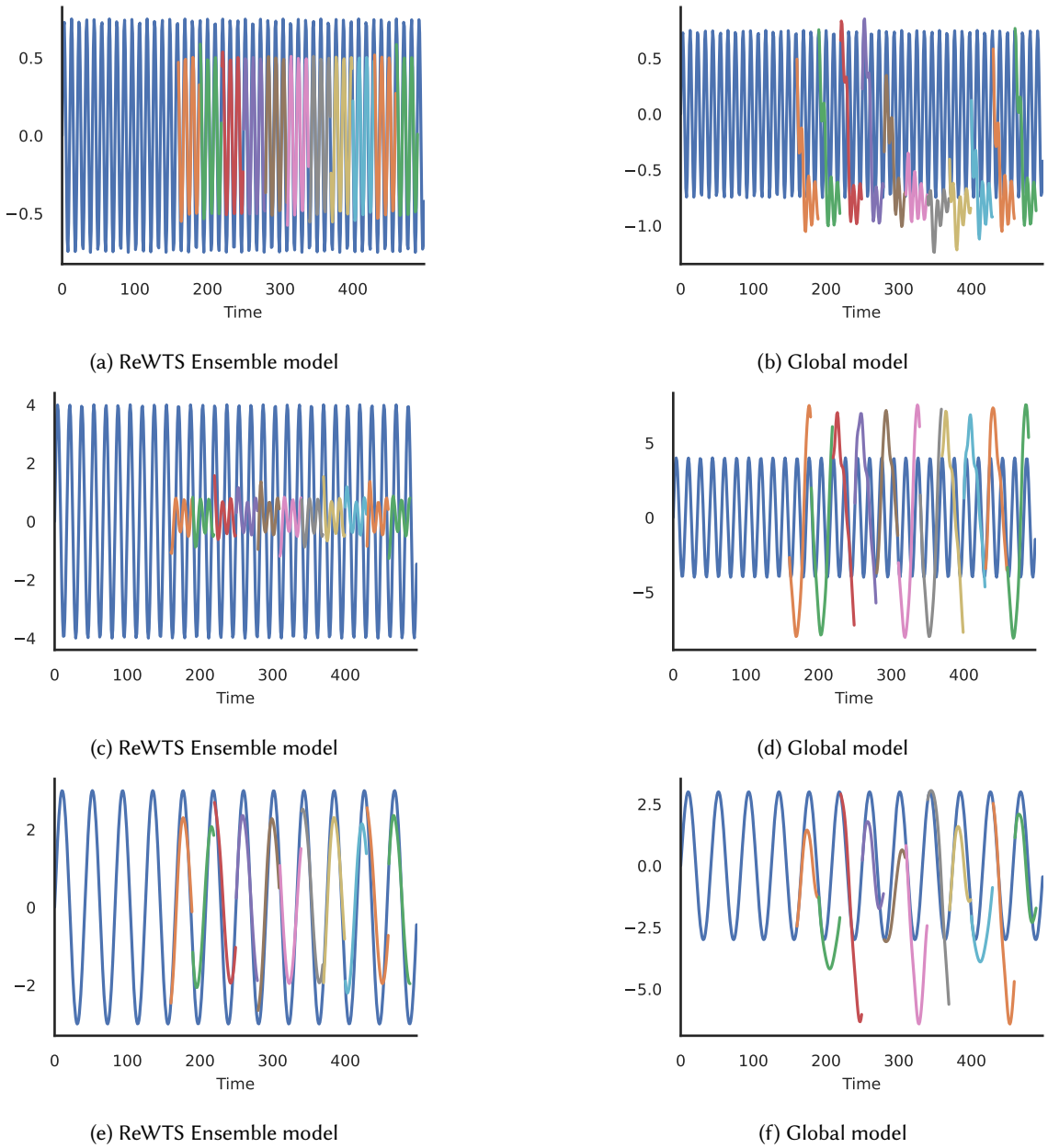
(e) ReWTS Ensemble model



(f) Global model

Fig. 17. Training set, chunks 6, 7 and 8

B.2 Test Set



(a) ReWTS Ensemble model

(b) Global model

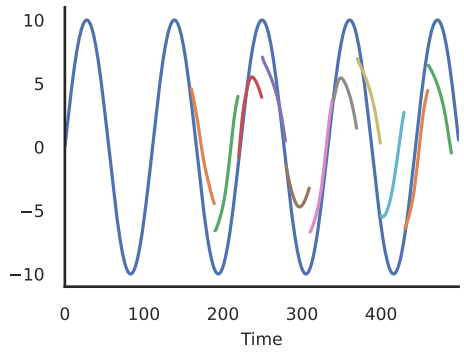
(c) ReWTS Ensemble model

(d) Global model

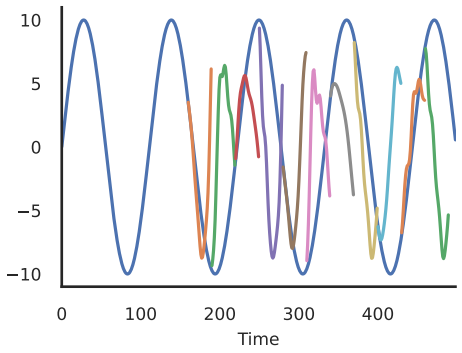
(e) ReWTS Ensemble model

(f) Global model

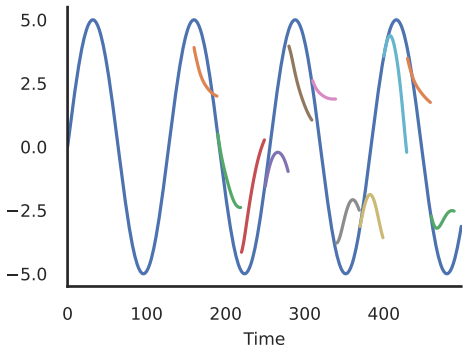
Fig. 18. Test set, chunks 1, 2 and 3



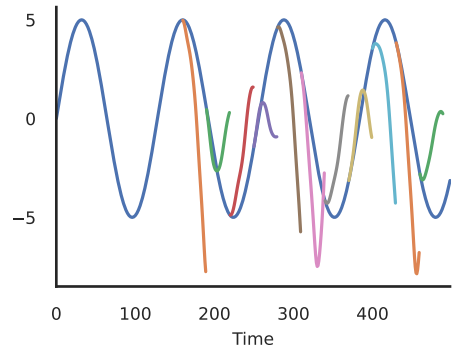
(a) ReWTS Ensemble model



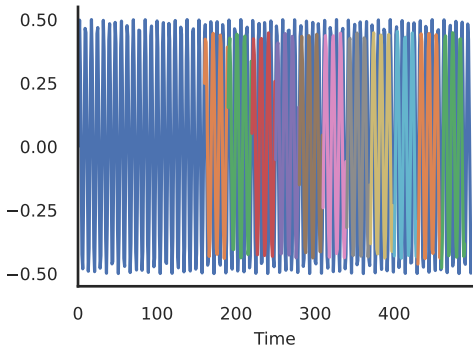
(b) Global model



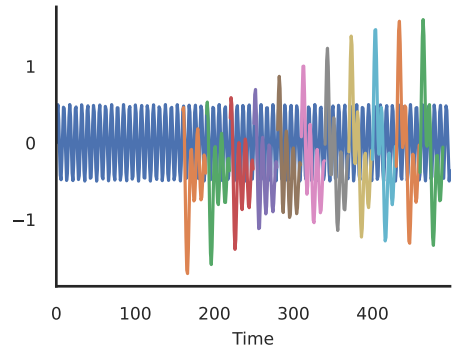
(c) ReWTS Ensemble model



(d) Global model

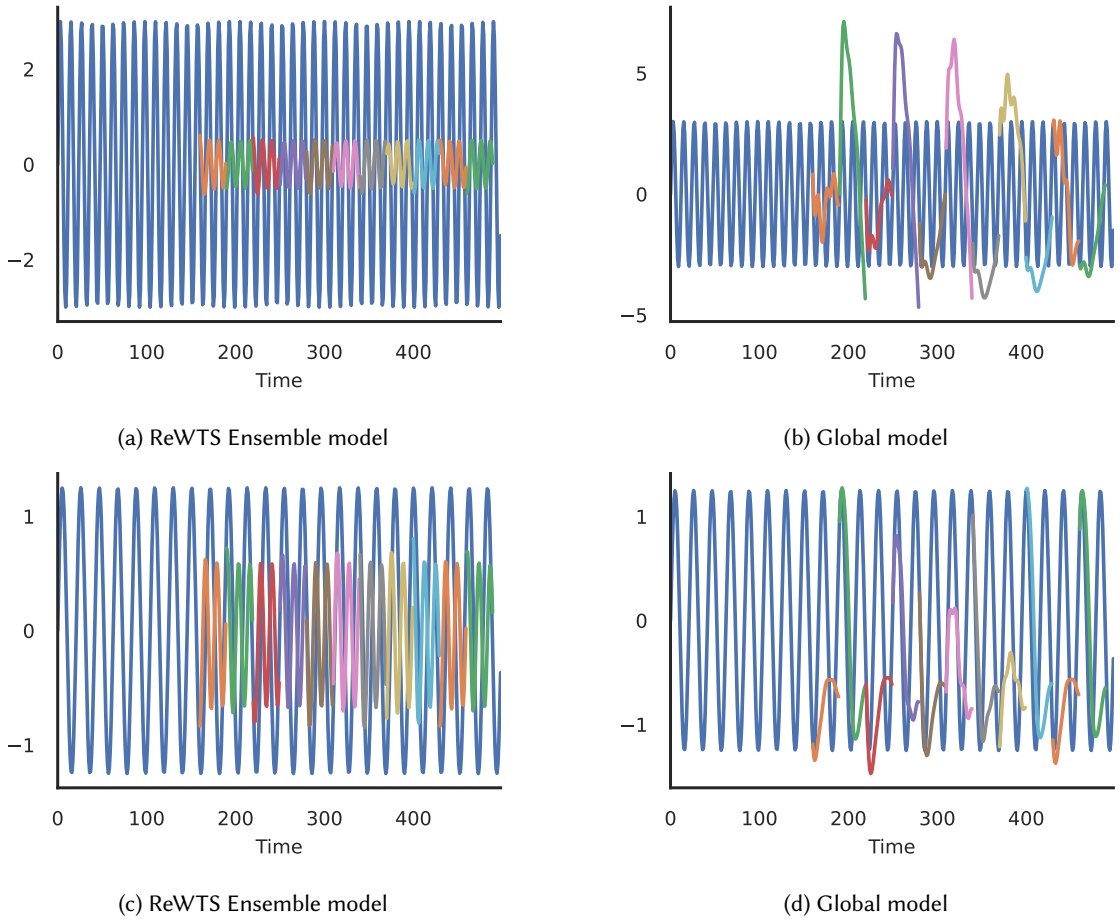


(e) ReWTS Ensemble model



(f) Global model

Fig. 19. Test set, chunks 4, 5 and 6



(a) ReWTS Ensemble model

(b) Global model

(c) ReWTS Ensemble model

(d) Global model

Fig. 20. Test set, chunks 7 and 8

C Results on Drinking Water Treatment Plant

Figure 21 shows the chunk-by-chunk MSE result for the ReWTS ensemble model and the global model, for all model architectures, on the data from the water treatment plant with chunk length of 2016 time points (corresponding to two weeks of data), and a look-back data length of 300 time points (corresponding to two days of data). The story is the same across all the model architectures investigated in this work, namely that the ReWTS ensemble outperforms the global model, with or without up-scaling of trainable parameters.

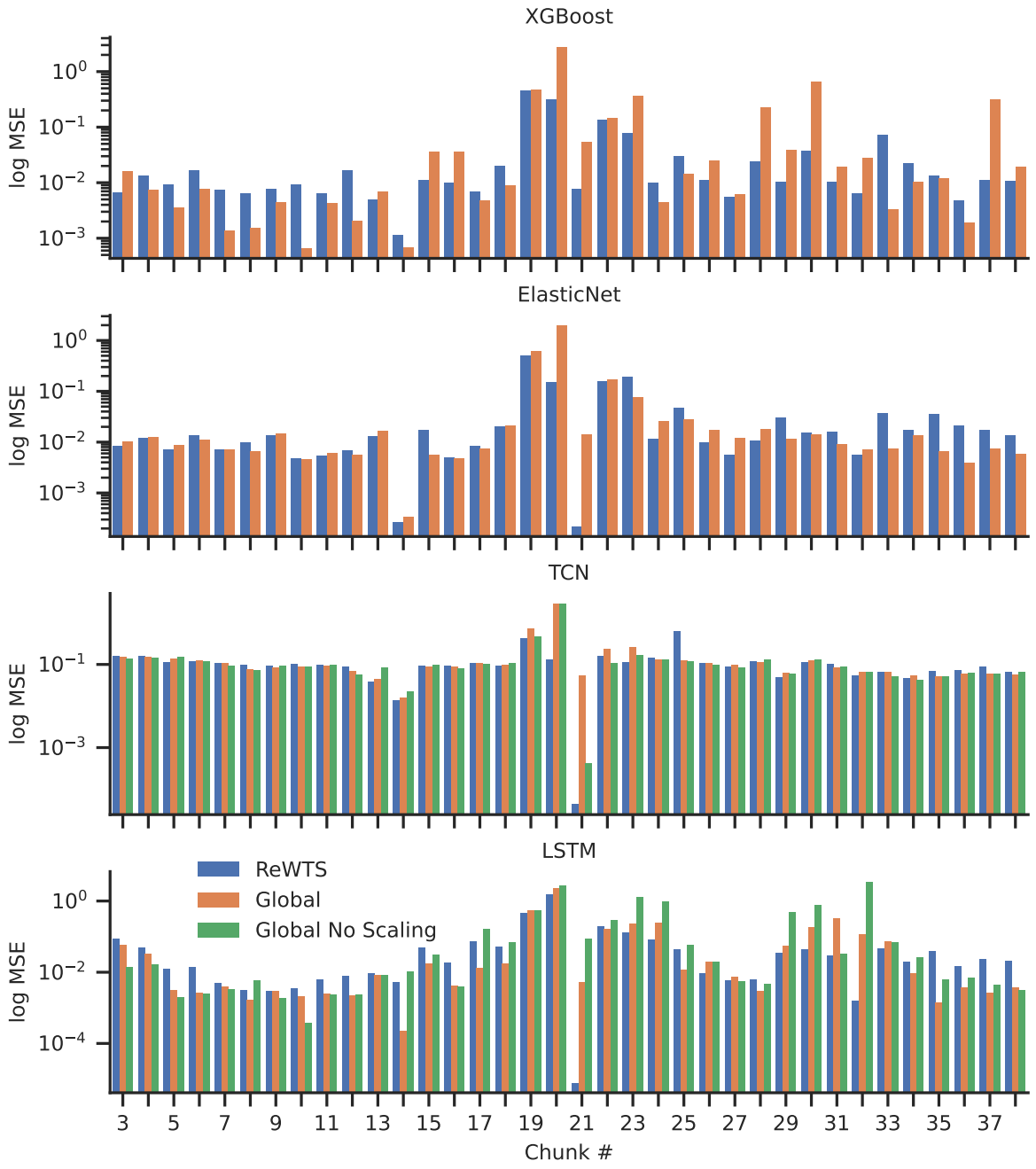


Fig. 21. Comparing the ReWTS ensemble model with the corresponding global model for the wastewater plant data for the model architectures elastic net, XGBoost, TCN and LSTM. The TCN and LSTM models were upscaled for each new chunk such as to match the total number of trainable parameters by the ReWTS ensemble model. For comparison, a global model without upscaling (Global No Scaling) is also presented.

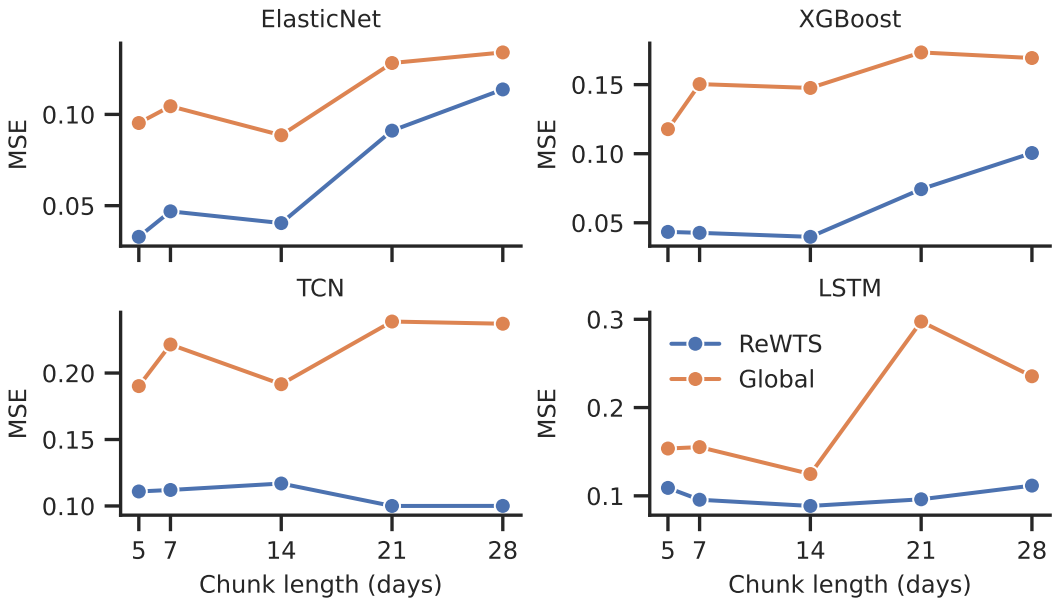


Fig. 22. The average MSE, for different model architectures, of the ReWTS ensemble model for varying chunk length (with a constant look-back length of 300 data points). The results are compared with the corresponding global model.

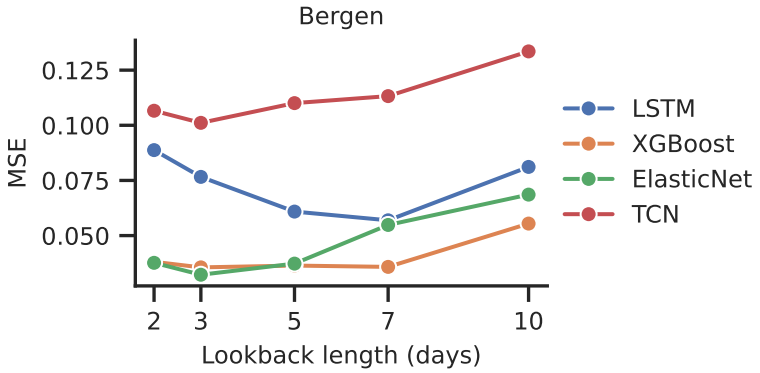


Fig. 23. The average MSE, for different model architectures, of the ReWTS ensemble model for varying look-back length (with constant chunk length equal to two weeks of data). Results are given for the wastewater treatment data.

Figure 22 shows the average MSE of the ReWTS ensemble model for varying chunk lengths when applied on the drinking water treatment dataset. Figure 23 shows the average MSE for varying look-back lengths. Mostly, the trend is the same for both the wastewater and drinking water datasets: Within the observed range, smaller chunks and look-back lengths yield smaller average MSE. Exceptions are the TCN model when varying chunk length, and the LSTM model when varying the look-back length.

D Result on ELEC2 Dataset with Chunk Length of Four Weeks

The results given in Table 5 for the ELEC2 dataset when the chunk length is changed to four weeks (1344 data points) instead of two weeks (672 data points) as given in Table 4 (look-back length still fixed to two days). The results show now that ReWTS outperforms the global model across all model architectures with respect to the average MSE (significant improvement for the Elastic net and XGBoost model according to the Wilcoxon signed-rank test with a significance value of 0.05). This confirms the importance of chunk length and look-back length, and that the choice of these may decide whether ReWTS outperforms the global model or not on a particular dataset.

Table 5. The average MSE over all chunks of the ELEC2 dataset for the global method and ReWTS ensemble when chunk length is four weeks and look-back length is two days. Included is the percentage difference for different model architectures, as well as the one-sided Wilcoxon signed-rank test which infers whether the ReWTS method outperforms the global method in terms of the MSE grouped by chunks.

		Architecture			
Dataset		Elastic net	XGBoost	TCN	LSTM
ELEC2 (45 312 data points, 33 chunks)	ReWTS MSE	**1.28E-03**	**7.57E-04**	**1.56E-03**	**1.18E-03**
	Global MSE	1.35E-03	9.90E-04	1.58E-03	1.44E-03
	% Difference	5.3	26.6	1.2	19.2
	Wilcoxon p-value	2.9E-02	4.35E-03	2.6E-01	3.3E-01

Received 07 November 2024; accepted 16 September 2025