ARTICLE IN PRESS
International Journal of Forecasting xxx (xxxx) xxx

Contents lists available at ScienceDirect

International Journal of Forecasting

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

Optimal hierarchical EWMA forecasting

Giacomo Sbrana [a,*], Matteo Pelagatti [b]

[a] NEOMA Business School, 1 Rue du Marechal Juin, 76130 Mont-Saint-Aignan, France
[b] Univeristy of Milano-Bicocca, Via Bicocca degli Arcimboldi 8, 20126 Milan, Italy

ARTICLE INFO

Keywords:
Multivariate EWMA
State-space models
Kalman filter
Likelihood estimation
Expectation-maximization algorithm

ABSTRACT

Prediction of demand at different levels of aggregation is a crucial task in many business and industrial activities. This task may be extremely challenging when the number of time series increases together with the number of parameters governing the dynamics of the underlying model. This paper proposes theoretical and empirical contributions providing practical tools for managers needing efficient, flexible, and timely instruments. We first derive optimal results for predicting a system of time series following multivariate *Exponentially Weighted Moving Average* (EWMA) dynamics. Our results have relevant practical consequences. Indeed, we propose a fast EM algorithm that maximizes the Gaussian multivariate likelihood regardless of the model's dimension. Secondly, we show optimal results for the hierarchies, deriving closed-form results for the underlying parameters. Finally, using more than one hundred Walmart sales time series, we show that our approach is competitive with the optimal forecast reconciliation approach based on univariate forecasts.

1. Introduction

The Exponentially Weighted Moving Average (EWMA) model represents a widely used benchmark when forecasting time series in business and industry. This approach has a long tradition and dates back to Brown (1959), Magee (1958), Muth (1960) and Winters (1960). Due to its simplicity and notable forecasting performance, the EWMA is one of the most popular approaches in predicting time series (Hyndman et al., 2008). It is well known that the exponential smoothing predictor is optimal when data are generated by a random walk plus white noise. Or, equivalently, by a first-order integrated moving average process, an IMA(1,1), whose forecasting ability in many real-world situations is well recognized by the literature. Indeed, according to the 2003 Nobel prize winner Clive Granger (Granger & Newbold, 1977, p.203):

> This model provides an excellent representation of a wide range of economic time series [...] we do not advocate the adoption of this model in all occasions. However, suppose some simple, specific model is to be assumed on a priori grounds. In that case, we feel that the first-order integrated moving average process is a serious candidate for economic time series in general.

The generalization of this simple model in the multivariate context was first discussed by Jones (1966) and subsequently by Enns et al. (1982) and Harvey (1986). However, the multivariate version has attracted less attention due to computational issues related to estimation. A notable exception is represented by Poloni and Sbrana (2015), who recently proposed a univariate approach defined as *Moments estimation through aggregation* (i.e., META). The main advantage of this approach is its simplicity since it is based only on univariate estimation. However, one of the disadvantages of this approach is that it does not guarantee that the estimated model's

* Corresponding author.
E-mail addresses: giacomo.sbrana@neoma-bs.fr (G. Sbrana), matteo.pelagatti@unimib.it (M. Pelagatti).

https://doi.org/10.1016/j.ijforecast.2022.12.008

Please cite this article as: G. Sbrana and M. Pelagatti, Optimal hierarchical EWMA forecasting. International Journal of Forecasting (2023), https://doi.org/10.1016/j.ijforecast.2022.12.008.

covariance matrix is positive definite (see discussions in Poloni & Sbrana, 2017; Sbrana et al., 2017).

This work introduces a novel method for estimating large-scale multivariate EWMA using its state-space form. First, we prove closed-form results for the multivariate EWMA system. Secondly, we propose a fast EM algorithm achieving the quick maximization of an approximation of the Gaussian multivariate likelihood regardless of the model's dimension. Our estimation method is asymptotically equivalent to standard Gaussian maximum likelihood. However, it drastically reduces computational times and, hence, makes forecasting large vectors of time series feasible.

A natural application of our approach is the prediction of hierarchies. Indeed, the usual approaches to hierarchical forecasting[1] are based on univariate models applied either to the lowest level of the hierarchy (bottom-up) and summed up to match the aggregate forecasts, or to the top-level time series (top-down), whose forecasts are, then, disaggregated according to some weighting rule (see Athanasopoulos et al., 2009, for an implementation of these two approaches). Univariate models have the advantage of being simple to apply and computationally light, but they neglect the use of common information shared by the time series belonging to the hierarchy.

Lütkepohl (1984) proved that if the data generating process (DGP) is a known vector ARMA model, the aggregation of forecasts based on the multivariate model is more accurate (in terms of mean squared errors) than the univariate predictions of aggregated time series based on the implied univariate specifications. The result is not surprising as univariate models use only a subset of the whole information set exploited by multivariate models. Of course, if the DGP is unknown and its estimation is based on a finite sample of observations, then there is no unique answer to the aggregation problem, and evidence may be mixed. In empirical applications, Lütkepohl (1984) finds that the aggregations of multivariate forecasts outperform univariate aggregated forecasts only for short time horizons. We find similar results in our empirical analysis in Section 5.

What we propose for the hierarchical framework can be defined as a *multivariate* bottom-up approach. Indeed, we first estimate the EWMA vector using the bottom-level time series and then aggregate their forecasts to predict the desired hierarchies. In principle, the higher-level time series could be added to the vector, and reconciliation methods such as those in Hyndman et al. (2011) or Wickramasuriya et al. (2019) could be applied to the forecasts generated by our multivariate EWMA. In particular, using our approach, an estimate of the covariance matrix of the h-step-ahead forecast errors is readily available and can be plugged into the optimal reconciliation formula of Wickramasuriya et al. (2019, Theorem 1).

Pennings and van Dalen (2017) and Villegas and Pedregal (2018) propose reconciliation methods based on the state-space representation of the hierarchy. Interestingly, Pennings and van Dalen (2017) consider the EWMA as a baseline model (see their Eq. (6)). Their approach is theoretically attractive, but their inference is unfeasible for high-dimensional systems because of the computational burden of the Kalman filter on large vectors. Finding a way to integrate our efficient estimation method with the proposals of Pennings and van Dalen (2017) and Villegas and Pedregal (2018) (e.g., by a two-step approach) will be an objective of our future research.

We compare the performance of our approach to those of univariate bottom-up, top-down, and the optimal reconciliation method proposed by Hyndman et al. (2011, 2016), and Wickramasuriya et al. (2019) using Walmart retail sales data as recently employed in the M5 competition (Makridakis et al., 2022).

There is a vast literature that compares bottom-up and top-down approaches based on univariate models (see for example Syntetos et al., 2016). However, to the best of our knowledge, our work is the only one comparing a multivariate bottom-up approach to top-down and reconciliation approaches based on univariate models.

Our algorithm has been implemented using *R* (R Core Team, 2020) and its associated *C++* integration libraries, namely, *Rcpp* (Eddelbuettel & François, 2011) and *RcppArmadillo* (Eddelbuettel & Sanderson, 2014), the latter providing high-performance matrix algebra. Our functions are available upon request but will soon be publicly posted as an open-source package.

The paper is organized as follows: Section 2 introduces the multivariate EWMA framework together with the main theoretical results. Section 3 extends the results to the case of hierarchical modeling. Section 4 describes the EM algorithm that maximizes the approximate likelihood of the model regardless of its dimension. Section 5 compares our method to relevant competitors using the Walmart retail sales dataset. Section 6 draws some conclusions and suggests future research directions following the results contained in this work.

2. Main results

In what follows, we use this convention: $\boldsymbol{M}$ is a matrix and $\boldsymbol{M}'$ is its transpose, $\boldsymbol{m}$ is a column vector such that $\boldsymbol{m}'$ is a row vector. A lower-case non-bold letter, such as x, represents a scalar. Finally, 0 is used indiscriminately for matrices, vectors, and scalars.

Consider the state-space representation of the multivariate EWMA[2] process:

$$\begin{aligned} \boldsymbol{y}_t &= \boldsymbol{\alpha}_t + \boldsymbol{\epsilon}_t, \qquad \boldsymbol{\epsilon}_t \sim \mathrm{WN}(\boldsymbol{\Sigma}_\epsilon) \\ \boldsymbol{\alpha}_{t+1} &= \boldsymbol{\alpha}_t + \boldsymbol{\eta}_t, \qquad \boldsymbol{\eta}_t \sim \mathrm{WN}(\boldsymbol{\Sigma}_\eta) \end{aligned} \tag{1}$$

where $t = 1, \ldots, n$, and $\mathrm{WN}(\boldsymbol{\Sigma})$ denotes a vector white noise sequence with covariance matrix $\boldsymbol{\Sigma}$. The definition of the first two moments of $\boldsymbol{\alpha}_1$, say $\boldsymbol{a}_1$ and $\boldsymbol{P}_1$, completes the system. In what follows it is assumed that $\boldsymbol{\Sigma}_\epsilon$ and $\boldsymbol{\Sigma}_\eta$ are symmetric positive definite matrices and $E(\boldsymbol{\epsilon}_t\boldsymbol{\eta}_s') = 0$ for any choice of t and s. Note that all vectors and matrices in (1) have dimensions d and $d \times d$, respectively. The maximum likelihood estimation of the model (1) requires the

[1] The reader not acquainted with hierarchical forecasting should consider reading Chapter 11 of Hyndman and Athanasopoulos (2021).

[2] We use the names *EWMA*, *random walk plus white noise*, and *local-level* processes interchangeably.

implementation of the Kalman filter. Its implementation is feasible for models with moderate values of d. However, when the dimension d is large, the standard Kalman filer-based approach becomes infeasible as a single run of the Kalman filter requires several matrix multiplications and inversions of $d \times d$ matrices. In what follows, we provide closed-form results that fully solve these issues.

The Kalman filter recursions for the multivariate EWMA model can be written as follows: for $t = 1, 2, \ldots, n$,

Innovation : $\boldsymbol{v}_t = \boldsymbol{y}_t - \boldsymbol{a}_t$,

Innovation variance : $\boldsymbol{F}_t = \boldsymbol{P}_t + \boldsymbol{\Sigma}_\epsilon$,

Kalman gain : $\boldsymbol{K}_t = \boldsymbol{P}_t \boldsymbol{F}_t^{-1}$,

Prediction : $\boldsymbol{a}_{t+1} = \boldsymbol{a}_t + \boldsymbol{K}_t \boldsymbol{v}_t$,

Prediction error covariance : $\boldsymbol{P}_{t+1} = \boldsymbol{P}_t - \boldsymbol{P}_t \boldsymbol{F}_t^{-1} \boldsymbol{P}_t + \boldsymbol{\Sigma}_\eta$.

It is well-known that after a few observations, $\boldsymbol{P}_t$ and $\boldsymbol{F}_t$, and $\boldsymbol{K}_t$ converge to a steady-state solution. The knowledge of the steady-state solution leads to considerable computational savings since many of the above recursions become redundant (see discussions in Durbin & Koopman, 2012; Harvey, 1989).

Indeed, in steady-state, $\boldsymbol{P}_t$, $\boldsymbol{F}_t$, and $\boldsymbol{K}_t$ are the constants matrices $\boldsymbol{P}$, $\boldsymbol{F}$, and $\boldsymbol{K}$ that solve the following equations:

$$\begin{aligned} \boldsymbol{P} &= \boldsymbol{P} - \boldsymbol{P}(\boldsymbol{P} + \boldsymbol{\Sigma}_\epsilon)^{-1}\boldsymbol{P} + \boldsymbol{\Sigma}_\eta \\ \boldsymbol{F} &= \boldsymbol{P} + \boldsymbol{\Sigma}_\epsilon \\ \boldsymbol{K} &= \boldsymbol{P}\boldsymbol{F}^{-1}. \end{aligned} \tag{2}$$

The following result provides the algebraic (matrix) solution of the three Eqs. (2), that is, the algebraic link between $\boldsymbol{P}$, $\boldsymbol{F}$, $\boldsymbol{K}$, and $\boldsymbol{\Sigma}_\epsilon$, $\boldsymbol{\Sigma}_\eta$.

Theorem 1. *Consider the system as in (1) where $\boldsymbol{\Sigma}_\eta$ is positive semi-definite while $\boldsymbol{\Sigma}_\epsilon$ is strictly positive definite. Moreover, consider the following Cholesky decomposition $\boldsymbol{\Sigma}_\epsilon = \boldsymbol{M}\boldsymbol{M}'$ and define $\boldsymbol{Q} = \boldsymbol{M}^{-1}\boldsymbol{\Sigma}_\eta \boldsymbol{M}^{-1'} = \boldsymbol{\Psi}\boldsymbol{\Delta}\boldsymbol{\Psi}'$ such that for the last eigendecomposition $\boldsymbol{\Psi}$ is a matrix of eigenvectors (i.e. $\boldsymbol{\Psi}\boldsymbol{\Psi}' = \boldsymbol{I}$) and $\boldsymbol{\Delta} = \mathrm{diag}(\delta_1, \delta_2, \ldots, \delta_d)$ is a diagonal matrix of eigenvalues. Then there exists a unique positive definite solution for $\boldsymbol{P}$. The solution is*

$$\boldsymbol{P} = \frac{1}{2}\boldsymbol{M}\boldsymbol{\Psi}\left[\boldsymbol{\Delta} + (\boldsymbol{\Delta}^2 + 4\boldsymbol{\Delta})^{\frac{1}{2}}\right]\boldsymbol{\Psi}'\boldsymbol{M}'. \tag{3}$$

In addition, we have that

$$\boldsymbol{F} = \frac{1}{2}\boldsymbol{M}\boldsymbol{\Psi}(2\boldsymbol{I} + \boldsymbol{\Delta} + (\boldsymbol{\Delta}^2 + 4\boldsymbol{\Delta})^{\frac{1}{2}})\boldsymbol{\Psi}'\boldsymbol{M}'.$$

Finally, the Kalman gain is

$$\boldsymbol{K} = \boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{\Lambda}\boldsymbol{\Psi}^{-1}\boldsymbol{M}^{-1}, \tag{4}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix whose elements are

$$\lambda_j = \frac{\delta_j + \sqrt{\delta_j^2 + 4\delta_j}}{2 + \delta_j + \sqrt{\delta_j^2 + 4\delta_j}} \qquad j = 1, \ldots, d. \tag{5}$$

Proof. Consider the following transformation $\tilde{\boldsymbol{y}}_t := \boldsymbol{\Psi}^{-1}\boldsymbol{M}^{-1}\mathbf{y}_t$, such that the system as in (1) can be re-parametrized as

$$\begin{aligned} \tilde{\boldsymbol{y}}_t &= \tilde{\boldsymbol{\alpha}}_t + \tilde{\boldsymbol{\epsilon}}_t & \tilde{\boldsymbol{\epsilon}}_t &\sim WN(\mathbf{I}) \\ \tilde{\boldsymbol{\alpha}}_{t+1} &= \tilde{\boldsymbol{\alpha}}_t + \tilde{\boldsymbol{\eta}}_t & \tilde{\boldsymbol{\eta}}_t &\sim WN(\boldsymbol{\Delta}) \end{aligned} \tag{6}$$

for $t = 1, 2, \ldots, n$. In the last system, each equation is decoupled from the others such that it can be handled separately. Therefore, using the well-known results for univariate exponential smoothing (Harvey, 1986), one can obtain the Riccati solution for each equation, say $\bar{p}_j = \left(\delta_j + \sqrt{\delta_j^2 + 4\delta_j}\right)/2$ where $j = 1, \ldots, d$. In addition, by stacking these solutions into a diagonal matrix, we obtain $\left(\Delta + (\Delta^2 + 4\Delta)^{\frac{1}{2}}\right)/2$. Pre-and post-multiplying by $\boldsymbol{M}\boldsymbol{\Psi}$ the solution (3) is obtained. Finally, one can immediately derive the steady-state results for $\boldsymbol{F}$ and $\boldsymbol{K}$. Indeed, in the steady state, we have that:

$$\boldsymbol{F} = \boldsymbol{P} + \boldsymbol{\Sigma}_\epsilon = \frac{1}{2}\boldsymbol{M}\boldsymbol{\Psi}\left(2\boldsymbol{I} + \boldsymbol{\Delta} + (\boldsymbol{\Delta}^2 + 4\boldsymbol{\Delta})^{\frac{1}{2}}\right)\boldsymbol{\Psi}'\boldsymbol{M}'$$

Finally, the Kalman gain is

$$\boldsymbol{K} = \boldsymbol{P}\boldsymbol{F}^{-1} = \boldsymbol{M}\boldsymbol{\Psi}\boldsymbol{\Lambda}\boldsymbol{\Psi}^{-1}\boldsymbol{M}^{-1}, \tag{7}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix whose elements correspond to those in (5). □

3. Hierarchical results

We now turn our attention to the hierarchical EWMA model. In particular, assuming that the variables follow EWMA dynamics as in (1), we aim to find the optimal weight matrix for the hierarchies $\boldsymbol{y}_t^h = \boldsymbol{S}\boldsymbol{y}_t$, where $\boldsymbol{S}$ represents a matrix of dimension $(g \times d)$ composed of zeros, ones, or proportions, depending on the type of aggregations considered. Here g represents the number of hierarchies. Note that the superscript h is added to discriminate the hierarchical model from model (1). Therefore, assuming (1) as the data generation process, we have that:

$$\begin{aligned} \boldsymbol{y}_t^h &= \boldsymbol{\alpha}_t^h + \boldsymbol{\epsilon}_t^h, & \boldsymbol{\epsilon}_t^h &\sim \mathrm{WN}(\boldsymbol{S}\boldsymbol{\Sigma}_\epsilon\boldsymbol{S}') \\ \boldsymbol{\alpha}_{t+1}^h &= \boldsymbol{\alpha}_t^h + \boldsymbol{\eta}_t^h, & \boldsymbol{\eta}_t^h &\sim \mathrm{WN}(\boldsymbol{S}\boldsymbol{\Sigma}_\eta\boldsymbol{S}') \\ \boldsymbol{a}_1^h &= \boldsymbol{S}\boldsymbol{a}_1, & \boldsymbol{P}_1^h &= \boldsymbol{S}\boldsymbol{P}_1\boldsymbol{S}'. \end{aligned} \tag{8}$$

Note that all vectors and matrices in (8) have dimensions g and $g \times g$, respectively. The Kalman filter recursions for the hierarchical model can then be written as follows:

Innovation : $\boldsymbol{v}_t^h = \boldsymbol{y}_t^h - \boldsymbol{a}_t^h$,

Innovation variance : $\boldsymbol{F}_t^h = \boldsymbol{P}_t^h + \boldsymbol{S}\boldsymbol{\Sigma}_\epsilon\boldsymbol{S}'$,

Kalman gain : $\boldsymbol{K}_t^h = \boldsymbol{P}_t^h(\boldsymbol{F}_t^h)^{-1}$,

Prediction : $\boldsymbol{a}_{t+1}^h = \boldsymbol{a}_t^h + \boldsymbol{K}_t^h\boldsymbol{v}_t^h$,

Prediction error : $\boldsymbol{P}_{t+1}^h = \boldsymbol{P}_t^h - \boldsymbol{P}_t^h(\boldsymbol{F}_t^h)^{-1}\boldsymbol{P}_t^h + \boldsymbol{S}\boldsymbol{\Sigma}_\eta\boldsymbol{S}'$.

The quantities $\boldsymbol{P}_t^h$, $\boldsymbol{F}_t^h$, and $\boldsymbol{K}_t^h$ also converge to their steady-state solutions. The following theorem shows how $\boldsymbol{P}^h$, $\boldsymbol{F}^h$, and $\boldsymbol{K}^h$ can be expressed as algebraic functions of $\boldsymbol{S}\boldsymbol{\Sigma}_\epsilon\boldsymbol{S}'$ and $\boldsymbol{S}\boldsymbol{\Sigma}_\eta\boldsymbol{S}'$.

Theorem 2. *Consider the system as in (8) where $\boldsymbol{\Sigma}_\eta$ is positive semi-definite while $\boldsymbol{\Sigma}_\epsilon$ is strictly positive definite. Moreover, consider the following Cholesky decomposition $\boldsymbol{S}\boldsymbol{\Sigma}_\epsilon\boldsymbol{S}' = \boldsymbol{W}\boldsymbol{W}'$ and define $\boldsymbol{R} = \boldsymbol{W}^{-1}\boldsymbol{S}\boldsymbol{\Sigma}_\eta\boldsymbol{S}'\boldsymbol{W}^{-1'} = \boldsymbol{\Phi}\boldsymbol{\Theta}\boldsymbol{\Phi}'$ such that for the last eigen-decomposition $\boldsymbol{\Phi}$ is a matrix of eigenvectors (i.e. $\boldsymbol{\Phi}\boldsymbol{\Phi}' = \boldsymbol{I}$) and $\boldsymbol{\Theta} = \text{diag}(\theta_1, \ldots, \theta_g)$ is a diagonal matrix of eigenvalues. Then, there exists a unique positive definite solution for $\boldsymbol{P}^h$, $\boldsymbol{F}^h$, and $\boldsymbol{K}^h$. The solutions are the following:*

$$\boldsymbol{P}^h = \frac{1}{2}\boldsymbol{W}\boldsymbol{\Phi}\left[\boldsymbol{\Theta} + (\boldsymbol{\Theta}^2 + 4\boldsymbol{\Theta})^{\frac{1}{2}}\right]\boldsymbol{\Phi}'\boldsymbol{W}'. \tag{9}$$

$$\boldsymbol{F}^h = \frac{1}{2}\boldsymbol{W}\boldsymbol{\Phi}\left(2\boldsymbol{I} + \boldsymbol{\Theta} + (\boldsymbol{\Theta}^2 + 4\boldsymbol{\Theta})^{\frac{1}{2}}\right)\boldsymbol{\Phi}'\boldsymbol{W}'. \tag{10}$$

$$\boldsymbol{K}^h = \boldsymbol{W}\boldsymbol{\Phi}\boldsymbol{\Omega}\boldsymbol{\Phi}^{-1}\boldsymbol{W}^{-1}, \tag{11}$$

where $\boldsymbol{\Omega}$ is a diagonal matrix whose elements are

$$\omega_j = \frac{\theta_j + \sqrt{\theta_j^2 + 4\theta_j}}{2 + \theta_j + \sqrt{\theta_j^2 + 4\theta_j}} \qquad j = 1, \ldots, d.$$

Proof. The proof follows the same lines as that of Theorem 1. □

As shown below, (11) represents the only "optimal" ingredient needed for hierarchical forecasting.

3.1. Optimal hierarchical approach

Here, we show how the results can be used for obtaining the desired hierarchical forecasts. As noted, once in steady-state, the Kalman filter recursions for the hierarchical model are the following:

Innovation : $\boldsymbol{v}_t^h = \boldsymbol{y}_t^h - \boldsymbol{a}_t^h$,

Prediction : $\boldsymbol{a}_{t+1}^h = \boldsymbol{a}_t^h + \boldsymbol{K}^h\boldsymbol{v}_t^h$.

Combining the above equations, we obtain the celebrated EWMA recursion:

$$\begin{aligned}\boldsymbol{a}_{t+1}^h &= \boldsymbol{a}_t^h + \boldsymbol{K}^h(\boldsymbol{y}_t^h - \boldsymbol{a}_t^h) = \\ &= \boldsymbol{K}^h\boldsymbol{y}_t^h + (\boldsymbol{I} - \boldsymbol{K}^h)\boldsymbol{a}_t^h = \\ &= \sum_{j=0}^{\infty}\boldsymbol{K}^h(\boldsymbol{I} - \boldsymbol{K}^h)^j\boldsymbol{y}_{t-j}^h,\end{aligned} \tag{12}$$

where $\boldsymbol{K}^h$ represents the "optimal" weight matrix since it is the exact matrix, implied by the data generation process, needed when forecasting the hierarchies.

In empirical analyses, when the optimal weight matrix has to be estimated, one can use two approaches:

1. estimate the parameters of model (1) and derive indirectly (11);
2. estimate directly the parameters of the hierarchical model (8) and therefore (11).

Note that, in general, the first option is the most efficient procedure since it uses complete information. On the other hand, the second option might face issues related to aggregation bias since the estimation is carried out directly on the hierarchical model, being a transformation of the generation process.

From a computational point of view, the second option is faster than the first one since the dimension of the model is generally lower. However, in the next section, we provide a fast estimation process that allows estimating the state-space representation of the EWMA model regardless of the model's dimension.

4. A fast EM algorithm

The maximum likelihood estimation of state-space models using the EM algorithm is fully discussed by Shumway and Stoffer (1982) (see also Durbin & Koopman, 2012). In practice, the EM algorithm for models (1) and (8) can be implemented using the simple updating expressions of Section 3 in Koopman (1993). More specifically, for model (1) these expressions can be restated as follows:

$$\boldsymbol{\Sigma}_\epsilon(\iota+1) = \boldsymbol{\Sigma}_\epsilon(\iota) + \boldsymbol{\Sigma}_\epsilon(\iota)\boldsymbol{\Theta}_e\boldsymbol{\Sigma}_\epsilon(\iota), \tag{13}$$

$$\boldsymbol{\Sigma}_\eta(\iota+1) = \boldsymbol{\Sigma}_\eta(\iota) + \boldsymbol{\Sigma}_\eta(\iota)\boldsymbol{\Theta}_r\boldsymbol{\Sigma}_\eta(\iota), \tag{14}$$

where $\iota = 0, 1, \ldots$. Here $\boldsymbol{\Sigma}_\epsilon(0)$, $\boldsymbol{\Sigma}_\eta(0)$ are arbitrary starting values. In addition,

$$\boldsymbol{\Theta}_r = \frac{1}{n}\sum_{t=1}^{n}(\boldsymbol{r}_t\boldsymbol{r}_t' - \boldsymbol{N}_t) \tag{15}$$

where $\boldsymbol{r}_t$ and $\boldsymbol{N}_t$ are the smoothing recursions as provided in Durbin and Koopman (2012). Those can be computed as:

$$\boldsymbol{r}_{t-1} = \boldsymbol{F}^{-1}\boldsymbol{v}_t + \boldsymbol{L}'\boldsymbol{r}_t \tag{16}$$

$$\boldsymbol{N}_{t-1} = \boldsymbol{F}^{-1} + \boldsymbol{L}'\boldsymbol{N}_t\boldsymbol{L}. \tag{17}$$

For $t = 1, \ldots, n-1$. For $t = n$, $\boldsymbol{r}_n = \boldsymbol{0}$, $\boldsymbol{N}_n = \boldsymbol{0}$ (see details in chapter 4 of Durbin & Koopman, 2012). Note that $\boldsymbol{L} = \boldsymbol{I} - \boldsymbol{K}$. Moreover:

$$\boldsymbol{\Theta}_e = \frac{1}{n}\sum_{t=1}^{n}(\boldsymbol{e}_t\boldsymbol{e}_t' - \boldsymbol{D}_t) \tag{18}$$

with

$$\boldsymbol{e}_t = \boldsymbol{F}^{-1}\boldsymbol{v}_t - \boldsymbol{K}'\boldsymbol{r}_t \tag{19}$$

and

$$\boldsymbol{D}_t = \boldsymbol{F}^{-1} + \boldsymbol{K}'\boldsymbol{N}_t\boldsymbol{K}. \tag{20}$$

Note that (16), (17), (19), (20) use the steady-state matrices as obtained in Theorem 1. That is, rather than waiting for the convergence of the Kalman filter to its steady state, one can impose it at Kalman filter initialization (that is, at time $t = 1$). Imposing the steady state implies that one needs to run only the Kalman filter's Innovation and Prediction recursions. This approach has already been advocated by Harvey (1986), who defines it as Approximate Maximum Likelihood (see also Harvey & Peters, 1990, p.93). For example, for the univariate EWMA model, Harvey shows that the likelihood can be maximized numerically by choosing the signal-to-noise parameter that minimizes the sum of squared innovations (see Harvey, 1986, p.376). The same approach, requiring the knowledge of the steady-state quantities, can also be implemented using the EM algorithm.

We now show a computationally feasible procedure to approximate the maximum likelihood estimation of the multivariate EWMA model. This represents the second main contribution of this paper.

Algorithm 1 (*Approximate Maximum Likelihood Estimation for Models (1) and (8)*). Fix arbitrary initial covariance matrices $\boldsymbol{\Sigma}_\epsilon$ and $\boldsymbol{\Sigma}_\eta$ and iterate as follows.

1. Obtain $\boldsymbol{P}$ from equation (3) and compute the steady-state Kalman filter matrices and start the Kalman filter with $\boldsymbol{a}_1 = \boldsymbol{y}_1$ (see Harvey, 1989, p.26).
2. Run $\boldsymbol{a}_t = \boldsymbol{K}\boldsymbol{y}_t + (\boldsymbol{I} - \boldsymbol{K})\boldsymbol{a}_{t-1}$ and for $t = 2, \dots, n$.
3. Run the steady-state smoothing formulae (16), (17), (19), (20), (15), and (18).
4. Run the EM step updating the parameters using (13) and (14).
5. Go to step 1. until the likelihood increment is negligible.
6. (For hierarchies) Imply (11) using the estimated $\boldsymbol{\Sigma}_\epsilon$, $\boldsymbol{\Sigma}_\eta$, and $\boldsymbol{S}$.

This procedure can be used both for estimating the parameters of models (1) and (8).

For the hierarchical model (8) an alternative algorithm is the following:

Algorithm 2 (*Direct Approximate maximum Likelihood Estimation for Model (8)*). First, establish $\boldsymbol{S}$ and fix arbitrary initial covariance matrices $\boldsymbol{S}\boldsymbol{\Sigma}_\epsilon\boldsymbol{S}'$ and $\boldsymbol{S}\boldsymbol{\Sigma}_\eta\boldsymbol{S}'$. Then iterate as follows.

1. Obtain $\boldsymbol{P}^h$ from equation (9) and compute the steady-state Kalman filter matrices and start the Kalman filter with $\boldsymbol{m}_1 = \boldsymbol{y}_1$.
2. Run $\boldsymbol{a}^h_{t+1} = \boldsymbol{K}^h\boldsymbol{y}^h_t + (\boldsymbol{I} - \boldsymbol{K}^h)\boldsymbol{a}^h_t$ and for $t = 1, \dots, n-1$.
3. Run the steady-state smoothing formulae (16), (17), (19), (20), (15), and (18).
4. Run the EM step updating the parameters using (13) and (14).
5. Go to step 1. until the likelihood increment is negligible.

One potential computational issue may arise when choosing arbitrary initial covariance matrices $\boldsymbol{\Sigma}_\epsilon$ and $\boldsymbol{\Sigma}_\eta$. For this reason, we recommend using diagonal matrices to avoid issues with non-positive covariances, as already discussed with the META approach advocated by Poloni and Sbrana (2015). In particular, we suggest filling the diagonal matrices $\boldsymbol{\Sigma}_\epsilon$ and $\boldsymbol{\Sigma}_\eta$ using the parameters estimated on the single time series. This is what we used in the empirical analysis below.

In principle, both algorithms can be employed when forecasting hierarchies. However, while Algorithm 2 estimates the optimal weight matrix directly, Algorithm 1 targets the whole system first and, then, derives the optimal weight matrix $\boldsymbol{K}$ analytically.

Assuming that (1) is the generation process, the first option represents the most efficient procedure since it uses the complete information. On the other hand, the second option might face issues since the estimation is carried out directly on a transformation of the generation process. Indeed, in our empirical application below, we show that the first option is superior.

In practice, however, since the generation process is usually unknown, we recommend checking which option provides more accurate results.

From a computational point of view, the second option is faster than the first one since the dimension is generally lower. However, given the suggested quick EM algorithm, we can reasonably claim that speed is not an issue.

4.1. Speed and accuracy of the algorithm

To assess the speed and precision of our approach and compare it to standard ML estimates based on Kalman filtering, we now show the results of a Monte Carlo experiment. We simulate sample paths of length $n = 1000$ from model (1) using Gaussian disturbances and random structural parameters generated as follows. Both matrices $\boldsymbol{\Sigma}_\eta$ and $\boldsymbol{\Sigma}_\epsilon$ were correlation matrices generated by the following steps:

1. we generated a matrix $\boldsymbol{A}$ of uniform numbers on $[0, 1]$;
2. we computed the symmetric matrix $\boldsymbol{B} = \boldsymbol{A}\boldsymbol{A}^\top$;
3. we extracted the eigenvalues and eigenvectors of $\boldsymbol{B}$ and built the matrix $\tilde{\boldsymbol{B}}$ with the same eigenvectors of $\boldsymbol{B}$ and with eigenvalues transformed as
$$\tilde{\lambda}_j = 1 + 29\frac{\lambda_j - \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}},$$
where $\lambda_{\max}, \dots, \lambda_{\min}$ are the ordered eigenvalues of $\boldsymbol{B}$;
4. finally, we set $\boldsymbol{\Sigma}$ equal to the correlation matrix obtained from the covariance matrix $\tilde{\boldsymbol{B}}$.

We used the eigenvalue transformation at step 3 to keep the condition number of the covariance matrices approximatively equal to 30 for any considered dimension and, thus, avoiding numerically non-invertible matrices.

Table 1 reports the timing (in seconds) and precision (in terms of mean absolute errors and root mean squared errors[3]) for both ML estimates based on the full Kalman filter and our EM algorithm, for different dimensions (i.e., $d = 3, 5, 10, 20, 40, 80, 160$).

It is clear from Table 1 that our approach provides huge computational gains compared to the standard ML, especially when the dimension of the system gets large. Moreover, considering that both estimation approaches are equally precise for any size d, this confirms that our approach is as accurate as the standard ML. When the system gets very large, the accuracy of the estimates tends to deteriorate. This is not surprising since, for $d = 160$, the EM needs to estimate 25760 parameters using only 1000 observations.

[3] The population covariances are correlations and, thus, all values are in the interval $[-1, 1]$. This makes our summary statistics (MAE and RMSE) meaningful as no single estimate dominates the others.

5. Hierarchical forecasting using walmart retail sales data

Forecasting retail sales is an important task for supermarkets. Indeed, the accurate prediction of sale dynamics provides several benefits regarding logistics, stock replenishment, and profits. We now show an empirical example of hierarchical forecasting when dealing with retail sales using Walmart data. In particular, we use the dataset already employed in the M5 competition whose data are available online[4]. We refer to Makridakis et al. (2022) for a detailed description of the data.

The Walmart dataset reports the number of items sold for each product observed from 2011-01-29 to 2016-06-19, at daily frequency, for a total of 1941 historical observations. Data refer to 3049 products sold in three states (i.e., California, Texas, and Wisconsin) and ten stores (four in California and three in Texas and Wisconsin). One of the main features of this dataset is the presence of many zeros. Since this feature may cause estimation issues, especially in our multivariate framework, we focus only on the time series of products with less than six zero observations. This corresponds to 101 time series belonging to the categories of *Foods*, *Hobbies*, and *Household*.[5]

The first relevant step consists in comparing the forecast performance of the multivariate model, using Algorithm 1, versus 101 univariate EWMA models estimated using exact maximum likelihood (see Harvey, 1989). Indeed, we wish to investigate the forecasting gains (if any) when using the whole information set rather than disaggregating the system into single unrelated equations. Given the high number of parameters ($101 \times 102 = 10302$), we employ our EM algorithm.

Our empirical exercise is carried out as follows: we use the first 1801 observations as a training sample; every seven days (corresponding to a week of sales), we produce 7-day-ahead forecasts with both approaches. We repeat this procedure until the end of the dataset so that we end up with 140 forecasts at horizons 1-, 2-, 3-, 4-, 5-, 6-, and 7-step ahead. Note that while keeping the starting point of the training sample fixed, we update the end point of the training sample by adding seven observations at a time once the 7-day forecasts are produced.

We used the mean squared error as the performance metric. However, given the huge amount of results, in Table 2, we report only the ratio of mean squared errors (univariate/multivariate): values that are larger than one signal the higher precision of our approach. In addition, we implemented the Diebold–Mariano test to establish if one model significantly outperforms its competitor for a specific time series.

In the tables, when the multivariate model significantly outperforms the univariate approach, we enhance this by displaying sequences of $^{\bullet}$ symbols with the following rationale: $^{\bullet\bullet\bullet}$ implies a p-value between 0 and 1%; $^{\bullet\bullet}$ implies a p-value between 1% and 5%; $^{\bullet}$ implies a p-value between 5% and 10%. On the other hand, when the univariate models provide significantly better results, we employ the same approach but using the symbol *.

Overall, Table 2 shows that there are advantages in using the full information rather than estimating an EWMA on each time series. Indeed, the results tend to be above one, and, in several cases, the DM test confirms that the multivariate model significantly outperforms the univariate.

We now turn to consider the hierarchical case. Our exercise consists of the following steps. Firstly, we create five hierarchies, aggregating the 101 items into five department groups, as shown in Table 3 in Appendix. Each group corresponds to the sum of products belonging to a specific category and department (for instance: G2 collects all products of the *FOODS* category of department 2, that is, all products indexed as follows *FOODS_2_XXX*). This aggregation is in line with the hierarchical structure of the dataset.

Secondly, using the same forecast horizons as above, we compare the performances of Algorithm 1 (multivariate bottom-up), Algorithm 2 (multivariate modeling of the five hierarchies), the direct univariate modeling of the five hierarchies using ETS and ARIMA, the classical univariate bottom-up approach, and the optimal combination of forecasts algorithm of Wickramasuriya et al. (2019). For the latter, we used the *R* package *"hts"* as in Hyndman et al. (2021), together with the ETS approach (Hyndman et al., 2008), and the Automatic ARIMA approach (Hyndman & Khandakar, 2008).

Our results show that Algorithm 1 is significantly more accurate than Algorithm 2 and the univariate bottom-up approach (results not reported but available on request). Moreover, Algorithm 1 also outperforms the univariate ETS-based direct forecasts of the hierarchies, as shown in Table 4. However, the direct ARIMA-based forecasts of the hierarchies do better than our method (see Table 5).

Finally, we compared the performance with the optimal combination approach of Wickramasuriya et al. (2019). As shown in Table 6, Algorithm 1 outperforms the optimal ETS combination approach. On the other hand, as shown in Table 7, the optimal ARIMA combination provides significantly more accurate results than those of Algorithm 1.

Therefore, our bottom-up approach based on Algorithm 1 outperforms all alternatives based on the same basic EWMA model or the richer class of ETS models. Only ARIMA models do better when forecasting the five hierarchies. This is not surprising since the ARIMA class is more flexible and versatile than pure EWMA, allowing for much richer parametrizations. On the other hand, Algorithm 1 imposes the same (simple) model on the entire system of equations.

6. Conclusions

The paper introduces closed-form results allowing the implementation of a fast EM algorithm to estimate large-scale multivariate EWMA systems.

The suggested algorithm can handle hundreds of variables, while the standard likelihood estimation is computationally feasible only for a small number of time series.

---

4 see the Kaggle website: *https://www.kaggle.com/c/m5-forecasting-accuracy*

5 The code is available upon request.

Moreover, our simulations show that the precision of our approach in estimating the model's parameters is comparable to that of ML estimation. Therefore, our estimation approach should be preferred when dealing with large systems of time series.

An empirical application using the 101 Walmart time series shows promising results. Indeed, the suggested approach outperforms forecasts based on univariate EWMA and, more generally, ETS on all levels of the hierarchy. The only method considered in the application outperforming our approach is the optimal reconciliation procedure using univariate ARIMA models. Overall, attacking the hierarchical forecasting problem using a multivariate bottom-up approach seems promising, despite the simplicity of the EWMA dynamics.

The results presented in this work open some new paths of research. Indeed, our steady-state Kalman filtering approach could be extended to more complex models featuring seasonal or trend components, making it applicable to a broader range of problems. Of course, closed-form results, such as that of Theorem 1, may not be available. However, one can rely on efficient numerical alternatives.

Moreover, when the number of parameters is very large, regularization and shrinkage are known to improve estimations and forecasts. Our EM algorithm naturally implements a form of shrinkage when iterations are stopped before full convergence. Indeed, EM moves monotonically towards the maximum of the objective function, and early stopping of the iterations can be seen as a form of shrinkage towards the initial covariance matrices. The direction of this shrinkage (initial values for the EM) and the stopping rule should be inquired.

Another way to reduce the number of parameters in large multivariate time series models is by factorization. Introducing a version of Theorem 1 and of the EM algorithm for a model in which the number of random walks is smaller than the number of time series could further improve the performance of forecasts and the estimation time. Similarly, the observational noise covariance matrix could also be factorized to reduce the number of parameters.

Finally, our approach can be adapted to fit into the reconciliation framework of Hyndman and Athanasopoulos (2021) and to *reconciliation through state-space representation* as in Villegas and Pedregal (2018) and Pennings and van Dalen (2017). In the latter case, a two-step approach is worth to be explored.

Appendix. Tables of the Walmart retail sales forecasts

See Tables 2–7.

Table 1
Comparisons of Kalman filter-based ML estimates with the estimates obtained by our EM algorithm on a multivariate local level model with $n = 1,000$ observations.

	Execution time (sec)			MAE Σ_ϵ		MAE Σ_η		RMSE Σ_ϵ		RMSE Σ_η	
d	ML	EM	Ratio	ML	EM	ML	EM	ML	EM	ML	EM
3	1.2	0.0	52	0.07	0.06	0.07	0.07	0.09	0.07	0.09	0.09
5	15.5	0.1	215	0.05	0.05	0.06	0.08	0.06	0.06	0.08	0.09
10	230.0	0.1	1605	0.06	0.05	0.07	0.08	0.08	0.07	0.09	0.10
20	7159.1	1.3	5570	0.06	0.07	0.07	0.06	0.07	0.08	0.09	0.08
40		11.9			0.08		0.09		0.09		0.10
80		49.4			0.07		0.09		0.08		0.11
160		131.2			0.09		0.10		0.11		0.13

[1] MAE refers to the mean absolute error and RMSE to the root mean squared error of the estimates with respect to the real value of the correlations.
[2] We stopped evaluating ML estimation after $d = 20$ as computation times became unfeasible.
[3] We set a tolerance of 10^{-5} and a maximum number of 100 iterations for the EM algorithm. We also tried 500 iterations, but the results were virtually the same, while execution times for large d were about five times longer. In high-dimensional systems, an early stop could generate *shrinkage estimators* and could be beneficial, as the related literature proves.
[4] Simulations were carried out on an Azure *Standard H16* virtual machine (16 vcpus, 112 GiB memory) running Linux Ubuntu 18.04, R 3.6.3, KFAS 1.4.6, Rcpp 1.0.8.3, RcppArmadillo 0.11.2.0.0.

Table 2
Multivariate vs. univariate EWMA forecasting performance.

	1-steps	2-steps	3-steps	4-steps	5-steps	6-steps	7-steps
FOODS-1–046	1.000	*1.001*	*1.001*	*1.000*	1.000	1.000	1.000
FOODS-1–054	*1.015*	*1.014*	0.998	*1.001*	*1.005*	0.982*	0.989
FOODS-1–085	*1.028*$^{\bullet\bullet}$	*1.013*	*1.003*	*1.015*	0.986	0.973	0.981
FOODS-1–086	0.967	0.986	0.970	*1.010*	0.991	0.988	0.970
FOODS-1–096	*1.011*	*1.001*	0.982	0.993	*1.006*	*1.004*	0.991
FOODS-1–172	*1.008*	*1.043*	*1.019*	*1.024*	*1.016*	*1.018*	*1.022*
FOODS-1–183	0.999	*1.024*	*1.042*	*1.099*$^{\bullet\bullet}$	*1.057*$^{\bullet}$	0.976	0.974
FOODS-1–218	*1.002*	*1.001*	*1.001*	*1.002*$^{\bullet\bullet}$	*1.000*	0.999	*1.000*
FOODS-2–013	*1.033*	*1.025*	*1.085*$^{\bullet\bullet}$	*1.105*$^{\bullet\bullet}$	*1.033*	0.964	*1.016*
FOODS-2–061	0.941	*1.013*	*1.030*	*1.036*	*1.001*	*1.072*	0.958
FOODS-2–065	0.990	0.991	*1.124*$^{\bullet\bullet}$	*1.189*$^{\bullet\bullet\bullet}$	*1.042*	0.951	0.950

(*continued on next page*)

Table 2 (*continued*).

	1-steps	2-steps	3-steps	4-steps	5-steps	6-steps	7-steps
FOODS-2–164	1.000	*1.075*	*1.091*	*1.001*	0.979	0.969	*1.006*
FOODS-2–171	0.980	0.965	*1.026*	*1.023*	*1.056*	0.950	0.947
FOODS-2–181	0.994	*1.166*	*1.132*••	*1.111*•	*1.126*	0.962	*1.002*
FOODS-2–204	*1.002*	*1.046*	0.999	0.975	*1.023*	0.972	0.994
FOODS-2–293	*1.154*	*1.160*	*1.260*••	*1.229*••	*1.263*•	0.890*	0.899
FOODS-2–322	0.883	*1.007*	*1.126*••	*1.155*••	*1.175*••	*1.022*	0.923*
FOODS-2–371	0.990	*1.013*	*1.044*•	*1.012*	*1.032*	0.977	0.995
FOODS-3–007	*1.011*	*1.064*	*1.058*	*1.045*	*1.030*	*1.017*	*1.021*
FOODS-3–011	0.990	*1.019*	*1.019*	*1.019*	0.995	*1.002*	0.998
FOODS-3–023	*1.017*	*1.001*	0.982	0.989	*1.023*	0.978	0.975
FOODS-3–042	*1.047*	*1.059*	*1.026*	*1.001*	*1.019*	0.986	0.952
FOODS-3–067	0.982	0.997	0.978	0.981	*1.019*	0.945**	*1.007*
FOODS-3–080	0.996**	0.997	0.997**	0.999	*1.001*	*1.002*	*1.000*
FOODS-3–089	0.965	*1.056*	0.987	*1.029*	*1.001*	0.951	0.950
FOODS-3–127	*1.001*	*1.001*	*1.001*	*1.001*	*1.000*	1.000	*1.000*
FOODS-3–144	*1.014*	*1.017*	0.989	*1.002*	*1.014*	*1.007*	*1.009*
FOODS-3–178	*1.012*	0.976	*1.033*	*1.022*	0.990	0.973	0.948**
FOODS-3–195	0.878	*1.126*	*1.059*	*1.066*	0.973	0.913	0.984
FOODS-3–217	*1.011*	0.980	*1.013*	0.994	*1.002*	0.985	0.981
FOODS-3–226	*1.002*••	*1.002*•	*1.002*•	*1.002*•	*1.004*••	*1.001*	*1.003*
FOODS-3–228	*1.002*	*1.003*••	*1.002*	*1.003*••	*1.001*	0.999	0.999
FOODS-3–230	*1.002*	*1.002*	*1.002*	*1.002*	*1.002*	*1.001*	*1.001*
FOODS-3–232	*1.014*	*1.023*•••	*1.021*••	0.998	0.986	0.982**	0.978***
FOODS-3–234	0.996	*1.021*	0.952	*1.025*	*1.105*•	0.985	0.987
FOODS-3–242	*1.014*	0.993	*1.003*	*1.005*	0.979	0.983	0.979*
FOODS-3–252	*1.004*••	*1.004*••	*1.003*••	*1.003*•	*1.004*	0.998	*1.001*
FOODS-3–265	0.995	0.973	0.984	0.967*	*1.005*	0.964*⁎	0.992
FOODS-3–304	0.962	0.953*	*1.014*	0.970	0.969	0.989	1.000
FOODS-3–331	*1.025*	*1.033*	*1.005*	0.973	0.925*	0.961	0.963*
FOODS-3–349	*1.001*••	*1.001*••	*1.002*•••	*1.001*•	*1.001*	*1.000*	*1.000*
FOODS-3–363	*1.070*••	*1.059*••	*1.066*••	*1.038*	0.945***	0.959**	0.964
FOODS-3–369	0.990	*1.009*	*1.003*	0.992	0.995	*1.022*	0.979*
FOODS-3–377	0.999	1.000	0.999	*1.000*	*1.002*	*1.002*•	*1.003*••
FOODS-3–389	*1.003*•••	*1.003*•••	*1.004*•••	*1.004*•••	*1.005*•••	0.999	*1.002*
FOODS-3–400	0.985	*1.021*	*1.040*••	*1.053*•••	0.979	0.913**	0.949**
FOODS-3–407	*1.013*	0.988	*1.067*•	*1.026*	*1.004*	0.956	0.980
FOODS-3–412	0.937	*1.026*	*1.032*	*1.064*	*1.000*	0.976	0.967*
FOODS-3–458	*1.016*••	*1.021*	*1.024*••	*1.007*•	0.983	0.956*	0.934*
FOODS-3–473	*1.003*•••	*1.003*•••	*1.003*•••	*1.003*•••	*1.003*•••	*1.002*•	*1.001*
FOODS-3–475	*1.034*	*1.005*	*1.048*	0.994	*1.050*	0.990	0.967
FOODS-3–477	*1.086*	*1.135*••	*1.036*	*1.108*	0.943	0.931***	*1.025*
FOODS-3–495	*1.068*	*1.006*	0.987	*1.010*	*1.051*	0.999	0.947⁎
FOODS-3–498	0.993	0.957	*1.004*	*1.031*	*1.187*•••	0.974	*1.037*
FOODS-3–547	0.996	0.901	*1.042*	*1.017*	*1.135*	0.922	0.920
FOODS-3–555	0.998	0.999	0.999	1.000	0.999	*1.001*	1.000
FOODS-3–562	0.991	0.991	*1.037*	*1.034*	*1.045*	0.987	*1.002*
FOODS-3–580	*1.013*	0.937	*1.078*	*1.103*••	*1.132*•	0.995	0.927*
FOODS-3–584	0.986	0.953	0.989	0.960	0.991	0.945	0.922**
FOODS-3–586	*1.009*•••	*1.008*•••	*1.008*•••	*1.008*•••	*1.009*•••	0.996	0.999
FOODS-3–610	0.998	*1.026*	*1.014*	*1.004*	*1.012*	0.959	0.975
FOODS-3–654	*1.001*••	*1.001*•	*1.001*••	*1.001*	*1.001*	*1.000*	*1.000*
FOODS-3–668	*1.021*	0.970	*1.092*	*1.000*	0.891	0.916**	0.929
FOODS-3–672	0.956	*1.169*	*1.286*	*1.090*	*1.110*	0.986	0.740**
FOODS-3–674	0.946	*1.061*	*1.029*	*1.046*	*1.058*	*1.004*	0.969
FOODS-3–694	0.999	1.000	1.000	*1.000*	*1.001*	*1.002*	*1.002*
FOODS-3–697	0.959	0.961**	0.977*	0.974**	*1.032*••	*1.020*•••	*1.015*••
FOODS-3–711	0.880	0.880⁎	0.963	0.886	0.982	0.870	0.912
FOODS-3–714	*1.003*•	*1.003*••	*1.003*••	*1.003*•	*1.004*	0.999	*1.002*
FOODS-3–730	*1.039*	*1.038*	*1.054*	*1.071*•	0.978	0.954**	0.917*
FOODS-3–752	0.977	*1.008*	*1.027*	*1.033*	0.978	0.955	0.965
FOODS-3–792	0.958	*1.151*••	*1.041*	*1.024*	*1.118*	0.961	0.905*
FOODS-3–820	0.939	0.993	*1.012*	*1.013*	0.989	*1.013*	*1.035*
HOBBIES-1–015	0.989	0.999	*1.043*	0.977	*1.029*	0.969	*1.023*
HOBBIES-1–074	0.996	*1.008*•	*1.007*	*1.002*	*1.010*	0.989*	0.995
HOBBIES-1–103	*1.020*•	0.998	0.991	0.992	*1.003*	*1.003*	1.000
HOBBIES-1–254	*1.008*	0.992	0.997	*1.007*	*1.007*	0.989	0.986*
HOBBIES-1–323	0.970	0.984	*1.024*	*1.019*•	*1.016*	0.994	*1.002*
HOBBIES-1–381	*1.011*	*1.000*	0.997	*1.014*	1.000	1.000	0.993
HOUSEHOLD-1–055	*1.028*	0.979	0.983	0.946	0.966	*1.006*	0.980
HOUSEHOLD-1–072	*1.003*•••	*1.002*•••	*1.003*•••	*1.003*•••	*1.003*•••	*1.001*	*1.002*

(*continued on next page*)

Table 2 (*continued*).

	1-steps	2-steps	3-steps	4-steps	5-steps	6-steps	7-steps
HOUSEHOLD-1–083	*1.111*$^{\bullet}$	*1.111*$^{\bullet\bullet}$	*1.115*$^{\bullet}$	*1.049*	0.906	0.901*	0.937
HOUSEHOLD-1–087	*1.004*	*1.027*	*1.031*	*1.057*$^{\bullet\bullet}$	*1.050*$^{\bullet\bullet}$	0.973	0.972*
HOUSEHOLD-1–109	0.999	*1.110*$^{\bullet\bullet}$	*1.078*$^{\bullet}$	0.994	*1.076*	*1.006*	0.988
HOUSEHOLD-1–118	0.996	1.000	*1.007*	*1.003*	*1.006*	0.991	0.997
HOUSEHOLD-1–149	*1.005*	0.916*	0.990	0.945	*1.078*	0.945*	0.968
HOUSEHOLD-1–197	0.995	0.989	*1.020*	*1.021*	*1.000*	0.992	*1.012*
HOUSEHOLD-1–198	0.986	*1.005*	*1.016*	*1.065*	0.968	0.904*	0.957
HOUSEHOLD-1–235	0.915	0.904	0.932	0.892	*1.035*	0.861*	0.803*
HOUSEHOLD-1–243	*1.000*	0.985	0.995	0.994	0.993	0.985	0.979
HOUSEHOLD-1–247	*1.015*	*1.019*	0.904***	*1.063*$^{\bullet}$	*1.003*	0.991	0.981
HOUSEHOLD-1–251	*1.001*	*1.016*	*1.007*	0.996	*1.029*$^{\bullet\bullet}$	0.976*	0.988
HOUSEHOLD-1–294	*1.018*	*1.009*	*1.017*	*1.023*	0.981	0.970	0.975
HOUSEHOLD-1–319	*1.023*	*1.110*	*1.086*$^{\bullet}$	*1.091*	*1.040*	0.893**	0.927
HOUSEHOLD-1–373	*1.004*$^{\bullet\bullet\bullet}$	*1.004*$^{\bullet\bullet\bullet}$	*1.003*$^{\bullet\bullet\bullet}$	*1.004*$^{\bullet\bullet\bullet}$	*1.004*$^{\bullet\bullet\bullet}$	*1.004*$^{\bullet\bullet\bullet}$	*1.001*
HOUSEHOLD-1–379	0.995	*1.014*	*1.043*$^{\bullet\bullet}$	0.988	0.971*	*1.014*	*1.025*$^{\bullet}$
HOUSEHOLD-1–416	0.926	0.940	*1.027*	0.994	0.981	0.867*	0.938
HOUSEHOLD-1–478	0.986	0.987*	*1.007*	0.966	0.996	0.999	0.999
HOUSEHOLD-1–503	*1.055*	0.965	*1.008*	*1.000*	*1.004*	*1.016*	0.999
HOUSEHOLD-1–506	0.948	*1.086*	0.993	0.996	*1.173*$^{\bullet}$	0.930	0.837**
HOUSEHOLD-1–516	0.989	*1.161*$^{\bullet\bullet}$	*1.000*	*1.022*	0.927	0.859**	0.921

Table 3
Hierarchies: based on the name and first digit of the product code.

G1	G2	G3		G4	G5
FOODS_1_046	FOODS_2_013	FOODS_3_007	FOODS_3_407	HOBBIES_1_015	HOUSEHOLD_1_055
FOODS_1_054	FOODS_2_061	FOODS_3_011	FOODS_3_412	HOBBIES_1_074	HOUSEHOLD_1_072
FOODS_1_085	FOODS_2_065	FOODS_3_023	FOODS_3_458	HOBBIES_1_103	HOUSEHOLD_1_083
FOODS_1_086	FOODS_2_164	FOODS_3_042	FOODS_3_473	HOBBIES_1_254	HOUSEHOLD_1_087
FOODS_1_096	FOODS_2_171	FOODS_3_067	FOODS_3_475	HOBBIES_1_323	HOUSEHOLD_1_109
FOODS_1_172	FOODS_2_181	FOODS_3_080	FOODS_3_477	HOBBIES_1_381	HOUSEHOLD_1_118
FOODS_1_183	FOODS_2_204	FOODS_3_089	FOODS_3_495		HOUSEHOLD_1_149
FOODS_1_218	FOODS_2_293	FOODS_3_127	FOODS_3_498		HOUSEHOLD_1_197
	FOODS_2_322	FOODS_3_144	FOODS_3_547		HOUSEHOLD_1_198
	FOODS_2_371	FOODS_3_178	FOODS_3_555		HOUSEHOLD_1_235
		FOODS_3_195	FOODS_3_562		HOUSEHOLD_1_243
		FOODS_3_217	FOODS_3_580		HOUSEHOLD_1_247
		FOODS_3_226	FOODS_3_584		HOUSEHOLD_1_251
		FOODS_3_228	FOODS_3_586		HOUSEHOLD_1_294
		FOODS_3_230	FOODS_3_610		HOUSEHOLD_1_319
		FOODS_3_232	FOODS_3_654		HOUSEHOLD_1_373
		FOODS_3_234	FOODS_3_668		HOUSEHOLD_1_379
		FOODS_3_242	FOODS_3_672		HOUSEHOLD_1_416
		FOODS_3_252	FOODS_3_674		HOUSEHOLD_1_478
		FOODS_3_265	FOODS_3_694		HOUSEHOLD_1_503
		FOODS_3_304	FOODS_3_697		HOUSEHOLD_1_506
		FOODS_3_331	FOODS_3_711		HOUSEHOLD_1_516
		FOODS_3_349	FOODS_3_714		
		FOODS_3_363	FOODS_3_730		
		FOODS_3_369	FOODS_3_752		
		FOODS_3_377	FOODS_3_792		
		FOODS_3_389	FOODS_3_820		
		FOODS_3_400			

Table 4
Multivariate vs. direct hierarchical forecasts using ETS.

	1-steps	2-steps	3-steps	4-steps	5-steps	6-steps	7-steps
G1	*2.825*$^{\bullet}$	*1.186*	*2.088*$^{\bullet}$	*1.517*	0.282***	0.369***	0.406**
G2	0.897	0.986	*1.471*	*1.540*	*2.374*	*1.562*	*1.914*$^{\bullet}$
G3	*2.549*$^{\bullet\bullet\bullet}$	*2.377*$^{\bullet\bullet\bullet}$	*2.351*$^{\bullet\bullet\bullet}$	*2.614*$^{\bullet\bullet\bullet}$	*3.023*$^{\bullet\bullet\bullet}$	0.361***	0.219***
G4	0.896*	0.801***	0.918	0.885**	0.992	*1.145*$^{\bullet\bullet\bullet}$	*1.233*$^{\bullet\bullet}$
G5	*1.129*	0.933	*1.070*	*1.063*	0.814	0.944	0.929

Table 5
Multivariate vs. direct hierarchical forecasts using ARIMA.

	1-steps	2-steps	3-steps	4-steps	5-steps	6-steps	7-steps
G1	*1.523*	*1.074*	*1.900*	*1.214*	0.321***	0.389***	0.452**
G2	0.742	0.640	0.545***	0.586**	0.534**	*1.890*$^{\bullet\bullet\bullet}$	*1.743*$^{\bullet\bullet}$
G3	0.779	0.311**	0.475*	0.783	*1.716*	0.483***	0.388***
G4	0.835	0.609**	0.726	0.735	*1.141*	0.991	*1.182*$^{\bullet}$
G5	*2.472*$^{\bullet\bullet}$	0.748	0.378**	0.569	*1.091*	0.502***	0.576***

Table 6
Multivariate bottom-up vs. optimal ETS combination: performance comparisons.

	1-steps	2-steps	3-steps	4-steps	5-steps	6-steps	7-steps
G1	0.964	*1.087*	*1.089*	*1.109*	*1.347*	*1.277*	*1.436*
G2	*2.895*$^{\bullet\bullet}$	*2.660*$^{\bullet\bullet}$	*1.818*$^{\bullet}$	*1.886*$^{\bullet\bullet}$	*2.665*$^{\bullet\bullet}$	*1.275*	*1.415*
G3	*1.509*$^{\bullet\bullet\bullet}$	*1.487*$^{\bullet\bullet\bullet}$	*1.418*$^{\bullet\bullet\bullet}$	*1.485*$^{\bullet\bullet\bullet}$	*1.917*$^{\bullet\bullet\bullet}$	0.689**	0.713
G4	*2.190*$^{\bullet\bullet}$	*1.504*	*2.738*$^{\bullet\bullet\bullet}$	*1.905*	*3.298*$^{\bullet\bullet}$	*1.672*	*2.601*$^{\bullet\bullet}$
G5	0.767	0.557*	0.806	0.611	*1.417*	*1.786*$^{\bullet\bullet}$	*1.610*$^{\bullet\bullet}$

Table 7
Multivariate EWMA vs. optimal combination of ARIMA: performance comparison.

	1-steps	2-steps	3-steps	4-steps	5-steps	6-steps	7-steps
G1	0.399**	0.864	0.812	0.748	0.888	0.803$\ast$	0.958
G2	*1.390*	*1.070*	0.635**	0.688*	0.556$^{*}\ast$	0.934	0.919
G3	0.461***	0.207***	0.306***	0.482***	*1.200*	0.804	*1.019*
G4	0.915	0.885	*1.537*	*1.478*	*2.527*$^{\bullet}$	0.615**	*1.022*
G5	*1.234*	0.509	0.268***	0.452**	*1.224*	0.499***	0.699**

References

Athanasopoulos, G., Ahmed, R. A., & Hyndman, R. J. (2009). Hierarchical forecasts for Australian domestic tourism. *International Journal of Forecasting*, *25*, 146–166.

Brown, R. G. (1959). *Statistical forecasting for inventory control*. McGraw-Hill Book Company.

Durbin, J., & Koopman, S. J. (2012). *Time series analysis by state space methods*. Oxford University Press.

Eddelbuettel, D., & François, R. (2011). Rcpp: Seamless R and C++ integration. *Journal of Statistical Software*, *40*(8), 1–18.

Eddelbuettel, D., & Sanderson, C. (2014). RcppArmadillo: Accelerating R with high-performance C++ linear algebra. *Computational Statistics & Data Analysis*, *71*, 1054–1063.

Enns, P. G., Machak, J. A., Spivey, W. A., & Wrobleski, W. J. (1982). Forecasting applications of an adaptive multiple exponential smoothing model. *Management Science*, *28*(9), 1035–1044.

Granger, C., & Newbold, P. (1977). *Forecasting economic time series*. Academic Press.

Harvey, A. (1986). Analysis and generalisation of a multivariate exponential smoothing model. *Management Science*, *32*(3), 374–380.

Harvey, A. C. (1989). *Forecasting, structural time series analysis, and the Kalman Filter*. Cambridge University Press.

Harvey, A. C., & Peters, S. (1990). Estimation procedures for structural time series models. *Journal of Forecasting*, *9*(2), 89–108.

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.

Hyndman, R. J., & Athanasopoulos, G. (2021). *Forecasting: principles and practice* (3rd ed.). Melbourne, Australia: OTexts, . (Accessed on 01 July 2022).

Hyndman, R., & Khandakar, Y. (2008). Automatic time series forecasting: The forecast package for R. *Journal of Statistical Software*, *027*.

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.

Hyndman, R., Lee, A., Wang, E., & Wickramasuriya, S. (2021). Package 'HTS' R-project.org.

Jones, R. H. (1966). Exponential smoothing for multivariate time series. *Journal of the Royal Statistical Society. Series B. Statistical Methodology*, *28*(1), 241–251.

Koopman, S. J. (1993). Disturbance smoother for state space models. *Biometrika*, *11*, 7–126.

Lütkepohl, H. (1984). Forecasting contemporaneously aggregated vector ARMA processes. *Journal of Business & Economic Statistics*, *2*(3), 201–214.

Magee, J. F. (1958). *Production planning and inventory control*. McGraw-Hill Book Company.

Makridakis, S., Spiliotis, E., & Assimakopoulos, V. (2022). The M5 accuracy competition: Results, findings and conclusions. *International Journal of Forecasting*, (Available online).

Muth, J. F. (1960). Optimal properties of exponentially weighted forecasts. *Journal of the American Statistical Association*, *55*(290), 299–306.

Pennings, C. L., & van Dalen, J. (2017). Integrated hierarchical forecasting. *European Journal of Operational Research*, *263*(2), 412–418.

Poloni, F., & Sbrana, G. (2015). A note on forecasting demand using the multivariate exponential smoothing framework. *International Journal of Production Economics*, *162*, 143–150.

Poloni, F., & Sbrana, G. (2017). Multivariate trend–cycle extraction with the Hodrick–Prescott filter. *Macroecon. Dynam.*, *21*(6), 1336–1360.

R Core Team (2020). *R: A language and environment for statistical computing*. Vienna, Austria: R Foundation for Statistical Computing.

Sbrana, G., Silvestrini, A., & Venditti, F. (2017). Short-term inflation forecasting: The meta approach. *International Journal of Forecasting*, *33*(4), 1065–1081.

Shumway, R. H., & Stoffer, D. S. (1982). An approach to time series smoothing and forecasting using the EM algorithm. *Journal of Time Series Analysis*, *3*(4), 253–264.

Syntetos, A. A., Babai, Z., Boylan, J. E., Kolassa, S., & Nikolopoulos, K. (2016). Supply chain forecasting: Theory, practice, their gap and the future. *European Journal of Operational Research*, *252*(1), 1–26.

Villegas, M. A., & Pedregal, D. J. (2018). Supply chain decision support systems based on a novel hierarchical forecasting approach. *Decision Support Systems*, *114*, 29–36.

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.

Winters, P. R. (1960). Forecasting sales by exponentially weighted moving averages. *Management Science*, *6*(3), 324–342.