International Journal of Forecasting xxx (xxxx) xxx

Contents lists available at ScienceDirect

International Journal of Forecasting

journal homepage: www.elsevier.com/locate/ijforecast

Hierarchical forecasting at scale☆

Olivier Sprangers[a,*], Wander Wadman[b], Sebastian Schelter[a], Maarten de Rijke[c]

[a] AIRLab, University of Amsterdam, Science Park 900, 1098 XH Amsterdam, The Netherlands
[b] bol.com, Papendorpseweg 100, 3528 BJ Utrecht, The Netherlands
[c] University of Amsterdam, Science Park 900, 1098 XH Amsterdam, The Netherlands

A R T I C L E I N F O

Keywords:
Hierarchical forecasting
Large-scale forecasting
Efficiency in forecasting methods
Hierarchical time series
Grouped time series
Temporal aggregation

A B S T R A C T

Hierarchical forecasting techniques allow for the creation of forecasts that are coherent with respect to a pre-specified hierarchy of the underlying time series. This targets a key problem in e-commerce, where we often find millions of products across many product hierarchies, and forecasts must be made for individual products and product aggregations. However, existing hierarchical forecasting techniques scale poorly when the number of time series increases, which limits their applicability at a scale of millions of products.

In this paper, we propose to learn a coherent forecast for millions of products with a single bottom-level forecast model by using a loss function that directly optimizes the hierarchical product structure. We implement our loss function using sparse linear algebra, such that the number of operations in our loss function scales quadratically rather than cubically with the number of products and levels in the hierarchical structure. The benefit of our sparse hierarchical loss function is that it provides practitioners with a method of producing bottom-level forecasts that are coherent to any chosen cross-sectional or temporal hierarchy. In addition, removing the need for a post-processing step as required in traditional hierarchical forecasting techniques reduces the computational cost of the prediction phase in the forecasting pipeline and its deployment complexity.

In our tests on the public M5 dataset, our sparse hierarchical loss function performs up to 10% better as measured by RMSE and MAE than the baseline loss function. Next, we implement our sparse hierarchical loss function within a gradient boosting-based forecasting model at bol.com, a large European e-commerce platform. At bol.com, each day, a forecast for the weekly demand of every product for the next twelve weeks is required. In this setting, our sparse hierarchical loss resulted in an improved forecasting performance as measured by RMSE of about 2% at the product level, compared to the baseline model, and an improvement of about 10% at the product level as measured by MAE. Finally, we found an increase in forecasting performance of about 5%–10% (both RMSE and MAE) when evaluating the forecasting performance across the cross-sectional hierarchies we defined. These results demonstrate the usefulness of our sparse hierarchical loss applied to a production forecasting system at a major e-commerce platform.

☆ This research was (partially) funded by Ahold Delhaize, the Hybrid Intelligence Center, a 10-year program funded by the Dutch Ministry of Education, Culture and Science through the Netherlands Organisation for Scientific Research under grant no. NWO 24.004.022, Project LESSEN with project number NWA.1389.20.183 of the research program NWA ORC 2020/21, which is (partly) financed by the Dutch Research Council (NWO), and Project FINDHR that received funding from the European Union's Horizon Europe research and innovation program under grant agreement no. 101070212.

* Corresponding author.
E-mail addresses: o.r.sprangers@uva.nl (O. Sprangers), wwadman@bol.com (W. Wadman), s.schelter@uva.nl (S. Schelter), m.derijke@uva.nl (M. de Rijke).

https://doi.org/10.1016/j.ijforecast.2024.02.006

1. Introduction

In e-commerce, we are often faced with two forecasting challenges. First, forecasts at the lowest granularity – usually the individual product level – are required. Still, we also need forecasts at higher granularities, for example, at the category, department, or regional level, as higher-level forecasts are often required in logistics and financial planning. Second, forecasts at different time granularities are necessary, for example, daily or weekly forecasts. It is common that separate forecast models are made for each separate (temporal) granularity, and as such, these forecasts may not be coherent with each other. Hierarchical forecasting (Hyndman, Ahmed, Athanasopoulos, & Shang, 2011) and temporal hierarchical forecasting techniques (Athanasopoulos, Hyndman, Kourentzes, & Petropoulos, 2017; Rangapuram et al., 2023; Theodosiou & Kourentzes, 2021) aim to solve the problem of creating forecasts that are coherent with respect to a pre-specified cross-sectional and/or temporal hierarchy of the underlying time series.

Challenges with existing cross-sectional and temporal hierarchical forecasting techniques. Reconciliation methods adjust the forecasts for each level in the hierarchy by minimizing the errors at each forecast level. These methods are applied as a post-processing step that requires a matrix inversion that scales cubically with the number of products or product hierarchies (Athanasopoulos et al., 2017; Hyndman et al., 2011; Wickramasuriya, Athanasopoulos, & Hyndman, 2019). In settings with millions of products, such as e-commerce, this becomes computationally expensive at prediction time. Neural network methods can optimize for the hierarchy in an end-to-end manner; however, these are either multivariate methods that scale poorly to millions of time series (Rangapuram et al., 2021) or they can only optimize for the temporal hierarchy (Rangapuram et al., 2023).

Sparse loss function. To overcome these scaling issues, we design a sparse *hierarchical loss* (HL) function that directly optimizes both cross-sectional and temporal hierarchical structures. Our corresponding sparsity-aware implementation ensures that the number of operations in our loss function scales quadratically rather than cubically with the number of products and levels in the hierarchical structure, enabling computationally efficient training. The benefit of our sparse hierarchical loss function is that it provides practitioners with a method of producing bottom-level forecasts that are coherent to any chosen cross-sectional and temporal hierarchy. In addition, removing the need for a post-processing step as used in traditional hierarchical forecasting techniques reduces the computational cost of the prediction phase in the forecasting pipeline. Furthermore, this also reduces the deployment complexity of the forecasting pipeline.

Evaluation. We evaluate our sparse HL function on a gradient-boosted forecasting system on the public M5 dataset (Makridakis, Spiliotis, & Assimakopoulos, 2022) and a proprietary dataset from our e-commerce partner. For the M5 dataset, we demonstrate that our implementation provides up to 10% better forecasting performance as measured by both RMSE and MAE compared with (i) reconciliation methods and (ii) baseline bottom-level forecasting methods that use a standard loss function. For the proprietary dataset, we present the results of an offline test on the product-level forecast system of bol.com, a European e-commerce company with a catalogue of millions of unique products. Our sparse HL function improves the forecasting performance by about 2% on RMSE and 10% on MAE compared to the baseline forecasting system. This demonstrates the usefulness of our sparse HL function in a large-scale setting.

Contributions. In summary, the main contributions of this paper are:

1. We design a sparse hierarchical loss function that enables direct end-to-end training of cross-sectional and temporal hierarchical forecasts in large-scale settings in Section 4.
2. We empirically demonstrate that our sparse hierarchical loss function can outperform existing hierarchical forecasting reconciliation methods by up to 10% in Section 5.1. Contrary to most end-to-end hierarchical forecasting methods that leverage neural networks (Rangapuram et al., 2023, 2021), we use LightGBM (Ke et al., 2017) as our base forecasting model. This highly popular gradient boosting-based forecasting method is widely used in industry (Januschowski et al., 2022) and was used by the majority of the top performing solutions in the M5 forecasting competition (Makridakis et al., 2022).
3. We show how our sparse hierarchical loss function scales to large-scale settings and demonstrate a reduction of training and prediction time of up to an order of magnitude compared to the best hierarchical forecasting reconciliation methods (Section 5.1).
4. We present the results of an offline test of our method for the primary product demand forecasting model at bol.com, a European e-commerce company with a catalogue of millions of unique products, demonstrating an improvement of 2% on RMSE and 10% on MAE as compared to the baseline forecasting system, in Section 5.2.

2. Related work

Forecasting for large-scale settings. Contemporary large-scale forecasting applications require forecasting many time series concurrently (Böse et al., 2017). In academia, there has been a surge in the use of neural network-based forecasting methods, which commonly learn a single forecast model that can produce forecasts for many time series. We refer the interested reader to the recent survey of Benidis et al. (2023) for an overview of these methods. However, tree-based methods topped the M5 forecasting competition (Makridakis et al., 2022), which is believed to be due to the strong implementations available of these algorithms (Januschowski et al., 2022), such as the LightGBM (Ke et al., 2017) or XGBoost (Chen & Guestrin, 2016) packages. Our own experience within bol.com confirms this view: the ease of use, execution speed and strong default performance are key reasons a tree-based method is often the default choice when creating a new forecasting model.

Hierarchical forecasting. Hierarchical forecasting (Ben Taieb & Koo, 2019; Ben Taieb, Taylor, & Hyndman, 2017; Hyndman et al., 2011; Hyndman, Lee, & Wang, 2016; Wickramasuriya et al., 2019) and temporal hierarchical forecasting techniques (Athanasopoulos et al., 2017; Ben Taieb, 2017; Rangapuram et al., 2023; Theodosiou & Kourentzes, 2021) aim to solve the problem of creating forecasts that are coherent with respect to a pre-specified cross-sectional and/or temporal hierarchy of the underlying time series. We divide hierarchical forecasting methods into *Reconciliation methods* and *Other methods*.

Reconciliation methods. For a detailed overview of reconciliation methods, we refer the interested reader to the recent survey of Athanasopoulos, Hyndman, Kourentzes, and Panagiotelis (2024). Reconciliation methods solve the hierarchical forecasting problem as a post-processing step by reconciling the forecasts to a pre-specified cross-sectional and/or temporal hierarchy (Ben Taieb & Koo, 2019; Ben Taieb et al., 2017; Girolimetto & Di Fonzo, 2023; Hyndman et al., 2011, 2016; Panagiotelis, Athanasopoulos, Gamakumara, & Hyndman, 2021; Wickramasuriya et al., 2019). Limitations of these approaches are (i) they require a post-processing step, (ii) computing the reconciliation may be computationally expensive, as we show in Section 3.2, and (iii) approaches that are computationally less expensive tend to perform worse, as we show in Section 5. Recent work by Ben Taieb (2017) and Ben Taieb and Koo (2019) has improved the forecasting performance of previous reconciliation approaches but at the expense of even higher computational costs, as we explain in Section 3.

Other methods. In Rangapuram et al. (2023, 2021), neural network-based end-to-end hierarchical probabilistic forecasting methods are proposed to solve the hierarchical forecasting problem. More recently and most closely related to our work, Han, Dasgupta, and Ghosh (2021) introduced SHARQ, a method that reconciles probabilistic hierarchical forecasts during training by employing a regularized loss function that aims to improve hierarchical consistency of bottom-up forecasts through regularization. However, the regularization does not strictly enforce the cross-sectional hierarchy in this method.

3. Background

To understand our problem setting and the issues we identify with existing hierarchical forecasting methods, we introduce the hierarchical forecasting problem and common methods of solving the hierarchical forecasting problem.

3.1. Problem definition

Suppose we have n time series written as $\mathbf{y}_t \in \mathbb{R}^n$, where t denotes the time stamp. We are interested in finding h-step ahead estimates $\hat{\mathbf{y}}_h$ of the time series $\mathbf{y}_{T+h}$ using past values $\mathbf{y}_1, \ldots, \mathbf{y}_T$. In our hierarchical forecasting setting, we aim to concurrently create forecasts for many time series whilst adhering to pre-specified hierarchical relationships between the time series. This can be formalized as follows (Athanasopoulos et al., 2024; Hyndman & Athanasopoulos, 2021):

$$\tilde{\mathbf{y}}_h = SG\hat{\mathbf{y}}_h \;, \tag{1}$$

where $S \in \{0, 1\}^{n \times n_b}$ is a matrix that defines the hierarchical relationship between the n_b bottom-level time series and the $n_a = n - n_b$ aggregations, $G \in \mathbb{R}^{n_b \times n}$ is a matrix that encapsulates the contribution of each forecast to the final estimate, and $\tilde{\mathbf{y}}_h \in \mathbb{R}^n$ is the vector of forecasts adjusted for the hierarchy. We can use the matrix G to define various forecast contribution scenarios. Note that we can straightforwardly extend Eq. (1) to the setting of *temporal hierarchies* (Athanasopoulos et al., 2017; Rangapuram et al., 2023) by considering forecasts of different time granularities in our vector of base forecasts $\hat{\mathbf{y}}_h$ and using an appropriate choice of S to aggregate series of a different time granularity. We will show how cross-sectional and temporal hierarchical forecasting can be combined in Section 4.

The optimal solution to the problem in Eq. (1) can be found using *Reconciliation methods* and *Other methods*.

Reconciliation methods. *MinTShrink* (Athanasopoulos et al., 2024; Wickramasuriya et al., 2019) and variants find the optimal G matrix by solving a minimization problem that has the following solution (ref. Theorem 1 of Wickramasuriya et al., 2019):

$$G = (J - JWU(U^T WU)^{-1}U^T) \;, \tag{2}$$

in which S is partitioned as $S^T = [C^T \; I_{n_b}]$, $J = [0_{n_b \times n_a} \; I_{n_b}]$, $U^T = [I_{n_a} \; -C]$. In *MinTShrink*, W is estimated using the shrunk empirical covariance estimate of Schäfer and Strimmer (2005). Simpler choices for W, such as the identity matrix, reduce the solution to the *Ordinary Least Squares (OLS)* solution of Hyndman et al. (2011). In *ERM*, Ben Taieb and Koo (2019) note than *MinTShrink* and variants rely on the assumption of unbiasedness of the base forecasts. They relax this assumption by formulating the hierarchical reconciliation problem as an *Empirical Risk Minimization* problem, introducing the *ERM* method. In addition, they propose two regularized variants of *ERM* aimed at reducing forecast variance.

Other methods. *Hier-E2E* (Rangapuram et al., 2021) solves the problem of Eq. (1) by learning a neural network model that combines the forecasting and reconciliation step in a single model, resulting in an end-to-end solution removing the need for a post-processing step. Similarly, *COPDeepVAR* (Rangapuram et al., 2023) is an end-to-end neural network method that enforces temporal hierarchies; however, this is a univariate method that is not able to implement structural hierarchies (i.e., cross-sectional hierarchies) simultaneously, and therefore not suited to our task. *SHARQ* (Han et al., 2021) also moves the reconciliation step into the training phase and achieves reconciliation using a regularized loss function, where the regularization enforces the coherency. However, this method does not implement absolute coherency in the hierarchy.

3.2. Scaling issues of hierarchical forecasting methods

Our main motivation for this paper is the limitations of prior work for problem settings with many time series.

Scaling issues with reconciliation methods. In reconciliation methods, we encounter the following issues when scaling to many time series:

- The reconciliation is performed as a *post-processing* step and thus has to be performed as an additional step after generating the base forecasts. Even though G in Eq. (1) needs to be computed only once using Eq. (2), the reconciliation still needs to be performed after each base forecast is produced. Also, G ideally is sparse (Ben Taieb & Koo, 2019). Still, no reconciliation method guarantees this, so computing Eq. (1) will generally be a dense matrix–vector product that scales with the number of time series.
- For *MinTShrink* (Wickramasuriya et al., 2019), estimating W according to the method of Schäfer and Strimmer (2005) is computationally expensive, with a computational complexity of $O(Nn^2)$, where N denotes the number of training samples used to compute the shrunk covariance estimate. In addition, the shrunk covariance estimate of Schäfer and Strimmer (2005) is not guaranteed to give consistent results in high-dimensional settings (Touloumis, 2015), making it less applicable for problem settings with many time series. Finally, the estimate for W will generally be a dense matrix, so we cannot use efficient sparse algorithms to solve Eq. (2). However, even for simpler, sparse choices of W (such as the identity matrix of *OLS* (Hyndman et al., 2011)), we still need to invert a matrix of size $n_a \times n_a$ to solve Eq. (2), which becomes computationally costly for problems with many aggregations, which naturally arise in retail forecasting scenarios. For example, for the M5 retail forecasting competition (Makridakis, Spiliotis, & Assimakopoulos, 2021), $n_a = 12{,}350$, even though there are only 3049 unique products in this dataset.
- For *ERM* and its regularized variants (Ben Taieb & Koo, 2019), we need to either invert multiple dense matrices that scale quadratically with the number of time series or we need to compute a Kronecker product that scales quadratically with the number of time series, followed by an expensive lasso search procedure. Improving the computational complexity of the *ERM* methods is also mentioned in Ben Taieb and Koo (2019) as an avenue for future work.

Scaling issues with other methods. *Hier-E2E* (Rangapuram et al., 2021) is a multivariate method, which means both input and output of the neural network scale with the number of time series. This significantly adds to the training and parameter costs for neural networks, as many parameters are required to handle all the separate time series. This, in turn, requires GPUs with more memory to train these models, which increases operating costs.

4. Sparse hierarchical loss

This section presents our main technical contribution, the sparse hierarchical loss. First, we show how cross-sectional and temporal hierarchical forecasting can be combined. Then, we introduce and demonstrate our loss function via a toy example.

Combining cross-sectional and temporal hierarchical forecasting. We are interested in finding forecasts that can be aggregated according to a pre-specified cross-sectional hierarchy $S^{cs} \in \{0,1\}^{n^{cs} \times n_b^{cs}}$ and temporal hierarchy $S^{te} \in \{0,1\}^{n^{te} \times n_b^{te}}$:

$$\tilde{\mathbf{y}}_h^{cs} = S^{cs} G^{cs} \hat{\mathbf{y}}_h^{cs} \, , \tag{3}$$

$$\tilde{\mathbf{y}}^{te} = S^{te} G^{te} \hat{\mathbf{y}}^{te} \, . \tag{4}$$

These equations can be interpreted as follows:

- In Eq. (3), we aggregate n_b^{cs} bottom-level time series from the same forecast h across a set of $n^{cs} = n_b^{cs} + n_a^{cs}$ cross-sectional aggregations.
- In Eq. (4), we aggregate each time series consisting of n_b^{te} timesteps across a set of $n^{te} = n_b^{te} + n_a^{te}$ temporal aggregations, hence we drop the subscript h.

We will only create bottom-level forecasts, thus $G^{cs} = [0_{n_b^{cs}} \times n_a^{cs} \; I_{n_b^{cs}}]$ and $G^{te} = [0_{n_b^{te}} \times n_a^{te} \; I_{n_b^{te}}]$, yielding:

$$\tilde{\mathbf{y}}_h^{cs} = S^{cs} \hat{\mathbf{y}}_h^{n_b^{cs}} \, , \tag{5}$$

$$\tilde{\mathbf{y}}^{te} = S^{te} \hat{\mathbf{y}}^{n_b^{te}} \, , \tag{6}$$

where $\hat{\mathbf{y}}_h^{n_b^{cs}}$ and $\hat{\mathbf{y}}^{n_b^{te}}$ denote the bottom-level base forecasts for the cross-sectional and temporal hierarchies, respectively. Considering only bottom-level forecasts has a number of benefits: (i) each forecast is coherent to any hierarchy by design, and (ii) we reduce the number of required forecasts from n to n_b, which can be a significant reduction (there is no need for a forecast for n_a aggregations in the hierarchy). We now construct a matrix of bottom-level base forecasts $\hat{\mathbf{Y}}^{n_b} \in \mathbb{R}^{n_b^{cs} \times n_b^{te}}$, in which the columns represent the forecasts of the bottom-level time series at a timestep h. This allows us to combine (5) and (6) as follows:

$$\tilde{\mathbf{Y}} = S^{cs} \hat{\mathbf{Y}}^{n_b} (S^{te})^\intercal \, , \tag{7}$$

in which $\tilde{\mathbf{Y}} \in \mathbb{R}^{n^{cs} \times n^{te}}$ represents the matrix of forecasts aggregated according to both cross-sectional and temporal hierarchies. Equivalently, we can aggregate our bottom-level ground truth values $\mathbf{Y}^{n_b} \in \mathbb{R}^{n_b^{cs} \times n_b^{te}}$:

$$\mathbf{Y} = S^{cs} \mathbf{Y}^{n_b} (S^{te})^\intercal \, . \tag{8}$$

Sparse hierarchical loss. To find the best forecasts for the hierarchical forecasting problem (7), we try to find a forecasting model using gradient-based optimization of the following loss function:

$$L = \sum \left[\frac{1}{2} \left(\left(\mathbf{Y} - \tilde{\mathbf{Y}} \right) \odot \left(\mathbf{Y} - \tilde{\mathbf{Y}} \right) \right) \oslash \left(d^{cs} d^{te} \right) \right] \, , \tag{9}$$

in which $\sum$ denotes the sum over all $n^{cs} \times n^{te}$ elements of the matrix contained in the summation, $\odot$ denotes element-wise multiplication, $\oslash$ denotes element-wise division, and the vectors d^{cs} and d^{te} read:

$$d^{cs} = l^{cs} S^{cs} \mathbf{1}^{cs} \, , \tag{10}$$

$$d^{te} = \left(l^{te} S^{te} \mathbf{1}^{te} \right)^\intercal \, , \tag{11}$$

where $S^{cs}\mathbf{1}^{cs}$ and $S^{te}\mathbf{1}^{te}$ denote the row-sum of S^{cs} and S^{te}, respectively, and l^{cs} and l^{te} denote the number of levels in hierarchies S^{cs} and S^{te}, respectively. We will detail the necessity of the element-wise division of Eq. (9) by the matrix $\left(d^{cs}d^{te}\right)$ later in this section. Note that Eq. (9) shares similarities with the *Weighted Root Mean Squared Error* from the M5 competition (Makridakis et al., 2022).

We can derive the gradient and the second-order derivative of (9) with respect to the bottom-level forecasts $\hat{\mathbf{Y}}^{n_b}$ (ref. Appendix A for the full derivation):

$$\frac{\partial L}{\partial \tilde{\mathbf{Y}}} = \left(\tilde{\mathbf{Y}} - \mathbf{Y}\right) \oslash \left(d^{cs}d^{te}\right) \ , \tag{12}$$

$$\frac{\partial L}{\partial \hat{\mathbf{Y}}^{n_b}} = (S^{cs})^{\intercal} \left(\frac{\partial L}{\partial \tilde{\mathbf{Y}}}\right) S^{te} \ , \tag{13}$$

$$\frac{\partial^2 L}{\partial \left(\hat{\mathbf{Y}}^{n_b}\right)^2} = (S^{cs})^{\intercal} \left(\mathbf{1} \oslash \left(d^{cs}d^{te}\right)\right) S^{te} \ . \tag{14}$$

Analysis. The best possible forecast is achieved when the loss (9) is minimized, or equivalently when the gradient (12) is zero:

$$\begin{aligned}\frac{\partial L}{\partial \tilde{\mathbf{Y}}} &= \left(\tilde{\mathbf{Y}} - \mathbf{Y}\right) \oslash \left(d^{cs}d^{te}\right) \ , \\ &= \left(S^{cs}\hat{\mathbf{Y}}^{n_b}(S^{te})^{\intercal} - S^{cs}\mathbf{Y}^{n_b}(S^{te})^{\intercal}\right) \oslash \left(d^{cs}d^{te}\right) \ , \\ &= \left(S^{cs}\left(\hat{\mathbf{Y}}^{n_b} - \mathbf{Y}^{n_b}\right)(S^{te})^{\intercal}\right) \oslash \left(d^{cs}d^{te}\right) \ ,\end{aligned}$$

which becomes zero when $\hat{\mathbf{Y}}^{n_b} = \mathbf{Y}^{n_b}$. Thus, the best forecast model is found when each bottom-level forecast equals the ground truth. This is equivalent to the standard (i.e., non-hierarchical) squared error loss often used in forecasting problems. We argue that our hierarchical loss gradient can be seen as a *smoothed* gradient compared to the standard squared error loss gradient (i.e., $\hat{\mathbf{Y}}^{n_b} - \mathbf{Y}^{n_b}$). For example, consider the canonical case where we have two bottom-level time series ($n_b^{cs} = 2$) consisting of two timesteps ($n_b^{te} = 2$). Furthermore, suppose we have a single cross-sectional aggregation (the sum of the two time series, thus $n_a^{cs} = 1$ and $n^{cs} = n_a^{cs} + n_b^{cs} = 3$), and a single temporal aggregation (the sum of the two timesteps, thus $n_a^{te} = 1$ and $n^{te} = n_a^{te} + n_b^{te} = 3$). Finally, there are two levels in our cross-sectional hierarchy and our temporal hierarchy, thus $l^{cs} = 2$ and $l^{te} = 2$. The standard squared error loss gradient for this problem is:

$$\begin{bmatrix} \frac{\partial L}{\partial \hat{\mathbf{y}}_{0,0}} & \frac{\partial L}{\partial \hat{\mathbf{y}}_{0,1}} \\ \frac{\partial L}{\partial \hat{\mathbf{y}}_{1,0}} & \frac{\partial L}{\partial \hat{\mathbf{y}}_{1,1}} \end{bmatrix} = \begin{bmatrix} e_{0,0} & e_{0,1} \\ e_{1,0} & e_{1,1} \end{bmatrix} , \tag{15}$$

in which $e_{i,j}$ denotes the bottom-level forecast error ($\hat{y}_{i,j} - y_{i,j}$) of the ith bottom-level timeseries and jth timestep, respectively. For our hierarchical loss, Eq. (7) reads:

$$\tilde{\mathbf{Y}} = \underbrace{\begin{bmatrix} 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}}_{S^{cs}} \underbrace{\begin{bmatrix} \hat{\mathbf{y}}_{0,0} & \hat{\mathbf{y}}_{0,1} \\ \hat{\mathbf{y}}_{1,0} & \hat{\mathbf{y}}_{1,1} \end{bmatrix}}_{\hat{\mathbf{Y}}^{n_b}} \underbrace{\begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}}_{(S^{te})^{\intercal}} , \tag{16}$$

and the gradient of the loss with respect to the bottom level time series Eq. (13) reads (ref. Appendix B for the full derivation):

$$\begin{bmatrix} \frac{\partial L}{\partial \hat{\mathbf{y}}_{0,0}} & \frac{\partial L}{\partial \hat{\mathbf{y}}_{0,1}} \\ \frac{\partial L}{\partial \hat{\mathbf{y}}_{1,0}} & \frac{\partial L}{\partial \hat{\mathbf{y}}_{1,1}} \end{bmatrix} = \begin{bmatrix} \frac{9}{16}e_{0,0} + \frac{3}{16}e_{1,0} + \frac{3}{16}e_{0,1} + \frac{1}{16}e_{1,1} & \frac{9}{16}e_{0,1} + \frac{3}{16}e_{0,0} + \frac{3}{16}e_{1,1} + \frac{1}{16}e_{1,0} \\ \frac{9}{16}e_{1,0} + \frac{3}{16}e_{0,0} + \frac{3}{16}e_{1,1} + \frac{1}{16}e_{0,1} & \frac{9}{16}e_{1,1} + \frac{3}{16}e_{1,0} + \frac{3}{16}e_{0,1} + \frac{1}{16}e_{0,0} \end{bmatrix} .$$

When we compare this result to the standard squared error loss gradient Eq. (15), we find that we *smooth* the bottom-level gradient by adding to it portions of the gradients of all cross-sectional and temporal aggregations the bottom-level series belongs to. This derivation also shows the motivation of adding the denominator matrix $\left(d^{cs}d^{te}\right)$ to the loss function (9). It is necessary to scale the aggregation gradients by the number of elements in the aggregation; otherwise, the magnitude of the gradient grows with the number of time series and the number of levels in the hierarchy, which we found to be undesirable when trying to facilitate stable learning. Thus, we add (portions of) the average gradient of the aggregations to the bottom-level gradient.

Sparsity. S^{cs} and S^{te} are highly sparse. For example, S^{cs} has at most $n_b^{cs}l^{cs}$ non-zero elements: the number of bottom-level time series multiplied by the number of aggregations in the hierarchy. Hence, the overall sparsity of S^{cs} is given by $1 - \frac{n_b^{cs}l^{cs}}{n^{cs}n_b^{cs}}$. For the M5 dataset (Makridakis et al., 2021), $n_b^{cs} = 3049$, $l^{cs} = 12$, $n^{cs} = 42{,}840$, corresponding to a sparsity of 99.97%. Next, the matrix of bottom-level ground truth values $\mathbf{Y}^{n_b}$ in (8) may be sparse too, for example, in the case of products that are not on sale for every timestep n_b^{te} in the dataset. All these sources of sparsity motivate using sparse linear algebra when computing Eqs. (9)–(14).

Implementation. We implement the hierarchical loss (9), the bottom-level gradient (12), (13) and second-order derivative (14) in Python using the sparse library from *SciPy* (Virtanen et al., 2020). Note that Eqs. (12)–(13) can be rearranged:

$$\frac{\partial L}{\partial \tilde{\mathbf{Y}}} = \left((S^{cs})^{\intercal} \oslash d^{cs}\right)\left(\tilde{\mathbf{Y}} - \mathbf{Y}\right)\left(S^{te} \oslash d^{te}\right) \ , \tag{17}$$

such that the parts containing S^{cs} and S^{te} can be precomputed as they do not depend on the forecast values $\tilde{\mathbf{Y}}$, avoiding a costly division operation inside a training iteration. Also note that the second-order derivative (14) does not depend on the forecast values $\tilde{\mathbf{Y}}$, so it can be precomputed as well. Our implementation, including the code to reproduce the experiments on public data from Section 5, is available on GitHub.[1]

5. Experiments

In this section, we empirically verify the usefulness of our sparse hierarchical loss. First, we evaluate forecasting accuracy using a set of experiments on the public M5 dataset (Makridakis et al., 2021). Then, we evaluate our sparse hierarchical loss in an offline experiment on a proprietary dataset from our e-commerce partner.

[1] https://github.com/elephaint/hfas

5.1. Public datasets

Task & dataset. Our task is forecasting product demand. We use the M5 dataset (Makridakis et al., 2021) for our offline, public dataset experiments. The M5 dataset contains product-level sales from Walmart for 3049 products across ten stores in the USA. Furthermore, the dataset contains 12 cross-sectional product aggregations (e.g. department, region), which allow us to test hierarchical forecasting performance. We preprocess the dataset, resulting in a set of features as described in Appendix C. We forecast 28 days into the future.

Baseline models. For our baseline forecasting model, we primarily use LightGBM (Ke et al., 2017), trained to predict one-day ahead. We subsequently recursively generate predictions for 28 days. Tree-based models dominated the M5 forecasting competition due to their strong performance and ease of use (Januschowski et al., 2022; Makridakis et al., 2022). Moreover, our e-commerce partner's primary product forecasting is a LightGBM-based model, so we expect results from offline experiments on public datasets to transfer to our proprietary setting when using the same base forecasting model. We compare the performance of our LightGBM models against traditional statistical methods ARIMA (Box & Pierce, 1970), ETS (Hyndman, Koehler, Ord, & Snyder, 2008), Theta (Assimakopoulos & Nikolopoulos, 2000), SeasonalNaive (Hyndman & Athanasopoulos, 2021), Naive (Hyndman & Athanasopoulos, 2021) and Croston (Croston, 1972). We note that deep learning-based approaches are becoming more prevalent in e-commerce (Kunz et al., 2023), especially with the rise of the Transformer-architecture in forecasting models (Li et al., 2019; Lim, Arık, Loeff, & Pfister, 2021). We considered this for future work and did not consider this for our study as (i) the cloud cost to operate these models is 10x higher for our e-commerce partner than a tree-based model, and (ii) none of the neural-network-based methods can scale to the size of our e-commerce partner, as explained in Section 3.2.

Experimental setup. To test our hierarchical sparse loss function against baseline forecasting systems, we consider the following scenarios:

1. `Bottom-up`. We train a single global model only on the bottom-level time series. Subsequently, the bottom-level forecasts are aggregated to obtain the aggregated (reconciled) forecasts.
2. `Separate aggregations`. We train separate models for every aggregation in the hierarchy, resulting in 12 models for the entire M5 dataset.
3. `Global`. We train a single global model on all time series in the dataset, including all the aggregations.

For the first scenario in our experiments (`Bottom-up`), we vary both the *objective* (i.e. loss function that LightGBM optimizes) and the *evaluation metric* (i.e. the loss function that governs early-stopping during hyperparameter optimization). For the *objective*, we consider the *squared error loss* (SL), the *Tweedie loss* (TL) and our *sparse hierarchical loss* (HL). The Tweedie loss is a loss function that assumes that the time series follows a distribution somewhere between a Poisson and Gamma distribution, which is useful in zero-inflated settings such as retail demand forecasting. It is a loss function favoured by contestants in the M5 forecasting competition (Januschowski et al., 2022), and it is the loss also used in the primary forecasting system of our e-commerce partner.

For the latter two scenarios, we will obtain non-coherent forecasts. Thus, these methods require a reconciliation post-processing step to reconcile the forecasts to the hierarchy. We employ the following cross-sectional reconciliation methods:

- **Base**. No reconciliation is performed.
- **OLS**. Ordinary Least Squares (OLS) (Hyndman et al., 2011), where W in Eq. (2) is the identity matrix.
- **WLS-struct** and **WLS-var**. Weighted Least Squares (WLS) (Wickramasuriya et al., 2019), where W in Eq. (2) is a diagonal matrix containing respectively the sum of the rows of S (*WLS-struct*) or the in-sample forecast errors (*WLS-var*).
- **MinT-shrink**. *Trace Minimization* (Wickramasuriya et al., 2019), where W in Eq. (2) is the shrunk covariance matrix of in-sample forecast errors. We also experimented with using the non-shrunk covariance matrix of the in-sample forecast errors (*MinT-cov*), but this produced erroneous/high variance results, which we attribute to precisely the motivation to shrink the covariance matrix in *MinT-shrink*: to reduce the variance when the amount of time series considered becomes very large.
- **ERM**. The *Empirical Risk Minimization* (ERM) method (Ben Taieb & Koo, 2019). Due to computational issues explained in Section 3.2, we could not apply the regularized ERM variants to our experiments, but only the unregularized variant.

We optimize the hyperparameters of each of the LightGBM models by Bayesian hyperparameter optimization using Optuna (Akiba, Sano, Yanase, Ohta, & Koyama, 2019). The settings for the hyperparameter optimization can be found in Appendix D. Each model is trained for ten different random seeds, and our results are based on the mean and standard deviation of those ten rollouts. For the traditional statistical methods, we use Nixtla's StatsForecast (Garza, Mergenthaler Canseco, Challú, & Olivares, 2022), which includes automatic optimization of the hyperparameters of the statistical methods.

Evaluation. We evaluate our results for every aggregation in the hierarchy using the Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) (Hyndman & Athanasopoulos, 2021). In the results section, we present the RMSE/MAE relative to the `Bottom-up` scenario using the squared-loss objective with the squared-loss metric. For the results' absolute values and standard deviation, see Appendix E.

Results – LightGBM as baseline model. For our first experiment, we only consider cross-sectional hierarchies (i.e., $S^{te} = I_{n_b^{te}}$). We present our results on relative RMSE using LightGBM as a baseline model in Table 4 and conclude the following:

Table 1
Forecasting results for all time series (incl. aggregations) on the M5 dataset, using different baseline models. We show absolute and relative RMSE and MAE. Lower is better, and bold indicates the best-performing method.

Model	Reconciliation	RMSE		MAE	
		Abs.	Rel.	Abs.	Rel.
LightGBM (SL/SL)	None	*22.39*	*1.00*	*2.20*	*1.00*
LightGBM (HL/HL)	None	**19.54**	**0.87**	**2.10**	**0.95**
LightGBM (HL/SL)	None	**19.59**	**0.88**	**2.10**	**0.95**
ARIMA	MinT-shrink	39.88	2.43	1.78	1.10
ETS	MinT-shrink	36.48	2.35	1.63	1.07
Theta	MinT-shrink	36.66	2.39	1.64	1.08
Croston	None	39.40	2.76	1.76	1.25
Naive	None	74.91	3.95	3.35	1.80
Seasonal Naive	None	39.40	2.76	1.76	1.25

Table 2
Forecasting results for all time series (incl. aggregations) on the M5 dataset, ablating for using cross-sectional and temporal hierarchies. We show absolute and relative RMSE and MAE, with the standard deviation in brackets. Lower is better, and bold indicates the best-performing method. Note that the hierarchical loss equals the standard squared error loss when not using cross-sectional or temporal aggregations.

Hierarchies		RMSE		MAE	
Cross-sectional	Temporal	Abs.	Rel.	Abs.	Rel.
No	No	*22.39* (0.16)	*1.00*	*2.20* (0.01)	*1.00*
Yes	No	**19.54** (0.38)	**0.87**	**2.10** (0.01)	**0.95**
Yes	Yes	29.81 (1.52)	1.33	2.47 (0.04)	1.12
No	Yes	26.65 (0.32)	1.19	2.36 (0.01)	1.07
Random	No	23.54 (0.73)	1.05	2.19 (0.02)	1.00

- The best method is the `Bottom-up`-scenario combined with our sparse hierarchical loss as objective, outperforming the baseline by 0%–20% across aggregations. This holds for both settings in which we use our sparse hierarchical loss.
- Even when we only use our sparse hierarchical loss as an evaluation metric during training whilst optimizing the standard squared loss (the SL/HL scenario), we already see a small improvement of ±5% across aggregations.
- Even though the Tweedie loss improves over the baseline loss, our sparse hierarchical loss function still outperforms it by ±5% across aggregations.
- From the reconciliation methods, *MinT-shrink* and *WLS-var* perform best in the `Separate aggregations`-scenario, although the performance delta across aggregations is still ±5%–30% as compared to the best (our) method.

We present our results for relative MAE in Table 5. Overall, our sparse hierarchical loss still performs best by ±5% compared to other loss functions and scenarios. However, the results are more nuanced: we find that *MinT-shrink* in the `Separate aggregations`-scenario performs strongly as well. In addition, we also find that the Tweedie loss (TL) performs relatively well. This finding corroborates the usefulness of the TL in intermittent demand settings, such as retail, where zero demand is often observed.

Table 3
Forecasting results for all time series (incl. aggregations) on the M5 dataset, ablating for using cross-sectional and temporal hierarchies. We show relative RMSE for several forecasting day buckets of the forecast. Lower is better, and bold indicates the best-performing method.

Hierarchies		Forecast day			
Cross-sectional	Temporal	1–7	8–14	15–21	22–28
No	No	*1.00*	*1.00*	*1.00*	*1.00*
Yes	No	**0.98**	0.99	**0.81**	**0.80**
Yes	Yes	1.75	1.50	0.96	1.66
No	Yes	1.37	1.24	0.97	1.41
Random	No	1.31	**0.94**	1.07	0.87

Next, we compare our findings against the forecasting results when employing different baseline models in Table 1. We only show the metrics for all time series combined (incl. aggregations) for brevity. In addition, we only show a single reconciliation method for the other baseline models, as we found little difference in results when employing different reconciliation methods. We then find that in terms of RMSE, our sparse hierarchical loss outperforms the other baseline models by at least 50% and in terms of MAE by at least 10%. This verifies that on this dataset and with this type of problem, using a more complex model such as LightGBM greatly improves forecasting performance, as was also shown in the M5 forecasting competition (Makridakis et al., 2022).

Analysis: impact of hierarchy. We investigate the impact of the choice of hierarchy.

Temporal hierarchies. As noted before, we only used cross-sectional aggregations in our first experiments. We now include temporal aggregations by aggregating our bottom-level time series across years, weeks and months. We ablate for every setting and show the results in Table 2. Interestingly, we find that using temporal hierarchies jointly with cross-sectional hierarchies reduces forecasting performance by ±35% (RMSE) and ±17% (MAE). This setting is even worse than only using temporal hierarchies, which performs worse than using only cross-sectional hierarchies by ±26% (RMSE) and ±12% (MAE). We further analyze these results by studying the RMSE across the forecast days in Table 3. As noted before, we forecast 28 days ahead, and each forecast is created by recursively applying the one-step ahead LightGBM model. We find that as we forecast further into the future, the setting with only using cross-sectional aggregations starts to perform better by up to 20% compared to the baseline where we do not use any aggregations. Again, the setting where we employ temporal hierarchies shows relatively bad performance across all forecast day buckets.

Random hierarchies. In hierarchical forecasting problems, the aggregation matrices S^{cs} and S^{te} are commonly fixed a priori and considered constant during training and prediction. We can modify these matrices as we perform the reconciliation end-to-end during training. This allows us to understand the robustness of our solution to possible misspecification of the hierarchy and, more generally, to what extent the choice of the hierarchy affects forecasting performance. We experiment by randomly sampling an S^{cs}-matrix before we start the LightGBM training process. We sample uniformly at random (i) a number of

Table 4
Forecasting results for all stores on the M5 dataset, using LightGBM as a baseline model. We report relative RMSE compared to the baseline (shown in italics). Lower is better, and bold indicates the best method for the aggregation, considering the best method's standard deviation across the ten seeds. For the results' absolute values and standard deviation, see Appendix E. The Bottom-up scenario using the HL loss commonly outperforms all other scenarios.

Scenario/Objective	Metric	Reconciliation	Product	Department	Category	Store			Product		State			Total	All series
						Department	Category	Total	Store	State	Department	Category	Total		
`Bottom-up`															
SL	*SL*	*None*	***1.00***	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*
SL	HL	None	**1.00**	0.97	0.95	0.98	0.97	0.98	0.99	1.00	0.98	0.96	0.99	1.00	0.98
HL	HL	None	**1.00**	**0.88**	**0.80**	**0.93**	**0.89**	**0.93**	**0.97**	**0.99**	**0.89**	**0.84**	**0.88**	**0.87**	**0.87**
HL	SL	None	**1.00**	**0.88**	**0.81**	**0.94**	**0.90**	**0.94**	**0.96**	**0.98**	**0.89**	**0.84**	**0.88**	**0.87**	**0.88**
TL	HL	None	**1.00**	0.95	0.96	0.99	1.00	1.00	0.99	1.00	0.97	0.98	0.98	0.96	0.97
TL	SL	None	**1.00**	0.94	0.93	1.00	1.00	1.00	0.99	1.00	0.97	0.98	0.98	0.93	0.96
TL	TL	None	1.17	2.72	2.81	1.76	1.83	1.73	1.52	1.33	2.15	2.18	2.08	2.71	2.38
`Sep. agg.`															
SL	SL	Base	**1.00**	1.44	1.29	1.19	1.14	1.14	1.01	0.99	1.23	1.34	1.27	1.60	1.35
SL	SL	OLS	**1.00**	1.39	1.41	1.10	1.06	1.07	1.00	1.00	1.20	1.19	1.23	1.50	1.30
SL	SL	WLS-struct	**1.00**	1.26	1.37	1.03	1.05	1.02	0.99	0.99	1.11	1.16	1.16	1.39	1.23
SL	SL	WLS-var	**1.00**	1.12	1.23	0.99	1.02	0.99	0.99	0.99	1.03	1.09	1.07	1.22	1.12
SL	SL	MinT-shrink	**1.00**	1.15	1.27	0.97	0.99	0.97	1.00	1.00	1.03	1.09	1.09	1.30	1.15
SL	SL	ERM	1.22	1.25	1.29	1.07	1.03	1.07	1.17	1.22	1.17	1.14	1.22	1.49	1.26
`Global`															
SL	SL	Base	1.02	1.33	1.45	1.09	1.10	1.10	1.03	1.03	1.25	1.27	1.81	1.57	1.46
SL	SL	OLS	1.01	1.32	1.39	1.07	1.09	1.16	1.02	1.02	1.20	1.25	1.38	1.49	1.34
SL	SL	WLS-struct	1.01	1.38	1.54	1.08	1.13	1.11	1.03	1.02	1.19	1.28	1.27	1.55	1.36
SL	SL	WLS-var	1.01	1.51	1.70	1.18	1.27	1.22	1.03	1.02	1.31	1.43	1.38	1.66	1.48
SL	SL	MinT-shrink	1.03	1.26	1.41	1.05	1.10	1.15	1.06	1.05	1.11	1.17	1.24	1.54	1.30
SL	SL	ERM	1.21	1.59	1.69	1.26	1.28	1.34	1.20	1.23	1.45	1.49	1.61	1.80	1.59

Table 5
Forecasting results for all stores on the M5 dataset, using LightGBM as a baseline model. We report relative MAE compared to the baseline (shown in italics). Lower is better, and bold indicates the best method for the aggregation, considering the best method's standard deviation across the ten seeds. For the results' absolute values and standard deviation, see Appendix E. The `Bottom-up` scenario using the HL loss commonly outperforms all other scenarios.

Scenario/Objective	Metric	Reconciliation	Product	Department	Category	Store			Product		State			Total	All series
						Department	Category	Total	Store	State	Department	Category	Total		
`Bottom-up`															
SL	*SL*	*None*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*	*1.00*
SL	HL	None	1.00	0.98	0.96	0.98	0.98	0.99	1.00	1.00	0.97	0.97	1.00	1.02	0.99
HL	HL	None	0.99	**0.81**	**0.78**	**0.90**	**0.89**	0.94	0.98	0.99	**0.85**	**0.83**	**0.89**	**0.88**	**0.95**
HL	SL	None	0.99	**0.81**	**0.77**	**0.90**	**0.90**	0.94	0.98	0.99	**0.85**	**0.83**	**0.89**	**0.87**	**0.95**
TL	HL	None	0.98	**0.83**	0.82	0.93	0.94	0.96	0.97	0.98	0.88	0.89	0.92	**0.88**	**0.95**
TL	SL	None	0.99	0.85	0.84	0.95	0.95	0.98	0.98	0.99	0.90	0.91	0.96	**0.90**	**0.96**
TL	TL	None	1.02	2.06	2.39	1.47	1.61	1.65	1.14	1.07	1.69	1.88	1.97	2.86	1.29
`Sep. agg.`															
SL	SL	Base	1.00	1.11	1.06	1.02	1.10	1.08	0.96	0.97	1.03	1.15	1.20	1.55	1.02
SL	SL	OLS	**0.96**	1.08	1.16	1.01	0.99	1.00	0.96	0.97	1.00	1.01	1.12	1.49	0.99
SL	SL	WLS-struct	0.97	0.99	1.13	0.92	0.95	0.95	**0.96**	**0.97**	0.93	0.98	1.04	1.42	0.98
SL	SL	WLS-var	0.98	0.91	1.00	0.91	0.92	**0.92**	0.96	0.97	0.90	0.93	0.95	1.16	**0.96**
SL	SL	MinT-shrink	0.97	0.93	1.05	**0.90**	**0.90**	**0.91**	0.96	0.97	0.89	0.92	0.98	1.30	**0.96**
SL	SL	ERM	1.18	1.17	1.21	1.09	1.06	1.07	1.19	1.22	1.11	1.12	1.19	1.57	1.18
`Global`															
SL	SL	Base	1.04	1.04	1.17	1.00	1.02	1.05	0.99	0.99	1.04	1.09	1.62	1.58	1.05
SL	SL	OLS	0.98	1.09	1.17	1.05	1.07	1.13	0.99	1.00	1.09	1.13	1.27	1.50	1.03
SL	SL	WLS-struct	0.99	1.09	1.26	0.98	1.01	1.04	1.00	0.99	1.00	1.07	1.15	1.57	1.02
SL	SL	WLS-var	0.99	1.24	1.40	1.09	1.13	1.13	1.01	1.00	1.14	1.21	1.25	1.69	1.06
SL	SL	MinT-shrink	0.99	1.13	1.24	1.06	1.08	1.15	1.02	1.01	1.05	1.07	1.15	1.60	1.04
SL	SL	ERM	1.17	1.49	1.62	1.30	1.34	1.37	1.19	1.20	1.39	1.49	1.58	1.96	1.27

levels for the cross-sectional hierarchy and (ii) a number of maximum categories for the level and construct a random S^{cs}-matrix to be used in the gradient (13) and second-order derivative (14). We validate and test on the 'true' S^{cs}-matrix. We present the results in Tables 2 and 3, under 'Random'. In Table 2, we find that on RMSE, forecasting performance deteriorates by about 5% as compared to the baseline using no hierarchies and by about 20% as compared to the best setting in which we use the correct cross-sectional hierarchies during training. In Table 3, we find that 'Random' performs poorly on the first forecast period, whereas it performs strongly on the second and final week of the forecast period. Thus, misspecification of the hierarchy can severely deteriorate forecasting performance. Still, the relatively strong performance at some forecast intervals in Table 3 could also indicate that a better hierarchy randomization strategy might lead to improved forecast results. We leave this for future work.

Analysis: time complexity. We investigate the computational time complexity required to perform training and prediction for each scenario and present the results in Table 6. The training and prediction time complexity is indicated by how the training time and prediction time

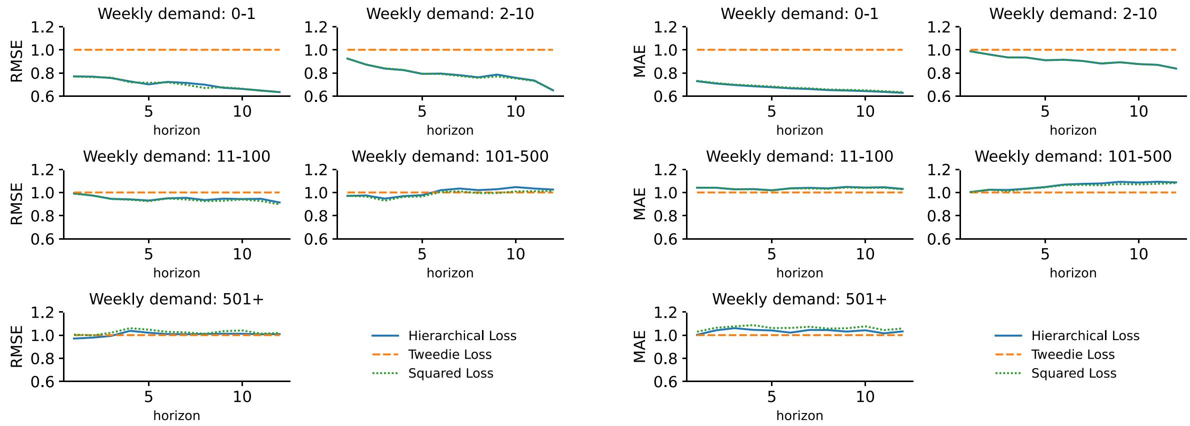
Table 6

Computational time complexity and observed timings in seconds for all scenarios. The complexity is indicated by how respectively the training time and prediction time scales with respect to the default LightGBM training/prediction time L, where n_b^{te} denotes the number of timesteps per time series, n_b^{cs} denotes the number of bottom-level time series in the hierarchy, $n_b^{cs_l}$ the number of time series in each level in the hierarchy and l^{cs} the number of levels in the cross-sectional hierarchy, n^{cs} (n^{te}) the total number of cross-sectional (temporal) aggregations, and $n_a = n - n_b$. The `Bottom-up` scenario using the HL loss is computationally more efficient than the `Separate aggregations` (both training and prediction) and `Global` (prediction) scenarios.

Scenario/Obj.	Metric	Reconciliation	Complexity		Training time (s)		Prediction time (s)	
			Training	Prediction	1 store	All stores	1 store	All stores
`Bottom-up`								
SL	SL	None	$O\left(L\left(n_b^{te}n_b^{cs}\right)\right)$	$O\left(L\left(n_b^{te}n_b^{cs}\right)+n_b^{te}\left(n_b^{cs}\right)^3\right)$	8	173	1.1	11
HL (dense)	HL (dense)	None	$O\left(L\left(n_b^{te}n_b^{cs}+n_b^{te}\left(n_b^{cs}\right)^3\right)\right)$	$O\left(L\left(n^{te}n^{cs}\right)+n_b^{te}\left(n_b^{cs}\right)^3\right)$	14	1,185	1.1	10
HL (sparse)	HL (sparse)	None	$O\left(L\left(n_b^{te}n_b^{cs}+n_b^{te}\left(n_b^{cs}\right)^2 l^{cs}\right)\right)$	$O\left(L\left(n^{te}n^{cs}\right)+n_b^{te}\left(n_b^{cs}\right)^2 l^{cs}\right)$	12	318	0.1	11
HL+ (sparse)	HL+ (sparse)	None	$O\left(L\left(n_b^{te}n_b^{cs}+n_b^{te}\left(n_b^{cs}\right)^2 l^{cs}n_b^{cs}\left(n_b^{te}\right)^2 l^{te}\right)\right)$	$O\left(L\left(n^{te}n^{cs}\right)+n_b^{te}\left(n_b^{cs}\right)^2 l^{cs}n_b^{cs}\left(n_b^{te}\right)^2 l^{te}\right)$		723		15
`Sep. agg.`								
SL	SL	Base	$O\left(l^{cs}\cdot L\left(n_b^{cs_l}n_b^{te_l}\right)\right)$	$O\left(l^{cs}\cdot L\left(n_b^{cs_l}n_b^{te_l}\right)\right)$	11	36,018	4.4	103
SL	SL	OLS		$O\left(l^{cs}\cdot T\left(n_b^{cs_l}n_b^{te_l}\right)+\left(n_a^{cs}\right)^3\right)$			4.5	149
SL	SL	WLS-struct		$O\left(l^{cs}\cdot T\left(n_b^{cs_l}n_b^{te_l}\right)+\left(n_a^{cs}\right)^3\right)$			4.5	151
SL	SL	WLS-var		$O\left(l^{cs}\cdot T\left(n_b^{cs_l}n_b^{te_l}\right)+\left(n_a^{cs}\right)^3\right)$			4.5	151
SL	SL	MinT-shrink		$O\left(l^{cs}\cdot T\left(n_b^{cs_l}n_b^{te_l}\right)+\left(n^{cs}\right)^3\right)$			5.8	305
SL	SL	ERM		$O\left(l^{cs}\cdot T\left(n_b^{cs_l}n_b^{te_l}\right)+\left(n^{cs}\right)^3\right)$			6.0	239
`Global`								
SL	SL	Base	$O\left(L\left(n^{te}n^{cs}\right)\right)$	$O\left(T\left(n^{te}n^{cs}\right)\right)$	4	173	2.4	71
SL	SL	OLS		$O\left(T\left(n^{te}n^{cs}\right)+\left(n_a^{cs}\right)^3\right)$			2.5	118
SL	SL	WLS-struct		$O\left(T\left(n^{te}n^{cs}\right)+\left(n_a^{cs}\right)^3\right)$			2.5	120
SL	SL	WLS-var		$O\left(T\left(n^{te}n^{cs}\right)+\left(n_a^{cs}\right)^3\right)$			2.5	120
SL	SL	MinT-shrink		$O\left(T\left(n^{te}n^{cs}\right)+\left(n^{cs}\right)^3\right)$			4.2	274
SL	SL	ERM		$O\left(T\left(n^{te}n^{cs}\right)+\left(n^{cs}\right)^3\right)$			4.0	207

scales with respect to the default LightGBM training and prediction time complexity. We first investigate the case where we only consider cross-sectional hierarchies. This case is indicated by 'HL' in Table 6. First, we note that adding our hierarchical loss objective adds a component to the time complexity that scales with $\left(n_b^{cs}\right)^3$, as we need to compute (12). However, our sparse implementation of the hierarchical loss reduces this component from $\left(n_b^{cs}\right)^3$ to $\left(n_b^{cs}\right)^2 l^{cs}$, effectively reducing the scaling from cubic to quadratic in the number of bottom-level time series, as l^{cs} is generally small. In the reconciliation scenarios, we always need to compute a matrix inversion to solve Eq. (2) that scales cubically with the number of cross-sectional aggregations n_a^{cs} or with the total number of time series n^{cs}. The first is not problematic as generally $n_a^{cs} \ll n_b^{cs}$ in large-scale settings, but methods with this time complexity consequently trade in performance, as we observed in Table 4. We recorded the training and prediction time for each scenario to verify the differences in asymptotic time complexity empirically. We show timings for training and prediction for a single store of the M5 dataset (4M training samples) and for the entire M5 dataset (52M training samples) to indicate scaling when the problem size increases by an order of magnitude. First, we note that using our sparse implementation of the HL reduces training time by a factor of 3× when training for all stores. Second, our sparse HL has a prediction time similar to the baseline (SL). Next, we find that the training time of our sparse hierarchical loss is two orders of magnitude faster than reconciliation methods in the `Separate aggregations`-scenario. This is mainly due to the many individual models that need to be trained in this scenario, and thus, it shows a clear benefit of having just a single model. We observe an order of magnitude difference in prediction time when comparing the sparse hierarchical loss to the `Separate aggregations`-scenario when predicting all stores. Again, this shows a clear benefit of having just a single model for this forecasting task. For the `Global`-scenario, we see that reconciliation methods require a shorter training time when training for all stores (about twice less); however, that scenario also did not give strong forecasting performance as we established in Table 4. Also, the prediction time using our sparse HL is an order of magnitude lower. As ML costs in production systems mainly consist of prediction costs, having a lower prediction time is beneficial.[2] Finally, we show the time complexity of using cross-sectional and temporal hierarchies jointly, as indicated by '*HL+*' in Table 6. Adding temporal hierarchies adds another matrix multiplication that scales with the number of timesteps to the complexity. In our experiments, we find that adding temporal hierarchies results in a twice higher training time when training for all stores and a 50% higher prediction time when predicting for all stores. We view it as potential future work to efficiently investigate how to perform this end-to-end learning of cross-sectional and temporal hierarchies.

2 For example, Google designed its first TPU for inference: https://techcrunch.com/2017/05/17/google-announces-second-generation-of-tensor-processing-unit-chips.



(a) RMSE (b) MAE

Fig. 1. Forecasting results for the primary product forecasting model at our e-commerce partner bol.com. We show RMSE (a, left) and MAE (b, right) by weekly demand bucket relative to the Tweedie loss baseline for each forecasting horizon (week). The Hierarchical loss outperforms the Tweedie loss on RMSE and MAE on smaller weekly demand buckets.

Table 7
Comparison of dataset characteristics between the M5 dataset and the proprietary dataset. We split the weekly demand into weekly demand buckets and showed the percentage of samples and percentage of demand for each bucket.

Weekly	% samples		% demand	
demand	M5	Proprietary	M5	Proprietary
0	40.73%	34.15%	0%	0%
1	7.92%	19.23%	1.02%	3.99%
2–10	33.28%	37.31%	20.93%	31.6%
11–100	17.21%	8.89%	57.38%	46.01%
101–500	0.84%	0.39%	18.59%	14.27%
501+	0.02%	0.02%	2.08%	4.13%
Total	100.00%	100.00%	100.00%	100.00%

To conclude, we showed that our sparse HL incurs some additional training overhead but no additional prediction overhead compared to the base case SL. In contrast, it does not require the additional reconciliation step that reconciliation methods require.

5.2. Proprietary datasets

Our e-commerce partner bol.com uses a LightGBM-based forecasting model as the primary product forecasting model. The model is used to forecast weekly product demand for 12 weeks. Every day, 12 separate models are trained, each tasked to forecast demand for a single week for every product. The model forecasts most of the products on sale at any moment, which is approximately 5 million unique items. We investigate using our sparse hierarchical loss function as a drop-in replacement for the existing Tweedie loss used within the company.

Dataset. The offline dataset consists of 36M training samples from January 2017 to the end of June 2021. We test on 55M samples from July 2021 to January 2022. We show statistics of the proprietary dataset compared to the M5 dataset in Table 7, in which we split the weekly demand of both datasets according to weekly demand buckets used by our e-commerce partner. In Table 7, we find that the M5 dataset and our proprietary dataset share demand characteristics in terms of sparsity (i.e., zero demand), which is 41% for M5 and 34% for our proprietary dataset, respectively. We generally find that the two datasets share sufficient weekly demand density characteristics to warrant using our sparse HL on our proprietary dataset. The proprietary dataset contains 19 proprietary features, which are similar to those used in the M5 dataset (ref. Table C.8), and consist of (i) product categorical features, (ii) weekly demand (target) lagged features, and (iii) seasonality features.

Experimental setup. The baseline model for every weekly forecast model is a LightGBM model with a *Tweedie loss* (TL). We replaced the TL with our HL and investigated forecasting performance on the test set. We apply log-scaling to the target values. For the HL, we use the proprietary aggregations *product_group* and *seasonality_group*, each containing respectively ±70 and ±6000 unique values. We have $n_b^{cs} = \pm 5\text{M}$ bottom-level time series and $n_a^{cs} = \pm 6070$ aggregated time series across $l^{cs} = 4$ levels: *product* (bottom-level), *product_group*, *seasonality_group* and *total*.

Results. On average, our sparse HL outperforms the existing TL model by about 1%–2% on RMSE and ±10% on MAE. We further investigate the performance by investigating how the RMSE and MAE vary across the 12 forecasting horizons and weekly demand buckets, and present the results in Fig. 1. Our sparse HL performs best on both RMSE and MAE on the lower weekly demand buckets (up to 100 products sold per week), outperforming the TL averaged over all the forecasting horizons. The TL is better for higher weekly demand buckets, commonly outperforming the HL and SL by up to 5%. Next, we investigate forecasting performance across the cross-sectional hierarchies that we defined. We show the results in Fig. 2. We find that for most forecasting horizons, the HL and SL outperform the TL, with an average outperformance of the HL over the TL of ±10% at the product level, ±5% at the product group level and ±4%–7% at seasonality group level. Hence, we can confirm some of the results we found in the M5 experiment, although the baseline SL performed quite strongly in this experiment. We believe this is due to

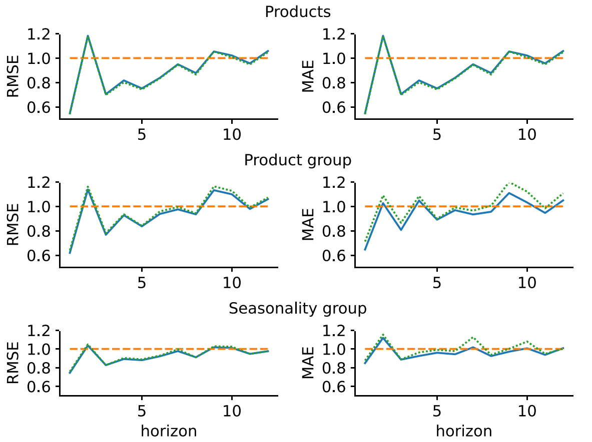


Fig. 2. Forecasting results for the primary product forecasting model at our e-commerce partner bol.com. We show RMSE (left column of figures) and MAE (right column of figures) by aggregation level relative to the Tweedie loss baseline for each forecasting horizon (week). The Hierarchical loss commonly outperforms the Tweedie loss on every aggregation level.

the M5 experiment having much more hierarchical levels (12 compared to the four we used for our proprietary dataset experiment) since the HL is equal to the SL in the case of no hierarchies. With fewer hierarchies, the HL thus becomes closer to the SL. Hence, our HL is most useful in settings with many time series and hierarchies.

To conclude, this experiment demonstrates the usefulness of our sparse HL applied to a production forecasting system at a major e-commerce platform.

6. Conclusion

We introduced a sparse hierarchical loss function to perform hierarchical forecasting in large-scale settings. We demonstrated that we could outperform existing hierarchical forecasting methods both in terms of performance as measured by RMSE and MAE by up to 10% as well as in terms of computational time required to perform the end-to-end hierarchical forecasting in large-scale settings, reducing prediction time as compared to the best hierarchical forecasting reconciliation method by order of magnitude. We empirically verified our sparse hierarchical loss in an offline test for bol.com, where we confirmed the results from our offline test on the public M5 dataset.

In addition to our main contributions, one of our main learnings has been that we could not find a benefit of having multiple models for separate aggregations in the hierarchy, as the bottom-up scenario we employed consistently outperformed other scenarios. Secondly, we did not find a benefit in training a model whilst jointly adhering to cross-sectional and temporal hierarchies.

Limitations of our work are that we did not consider the probabilistic forecasting setting, where reconciled forecasts are required across an entire forecast distribution.

For future work, we aim to extend our work to the setting of probabilistic forecasting by combining our sparse hierarchical loss with existing probabilistic forecasting frameworks from, e.g., Hasson, Wang, Januschowski, and Gasthaus (2021), Sprangers, Schelter, and de Rijke (2021), Stankeviciute, Alaa, and van der Schaar (2021). In addition, we seek to investigate solutions for efficiently combining cross-sectional and temporal hierarchies further. Finally, we aim to understand further the influence of hierarchy misspecification in the hierarchical forecasting setting.

CRediT authorship contribution statement

Olivier Sprangers: Conceptualization, Methodology, Experiments, Software, Writing – original draft, Writing – review & editing. **Wander Wadman:** Experiments support, Writing – review & editing. **Sebastian Schelter:** Experiments support, Writing – review & editing. **Maarten de Rijke:** Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Olivier Sprangers reports financial support was provided by Ahold Delhaize. Sebastan Schelter reports financial support was provided by Ahold Delhaize. Olivier Sprangers reports a relatonship with bol.com that includes: non-financial support. Sebastan Schelter reports a relatonship with Ahold Delhaize that includes: non-financial support. Wander Wadman reports a relatonship with bol.com that includes: employment. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to infuence the work reported in this paper.

Acknowledgments

We thank the editor and reviewers for their constructive feedback that helped us to improve our work.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.ijforecast.2024.02.006.

References

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 2623–2631). New York, NY, USA: Association for Computing Machinery, http://dx.doi.org/10.1145/3292500.3330701.

Assimakopoulos, V., & Nikolopoulos, K. (2000). The theta model: A decomposition approach to forecasting. *International Journal of Forecasting*, *16*(4), 521–530. http://dx.doi.org/10.1016/S0169-2070(00)00066-2.

Athanasopoulos, G., Hyndman, R. J., Kourentzes, N., & Panagiotelis, A. (2024). Forecast reconciliation: A review. *International Journal of Forecasting*, *40*(2), 430–456. http://dx.doi.org/10.1016/j.ijforecast.2023.10.010.

Athanasopoulos, G., Hyndman, R. J., Kourentzes, N., & Petropoulos, F. (2017). Forecasting with temporal hierarchies. *European Journal of Operational Research*, *262*(1), 60–74. http://dx.doi.org/10.1016/j.ejor.2017.02.046.

Ben Taieb, S. (2017). Sparse and smooth adjustments for coherent forecasts in temporal aggregation of time series. In *Proceedings of the time series workshop* (pp. 16–26). PMLR.

Ben Taieb, S., & Koo, B. (2019). Regularized regression for hierarchical forecasting without unbiasedness conditions. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining* (pp. 1337–1347). Anchorage AK USA: ACM, http://dx.doi.org/10.1145/3292500.3330976.

Ben Taieb, S., Taylor, J. W., & Hyndman, R. J. (2017). Coherent probabilistic forecasts for hierarchical time series. In *International conference on machine learning* (pp. 3348–3357).

Benidis, K., Rangapuram, S. S., Flunkert, V., Wang, Y., Maddix, D., Turkmen, C., et al. (2023). Deep learning for time series forecasting: tutorial and literature survey. *ACM Computing Surveys*, *55*, 1–36. http://dx.doi.org/10.1145/3533382, arXiv:2004.10240.

Böse, J.-H., Flunkert, V., Gasthaus, J., Januschowski, T., Lange, D., Salinas, D., et al. (2017). Probabilistic demand forecasting at scale. *Proceedings of the VLDB Endowment*, *10*(12), 1694–1705. http://dx.doi.org/10.14778/3137765.3137775.

Box, G. E. P., & Pierce, D. A. (1970). Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *Journal of the American Statistical Association*, *65*(332), 1509–1526. http://dx.doi.org/10.2307/2284333, arXiv:2284333.

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 785–794). San Francisco California USA: ACM, http://dx.doi.org/10.1145/2939672.2939785.

Croston, J. D. (1972). Forecasting and stock control for intermittent demands. *Operational Research Quarterly (1970-1977)*, *23*(3), 289–303. http://dx.doi.org/10.2307/3007885, arXiv:3007885.

Garza, F., Mergenthaler Canseco, M., Challú, C., & Olivares, K. G. (2022). StatsForecast: Lightning fast forecasting with statistical and econometric models. In *pyCon*. Salt Lake City, USA.

Girolimetto, D., & Di Fonzo, T. (2023). Point and probabilistic forecast reconciliation for general linearly constrained multiple time series. *Statistical Methods & Applications*, http://dx.doi.org/10.1007/s10260-023-00738-6

Han, X., Dasgupta, S., & Ghosh, J. (2021). Simultaneously reconciled quantile forecasting of hierarchically related time series. In *Proceedings of the 24th international conference on artificial intelligence and statistics* (pp. 190–198). PMLR.

Hasson, H., Wang, B., Januschowski, T., & Gasthaus, J. (2021). Probabilistic forecasting: A level-set approach. In *Advances in neural information processing systems, vol. 34* (pp. 6404–6416). Curran Associates, Inc.

Hyndman, R. J., Ahmed, R. A., Athanasopoulos, G., & Shang, H. L. (2011). Optimal combination forecasts for hierarchical time series. *Computational Statistics & Data Analysis*, *55*(9), 2579–2589. http://dx.doi.org/10.1016/j.csda.2011.03.006.

Hyndman, R. J., & Athanasopoulos, G. (2021). *Forecasting: Principles and practice* (3rd ed.). Melbourne, Australia: OTexts.

Hyndman, R., Koehler, A. B., Ord, J. K., & Snyder, R. D. (2008). *Forecasting with exponential smoothing: The state space approach*. Springer Science & Business Media.

Hyndman, R. J., Lee, A. J., & Wang, E. (2016). Fast computation of reconciled forecasts for hierarchical and grouped time series. *Computational Statistics & Data Analysis*, *97*, 16–32. http://dx.doi.org/10.1016/j.csda.2015.11.007.

Januschowski, T., Wang, Y., Torkkola, K., Erkkilä, T., Hasson, H., & Gasthaus, J. (2022). Forecasting with trees. *International Journal of Forecasting*, *38*(4), 1473–1481. http://dx.doi.org/10.1016/j.ijforecast.2021.10.004.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., et al. (2017). LightGBM: A highly efficient gradient boosting decision tree. In *Advances in neural information processing systems 30* (pp. 3146–3154). Curran Associates, Inc..

Kunz, M., Birr, S., Raslan, M., Ma, L., Li, Z., Gouttes, A., et al. (2023). *Deep Learning Based Forecasting: A Case Study from the Online Fashion Industry*. http://dx.doi.org/10.48550/arXiv.2305.14406, arXiv, arXiv:2305.14406.

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., et al. (2019). Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In *Advances in neural information processing systems 32* (pp. 5244–5254). Curran Associates, Inc..

Lim, B., Arık, S. Ö., Loeff, N., & Pfister, T. (2021). Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, *37*(4), 1748–1764. http://dx.doi.org/10.1016/j.ijforecast.2021.03.012.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2021). The M5 competition: Background, organization, and implementation. *International Journal of Forecasting*, http://dx.doi.org/10.1016/j.ijforecast.2021.07.007.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2022). M5 accuracy competition: Results, findings, and conclusions. *International Journal of Forecasting*, *38*(4), 1346–1364. http://dx.doi.org/10.1016/j.ijforecast.2021.11.013.

Panagiotelis, A., Athanasopoulos, G., Gamakumara, P., & Hyndman, R. J. (2021). Forecast reconciliation: A geometric view with new insights on bias correction. *International Journal of Forecasting*, *37*(1), 343–359. http://dx.doi.org/10.1016/j.ijforecast.2020.06.004.

Rangapuram, S. S., Kapoor, S., Nirwan, R. S., Mercado, P., Januschowski, T., Wang, Y., et al. (2023). Coherent probabilistic forecasting of temporal hierarchies. In *Proceedings of the 26th international conference on artificial intelligence and statistics* (pp. 9362–9376). PMLR.

Rangapuram, S. S., Werner, L. D., Benidis, K., Mercado, P., Gasthaus, J., & Januschowski, T. (2021). End-to-end learning of coherent probabilistic forecasts for hierarchical time series. In *Proceedings of the 38th international conference on machine learning* (pp. 8832–8843). PMLR.

Schäfer, J., & Strimmer, K. (2005). A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics. *Statistical Applications in Genetics and Molecular Biology*, *4*(1), http://dx.doi.org/10.2202/1544-6115.1175.

Sprangers, O., Schelter, S., & de Rijke, M. (2021). Probabilistic gradient boosting machines for large-scale probabilistic regression. In *Proceedings of the 27th ACM SIGKDD Conferenceon Knowledge Discovery & Data Mining* (pp. 1510–1520). New York, NY, USA: Association for Computing Machinery, http://dx.doi.org/10.1145/3447548.3467278.

Stankeviciute, K., Alaa, A. M., & van der Schaar, M. (2021). Conformaltime-series forecasting. In *Advances in Neural InformationProcessing Systems, vol. 34* (pp. 6216–6228). Curran Associates, Inc.

Theodosiou, F., & Kourentzes, N. (2021). *Forecasting with Deep Temporal Hierarchies, No. 3918315*. Rochester, NY: http://dx.doi.org/10.2139/ssrn.3918315.

Touloumis, A. (2015). Nonparametric stein-type shrinkage covariance matrix estimators in high-dimensional settings. *Computational Statistics & Data Analysis*, *83*, 251–261. http://dx.doi.org/10.1016/j.csda.2014.10.018, arXiv:1410.4726.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., et al. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*, 261–272. http://dx.doi.org/10.1038/s41592-019-0686-2.

Wickramasuriya, S. L., Athanasopoulos, G., & Hyndman, R. J. (2019). Optimal forecast reconciliation for hierarchical and grouped time series through trace minimization. *Journal of the American Statistical Association*, *114*(526), 804–819. http://dx.doi.org/10.1080/01621459.2018.1448825.