Schosser *Journal of Big Data* (2022) 9:26
https://doi.org/10.1186/s40537-022-00574-7

Journal of Big Data

SHORT REPORT Open Access

Tensor extrapolation: an adaptation to data sets with missing entries

Josef Schosser*

*Correspondence:
josefschosser01@gmail.com
School of Business, Economics and Information Systems, University of Passau, Passau, Germany

Abstract

Background: Contemporary data sets are frequently relational in nature. In retail, for example, data sets are more granular than traditional data, often indexing individual products, outlets, or even users, rather than aggregating them at the group level. Tensor extrapolation is used to forecast relational time series data; it combines tensor decompositions and time series extrapolation. However, previous approaches to tensor extrapolation are restricted to complete data sets. This paper adapts tensor extrapolation to situations with missing entries and examines the method's performance in terms of forecast accuracy.

Findings: To base the evaluation on time series with both diverse and controllable characteristics, the paper develops a synthetic data set closely related to the context of retailing. Calculations performed on these data demonstrate that tensor extrapolation outperforms the univariate baseline. Furthermore, a preparatory completion of the data set is not necessary. The higher the fraction of missing data, the greater the superiority of tensor extrapolation in terms of prediction error.

Conclusions: Forecasting plays a key role in the optimization of business processes and enables data-driven decision making. As such, tensor extrapolation should be part of the forecaster's toolkit: Even if large parts of the data are missing, the proposed method is able to extract meaningful, latent structure, and to use this information in prediction.

Keywords: Forecast accuracy, Missing values, Relational data, Tensor decomposition, Time series analysis

Introduction

In general, variables can be absolute or relational [1]. A person's age, education, gender, or level of environmental awareness are absolute variables. However, if a variable is defined by the relationship between entities, it is called a relational variable. Examples of relational variables are the difference in status between persons x and y, the intensity of friendships, the power of one person over another, and the existence and non-existence of trade relations between countries. Relational variables are often dynamic, i.e., they evolve over time.

Contemporary "big" data sets frequently exhibit relational variables [2, 3]. In retail, for instance, data sets are more granular than traditional data, often indexing individual products, outlets, or even users, rather than aggregating them at the group level [4, 5]. Consequently, there are different types of dependencies between the variables of interest (e.g., dependencies across products, dependencies among stores). The relational character of the data is, however, often neglected, notably in prediction tasks. Instead, univariate extrapolation approaches are applied; they cannot capture inter-series dependencies. Moreover, existing multivariate forecasting methods (e.g., vector autoregressions) are restricted to low-dimensional settings and are, hence, not suitable for practical use in large-scale forecasting problems [6, 7].

Tensor extrapolation intends to forecast relational time series data using multi-linear algebra. It proceeds as follows: Multi-way data are arranged in the form of multi-dimensional arrays, i.e., tensors. Tensor decompositions are then used to identify periodic patterns in the data. Subsequently, these patterns serve as input for time series methods. Tensor extrapolation originated in the work of Dunlavy et al. [8] and Spiegel et al. [9], but was limited to preselected time series approaches and binary data. Only recently, Schosser [10, 11] laid the foundations for applications in large-scale forecasting problems.

So far, tensor extrapolation has been restricted to complete data sets. However, values are often missing from time series data. Instances of corruption, inattention, or sensor malfunctions all require forecasting processes to be equipped to handle missing elements [12, 13]. The forecasting literature suggests two ways of dealing with missing entries in time series [6]. The first way is to use only the section of the time series following the last missing value. Obviously, this only makes sense if there is a long enough series of observations to produce meaningful forecasts. The second option is to insert estimates where values are missing. This could be done by means of (linear) interpolation or by carrying actual observations forward or backward.

The paper at hand contributes to the literature in adapting tensor extrapolation to situations with missing entries. We integrate tensor factorizations that support missing data [14] and demonstrate that data completeness is not required. Missing elements can be handled natively; the techniques mentioned above are, therefore, not needed. Our calculations show that tensor extrapolation clearly outperforms the conventional approach based on univariate extrapolation. The higher the fraction of missing data, the greater the superiority of the proposed method in terms of prediction error. Missing data thus act as an additional rationale for using tensor extrapolation, i.e., for taking the relational nature of the data into account.

The remainder of the paper progresses as follows. "Related work" section briefly reviews related work. "Proposed method" section presents the proposed method. It provides background material on tensor analysis, introduces the state of the art of tensor extrapolation, and explains our extensions. The method is applied to synthetic data in "Application" section. Results are shown in "Results and discussion" section. Finally, "Conclusions" section concludes the paper.

Related work

For an extensive overview of the forecasting literature, we refer to Hyndman and Athanasopoulos [6]. The following areas are particularly relevant to our problem: tensor autoregression, forecasting competitions, and meta-learning for time series forecasting.

Tensor autoregression

Over the last decade, tensor-based methods have received growing attention within the statistics community [15]. For time series data, autoregressive approaches are of particular interest. Hill et al. [16] introduce a matrix-on-tensor regression model for univariate time series with known seasonal periodicity. The model is used in predicting future development. Hoff [17] develops a tensor-on-tensor regression model for longitudinal relational data. The author concentrates on the estimation and interpretation of model parameters; predictions are not intended. Minhas et al. [18] apply the framework of Hoff [17] in the context of international relations. Tensor extrapolation employs a general class of state-space models and, hence, offers a richer setting compared to autoregressive approaches.

Forecasting competitions

Competitions (or "challenges", "common tasks") are ubiquitous in machine learning, statistics, and related fields [19, 20]. They lower barriers to entry by providing well-defined tasks and the resources needed to undertake them. Moreover, they create research communities with shared goals and assumptions and, maybe most important, offer proof of gradual scientific progress [21]. The so-called M competitions are the most influential and widely cited competitions in the field of time series forecasting [22]. The recent M5 competition focuses on a retail sales forecasting application [23]. The data provided consist of grouped, correlated time series, organized in a hierarchical structure. For the top and middle parts of the hierarchy, the best-performing (machine learning-based) methods offer considerable gains in accuracy over simple benchmarks. However, the results at the lower end of the hierarchy (i.e., the product-store level) are disappointing. We show the superiority of tensor extrapolation at a comparable level of detail.

Meta-learning for time series forecasting

In relevant literature, there are multiple classes of forecasting models such as exponential smoothing, Autoregressive Integrated Moving Average (ARIMA), or deep neural networks. Within individual model families, automation of model selection and tuning of forecasting algorithms has been extensively studied [e.g., 24, 25]. Meta-learning for time series forecasting is a natural evolution of these efforts. It intends to efficiently train, optimize, and choose the best-performing model among various classes of predictive models for a given data set. Recently, meta-learning platforms have been proposed by Gastinger et al. [26] and Shah et al. [13]. However, these approaches do not offer routines specifically designed for relational data. We

demonstrate that accounting for the relational character of the data pays off, and suggest tensor extrapolation as an additional base element in meta-learning frameworks.

Proposed method

This section gives background material on tensor analysis. Moreover, it demonstrates the current state of tensor extrapolation and explains our extensions.

Tensor analysis

Multi-way arrays, or *tensors*, are multi-dimensional collections of numbers. The dimensions are known as *ways*, *orders*, or *modes* of a tensor. Using this terminology, scalars, vectors, and matrices can be interpreted as zero-order, first-order, and second-order tensors, respectively. Tensors of order three and higher are called higher-order tensors [27, 28]. In the simplest high-dimensional case, the tensor can be thought of as a "data cube" [29, 30]. This case should be formalized in the following: Let $I, J, K \in \mathbb{N}$ denote index upper bounds, i.e., the number of entities in the modes of interest; a third-order tensor is denoted by $\underline{\mathbf{X}} \in \mathbb{R}^{I \times J \times K}$. Here, x_{ijk} describes the entry in the i^{th} row, j^{th} column, and k^{th} tube of $\underline{\mathbf{X}}$. The modes of $\underline{\mathbf{X}}$ are referred to as mode A, mode B, and mode C, respectively. Throughout this section, we will restrict ourselves to third-order tensors for reasons of simplicity. Nevertheless, the concepts introduced naturally extend to tensors of order four and higher. It should be noted that most of the notation and terminology used in our deliberations is borrowed from Kiers [31].

The so-called *Candecomp/Parafac* (*CP*) decomposition is one of the most common tensor factorizations. It decomposes a tensor as a set of factor matrices. The model was independently proposed by Hitchcock [32], Carroll and Chang [33], and Harshman [34]. A variant that supports missing values has been developed by Tomasi and Bro [14]. It operates only on the observed data and disregards entries that are missing in the optimization process:

$$\min_{a_{ir}, b_{jr}, c_{kr}} \sqrt{\sum_{i=1}^{I} \sum_{j=1}^{J} \sum_{k=1}^{K} w_{ijk} \cdot \left(x_{ijk} - \sum_{r=1}^{R} a_{ir} b_{jr} c_{kr} \right)^2}$$

Here, a_{ir}, b_{jr}, and c_{kr} denote elements of the component matrices $\mathbf{A}$ (for mode A), $\mathbf{B}$ (for mode B), and $\mathbf{C}$ (for mode C) of sizes $(I \times R)$, $(J \times R)$, and $(K \times R)$, respectively. The tensor mask $\underline{\mathbf{W}}$ is a Boolean array of the same shape as the original tensor $\underline{\mathbf{X}}$. Its elements w_{ijk} assume the value one if x_{ijk} is present and zero where it is not. The number of components R represents the only hyperparameter to be specified.

Tensor extrapolation: literature and extensions

Tensor extrapolation aims to forecast relational time series data. That means, relations for T time steps are given as input and the objective is to predict the relations at future

times $T+1, T+2, \ldots, T+L$. Without loss of generality, the exposition at hand is limited to the case where the snapshots of relations can be represented in the form of a matrix. Here, tensors provide a natural way to integrate the temporal dimension. Therefore, a third-order data array $\underline{\mathbf{X}}$ of size $(I \times J \times T)$ is given, with time being modeled in mode C. Extrapolation involves computing the tensor $\underline{\hat{\mathbf{X}}}$ of size $(I \times J \times L)$ that includes estimates concerning future relations. The approach relies on the use of tensor decomposition and exponential smoothing. It proceeds as follows: The multi-way data array is decomposed applying Candecomp/Parafac (CP) factorization. Every one of the component matrices provides information about the respective mode, i.e., detects latent structure in the data. For further processing, the "time" component matrix $\mathbf{C}$ of size $(T \times R)$ is transformed into a set of column vectors $\mathbf{c}_r$. These vectors record different periodic patterns, e.g., seasons or trends. The periodic patterns discovered are used as input for exponential smoothing techniques. This enables forecasts to be obtained and arranged as columns $\hat{\mathbf{c}}_r$ of the new "time" component matrix $\hat{\mathbf{C}}$ of size $(L \times R)$. Subsequently, the tensor $\underline{\hat{\mathbf{X}}}$ can be calculated; it contains estimates regarding future relations.

Originally, tensor extrapolation was introduced for binary data, meaning data indicating the presence or absence of relations of interest. Dunlavy et al. [8] use the temporal profiles computed by CP as a basis for the so-called *Holt-Winters Method*, i.e., exponential smoothing with additive trend and additive seasonality. Spiegel et al. [9] apply CP decomposition in connection with simple exponential smoothing, i.e., exponential smoothing without trend and seasonality. In both articles, the model parameters are deliberately set. Later on, Schosser [11] sets the stage for data-driven model selection and estimation in large-scale forecasting problems. First, he resorts to an automatic forecasting procedure based on a general class of state-space models subsuming all standard exponential smoothing methods [24, 35]. Model selection and optimization of both smoothing parameters and initial state variables are "individual" [36, 37], meaning that they are based on each single periodic pattern contained in a column of the matrix $\mathbf{C}$. Second, he investigates the conditions for use of the methodology with real-valued data. Thereby, the importance of data preprocessing becomes apparent. Given this preliminary work, our contribution is as follows: We integrate a CP variant that supports missing values and thus increase the practical relevance of the tensor extrapolation. Algorithm 1 displays our modification to tensor extrapolation. Numerical experiments suggest the effectiveness of the proposed method. Please note that we do not intend to complete or supplement given historical data. To fill in missing historical data, only the component matrices of the tensor factorization (i.e., $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$) have to be (re)combined [14]. Time series extrapolation procedures, on the other hand, are not required.

Algorithm 1 Tensor extrapolation for large-scale data with missing entries

Input $\underline{\mathbf{X}}$ (contains missing entries), R
Output $\underline{\hat{\mathbf{X}}}$

Preprocessing / transformation: $\underline{\mathbf{X}}_{trans} \leftarrow \underline{\mathbf{X}}$
CP decomp. (Tomasi and Bro [14]): $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C} \leftarrow \underline{\mathbf{X}}_{trans}$, R
Separate column vectors: $\mathbf{c}_1$, $\mathbf{c}_2$, ..., $\mathbf{c}_R \leftarrow \mathbf{C}$
for each $r \in 1, \ldots, R$ **do**
    Automatic exponential smoothing: $\hat{\mathbf{c}}_r \leftarrow \mathbf{c}_r$
end for
Merge column vectors: $\hat{\mathbf{C}} \leftarrow \hat{\mathbf{c}}_1$, $\hat{\mathbf{c}}_2$, ..., $\hat{\mathbf{c}}_R$
Calculate tensor: $\underline{\hat{\mathbf{X}}}_{trans} \leftarrow \mathbf{A}$, $\mathbf{B}$, $\hat{\mathbf{C}}$
Back-transformation: $\underline{\hat{\mathbf{X}}} \leftarrow \underline{\hat{\mathbf{X}}}_{trans}$

Application

It is necessary to base our evaluation on time series with both diverse and controllable characteristics [38]. In particular, the impact of different sized proportions of missing data needs to be investigated. This can be done in two ways [14]. In the first, the position of missing entries in a real data set is simulated. In the second, a fully synthetic data set is developed. We choose the latter, more consistent [39], route and generate an exhaustive synthetic data set closely related to the context of retailing. Relational data occur, for instance, in companies that maintain relationships with a large number of business partners or users and offer a variety of products or services. Therefore, our simulated data should be interpreted as sales volume depending on user, product, and time. We assume that the data can be modeled by $R = 4$ components and aim for a tensor of size $(I \times J \times T) = (160 \times 120 \times 100)$. Doing so, we obtain in total 19,200 time series with 100 observations each.

In developing our synthetic data set, we closely follow the procedure introduced by Dunlavy et al. [8] and adapted by Schosser [11]. Except for the scaling (and, of course, missing values), the resulting data correspond to those used by Schosser [11]. First of all, the component matrices are generated. Thereby, we are not subject to specific limitations. In particular, the CP model does not place orthogonality constraints on the component matrices [8, 29]. The matrices $\mathbf{A}$ and $\mathbf{B}$ of size $(I \times R)$ and $(J \times R)$, respectively, are "entity participation" matrices. In other words, column $\mathbf{a}_r$ (resp. $\mathbf{b}_r$) is the vector of participation levels of all the entities in component r. User participation is shown in matrix $\mathbf{A}$: Here, the users are supposed to react in a different way to a specific product-time combination. The extent of this reaction is measured according to the density

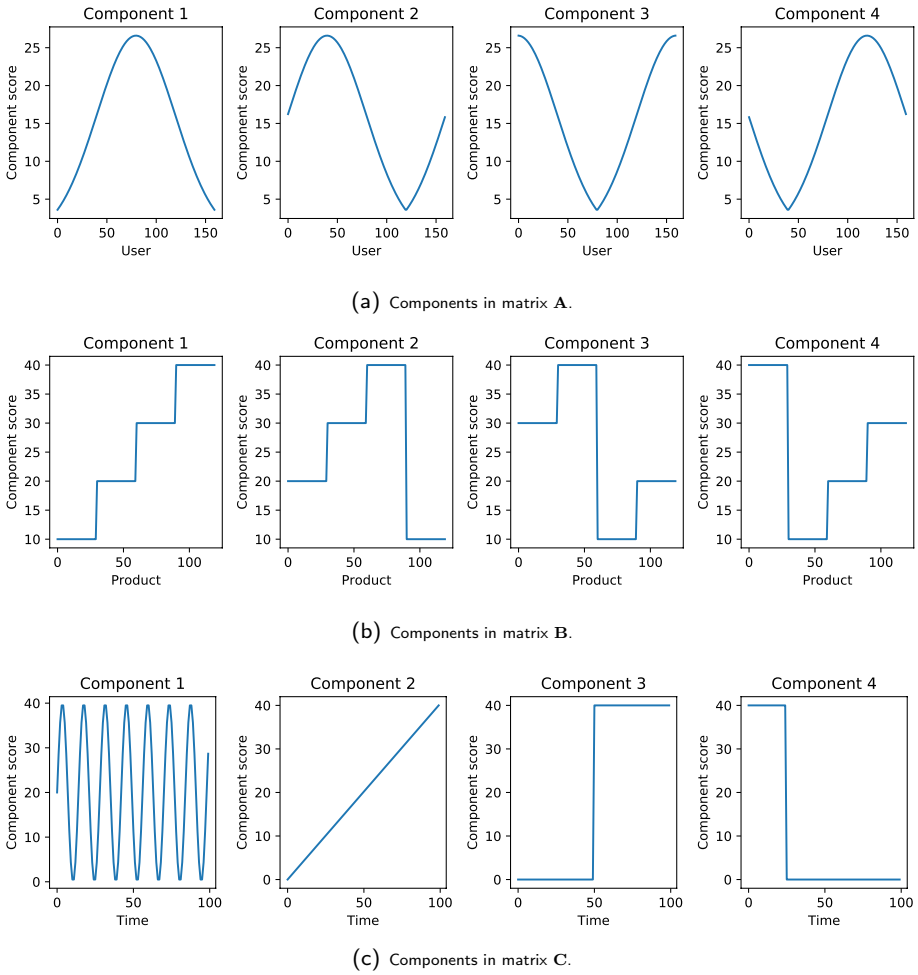

Fig. 1 Generation of synthetic data

function of a Gaussian distribution (cf. Fig. 1a). Product participation is demonstrated in matrix **B**: There are groups of products that respond similarly to the same time and user effect combination (cf. Fig. 1b). The columns of matrix **C** of size $(T \times R)$ record different periodic patterns. We create a sinusoidal pattern, a trend, a structural break, and another break (cf. Fig. 1c). By aggregating the matrices **A**, **B**, and **C**, a noise-free version of the tensor emerges. Finally, we add Gaussian noise to every entry. The standard deviation of the error term is assumed to equal half of the mean of the noise-free tensor entries.

To investigate the effect of missing entries, we eliminate different sized fractions of the data. In the words of Rubin [40], the elements are *missing completely at random*. That means there is no relationship between whether an entry is missing and any other entry in the data set, missing or observed. For instance, where the desired level of missingness equals 20%, each element has a probability of 0.2 of being missing. Our implementation uses consecutive draws of a Bernoulli random variable with probability of success equal

to the intended level of missingness [41]. Due to the large number of entries, the realized amount of missing values differs only slightly from the level intended. As a consequence of our procedure, the missing elements are randomly scattered across the array without any specific pattern. Please note that we do not investigate systematically missing entries (the subject-based literature uses the somewhat confusing terms *missing at random* and *missing not at random*; [40]). The CP factorization proposed by Tomasi and Bro [14] gets along with systematic missingness. For techniques of preparatory completion, this only applies to a limited extent or not at all. A meaningful comparison is, therefore, not possible.

For each fraction of missing values, we split the data into an estimation sample and a hold-out sample. The latter consists of the 20 most recent observations. We thus obtain $\underline{\mathbf{X}}^{est}$ of size $(160 \times 120 \times 80)$ and $\underline{\mathbf{X}}^{hold}$ of size $(160 \times 120 \times 20)$, respectively. The methods under consideration are implemented, or trained, on the estimation sample. The forecasts are produced for the whole of the hold-out sample and arranged as tensor $\hat{\underline{\mathbf{X}}}$ of size $(160 \times 120 \times 20)$. Finally, forecasts are compared to the actual withheld observations.

We should be aware of the fact that the included time series are differently scaled. This has two consequences. First, since CP minimizes squared error, differences in scale may lead to distortions. Therefore, data preprocessing is necessary [11]. We choose a simple centering across the time mode [31]. It is carried out by averaging the observations over the (available) elements of the respective time series, i.e., across mode *C*, and then subtracting each thus obtained average from all the observations that partake in it. Formally, the preprocessing step (please compare Line 1 in Algorithm 1) implies

$$x_{ijt,trans} = x_{ijt} - \bar{x}_{ij\cdot},$$

where the subscript dot is used to indicate the mean across $t \in 1, \ldots, T$. During back-transformation (please compare Line 9 in Algorithm 1), the averages previously deducted are added back. This involves

$$\hat{x}_{ijt} = \hat{x}_{ijt,trans} + \bar{x}_{ij\cdot}$$

for $t \in T+1, \ldots, T+L$. Second, only scale-free performance measures can be employed. We use the *Mean Absolute Percentage Error* (*MAPE*) and the *Symmetric Mean Absolute Percentage Error* (*sMAPE*) [11, 42].

Following Schosser [11], our application of tensor extrapolation connects a state-of-the-art tensor decomposition with an automatic forecasting procedure based on a general class of state-space time series models. For this purpose, we use the programming language `Python`. The function `parafac` (library `TensorLy`; [43]) supports the factorization proposed by Tomasi and Bro [14] and, hence, allows for missing values. When `parafac` is called, the number of components *R* must be specified. The function `ETSModel` (library `statsmodels`; [44]) offers a `Python`-based implementation of the automatic exponential smoothing algorithm developed by Hyndman et al. [35] and Hyndman and Khandakar [24]. This algorithm provides a rich set of possible models and

Table 1 Forecasting accuracy based on synthetic data in terms of MAPE and sMAPE

Method	Fraction of missing values						
	-	5%	10%	20%	30%	40%	50%
	MAPE						
Baseline							
`ETSModel`	229.216	229.472	231.271	237.175	238.353	243.600	244.766
Tensor extr.							
$R = 3$	185.401	185.852	185.971	186.892	185.449	187.114	186.936
$R = 4$	179.852	180.387	180.647	181.473	179.836	187.041	184.915
$R = 5$	181.227	180.432	182.375	183.499	179.717	181.763	180.465
$R = 6$	181.171	180.361	181.318	182.286	179.604	184.152	180.426
$R = 7$	180.548	182.183	181.131	181.483	179.523	182.712	180.630
Ensemble							
$R = 4 - 6$	180.738	180.424	182.027	182.587	180.870	183.901	181.857
$R = 3 - 7$	181.573	181.813	182.851	182.585	182.167	182.966	183.291
	sMAPE						
Baseline							
`ETSModel`	45.356	45.894	46.526	48.002	49.953	52.029	53.910
Tensor extr.							
$R = 3$	46.985	47.050	47.058	47.073	47.138	47.146	47.660
$R = 4$	46.849	46.900	46.914	46.945	47.144	47.153	47.644
$R = 5$	46.789	46.902	46.823	46.850	47.014	47.041	47.633
$R = 6$	46.794	46.912	46.897	46.935	47.032	46.919	46.636
$R = 7$	46.812	46.814	46.907	49.945	46.996	47.029	47.609
Ensemble							
$R = 4 - 6$	46.796	46.901	46.854	46.900	46.968	46.955	47.548
$R = 3 - 7$	46.761	46.825	46.828	46.917	46.930	46.965	47.500

The estimation sample includes the first 80 time steps, and the hold-out sample covers the next 20 time steps. Different sized fractions of the data are eliminated; the respective elements are missing completely at random. The results obtained from tensor extrapolation are grouped with respect to the selected hyperparameter, the number of components R. Ensemble contains the scores of (equally weighted) combinations of forecasts associated with different specifications of R

has been shown to perform well in comparisons with other forecasting methods [23, 26, 45, 46]. Further, in relation to more complex forecasting techniques, the requirements in terms of data availability and computational resources are fairly low [7, 11]. As a baseline, we use a univariate, i.e., per-series, extrapolation by means of `ETSModel`. Here, the missing values must be filled in. To this end, we propagate the last valid observation forward. Any remaining gaps are filled in backwards.

Results and discussion

Table 1 displays our results. With regard to small amounts of missing data, the situation is as follows: As measured by MAPE, tensor extrapolation outperforms the baseline. If, for instance, 5% of the data are missing, up to 21.40% of the prediction error can be reduced. On the basis of sMAPE, no clear ranking can be determined. As the level of missing data increases, tensor extrapolation becomes even more attractive. In the case of MAPE, the distance to univariate extrapolation increases. Where half of the data are

missing, up to 26.28% of the prediction error can be reduced. In terms of sMAPE, our proposed method now also dominates. Our results are largely unaffected by the hyper-parameter choice, i.e., the number of components R. One way to circumvent the problem of committing to a specific number of components is to use an (equally weighted) combination or ensemble of forecasts. Here, again, the results are encouraging, as is often the case with the combination of forecasts [47]. Moreover, the signal-to-noise ratio does not influence the hierarchy described. Detailed results on this are available upon request.

Using the `Python` library `timeit`, we quantify the computational burden associated with the methods in question. The measurements refer to a commodity notebook with Intel Core i5-6300 CPU 2x2.40 GHz and 8 GB RAM. By way of example, we assume 20% of the data to be missing. Given 100 executions, the average runtime of tensor extrapolation with $R = 4$ components equals 21.28 s. The baseline, `ETSModel`, takes on average 193.16 s. The reason for this difference lies in the computational cost associated with the automatic exponential smoothing algorithm, i.e., the selection and estimation of an adequate exponential smoothing method. Tensor extrapolation requires tensor decomposition, but significantly reduces the dimension of the forecasting problem. The automatic exponential smoothing algorithm is applied to $R = 4$ time series. In contrast, 19,200 function calls are necessary for the baseline approach. Regardless of computational resources, tensor extrapolation should be computationally cheaper even with the combination of forecasts.

Conclusions

In spite of the possibilities arising from the "big data revolution", the relational character of many time series is largely neglected in forecasting tasks. Recently, tensor extrapolation has been shown to be effective in forecasting large-scale relational data [11]. However, the results so far are limited to complete data sets. The paper at hand adapts tensor extrapolation to situations with missing entries. The results demonstrate that the method can be successfully applied for up to 50% missing values. Notwithstanding the missing elements, tensor extrapolation is able to extract meaningful, latent structure in the data and to use this information for prediction. A preparatory completion of the data set (e.g., by replacing missing elements) is not required. Given the importance of missing values in practice [48], the findings of this paper provide a compelling argument in favor of tensor extrapolation.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s40537-022-00574-7.

Additional file 1. Python code. The notebook highlights core components of the code applied in the study.

Authors' contributions

JS is the sole author. The author read and approved the final manuscript.

Availability of data and materials

All data generated or analyzed during this study are available in Additional file 1.

Received: 7 September 2021 Accepted: 6 February 2022
Published online: 25 February 2022

References

1. Wasserman S, Faust K. Social network analysis: Methods and applications. Cambridge: Cambridge University Press; 1994.
2. Kitchin R. Big Data, new epistemologies and paradigm shifts. Big Data Soc. 2014;1(1):1–12.
3. Kitchin R, McArdle G. What makes Big Data, Big Data? Exploring the ontological characteristics of 26 datasets. Big Data Soc. 2016;3(1):1–10.
4. Müller O, Junglas I, vom Brocke J, Debortoli S. Utilizing big data analytics for information systems research: Challenges, promises and guidelines. Eur J Inform Syst. 2016;25(4):289–302.
5. Fildes R, Ma S, Kolassa S. Retail forecasting: research and practice. Int J Forecasting. 2022. https://doi.org/10.1016/j.ijforecast.2019.06.004.
6. Hyndman RJ, Athanasopoulos G. Forecasting: Principles and Practice. Melbourne: OTexts; 2018.
7. De Stefani J, Bontempi G. Factor-based framework for multivariate and multi-step-ahead forecasting of large scale time series. Front Big Data. 2021;4(1):e690267.
8. Dunlavy DM, Kolda TG, Acar E. Temporal link prediction using matrix and tensor factorizations. ACM T Knowl Discov D. 2011;5(2):e10.
9. Spiegel S, Clausen J, Albayrak S, Kunegis J. Link prediction on evolving data using tensor factorization. In: New frontiers in applied data mining: PAKDD 2011 International Workshops. Springer; 2012. p. 100–110.
10. Schosser J. Multivariate extrapolation: A tensor-based approach. In: Neufeld JS, Buscher U, Lasch R, Möst D, Schönberger J, editors. Operations Research Proceedings 2019. New York: Springer; 2020. p. 53–9.
11. Schosser J. Tensor extrapolation: forecasting large-scale relational data. J Oper Res Soc. 2022. https://doi.org/10.1080/01605682.2021.1892460.
12. Alexandrov A, Benidis K, Bohlke-Schneider M, Flunkert V, Gasthaus J, Januschowski T, et al. `GluonTS`: Probabilistic time series models in `Python`. arXiv:1906.05264; 2019.
13. Shah SY, Patel D, Vu L, Dang XH, Chen B, Kirchner P, et al. AutoAI-TS: AutoAI for time series forecasting. In: Proceedings of the 2021 International Conference on Management of Data (SIGMOD). ACM; 2021. p. 2584–96.
14. Tomasi G, Bro R. PARAFAC and missing values. Chemometr Intell Lab. 2005;75(2):163–80.
15. Bi X, Tang X, Yuan Y, Zhang Y, Qu A. Tensors in statistics. Annu Rev Stat Appl. 2021;8(1):345–68.
16. Hill C, Li J, Schneider M. The tensor auto-regressive model. J Forecasting. 2021;40(4):636–52.
17. Hoff PD. Multilinear tensor regression for longitudinal relational data. Ann Appl Stat. 2015;9(3):1169–93.
18. Minhas S, Hoff PD, Ward MD. A new approach to analyzing coevolving longitudinal networks in international relations. J Peace Res. 2016;53(3):491–505.
19. Feuerverger A, He Y, Khatri S. Statistical significance of the Netflix challenge. Stat Sci. 2012;27(2):202–31.
20. Donoho D. 50 years of data science. J Comput Graph Stat. 2017;26(4):745–66.
21. Liberman M. Obituary: Fred Jelinek. Comput Linguist. 2010;36(4):595–9.
22. Makridakis S, Spiliotis E, Assimakopoulos V. The M4 competition: results, findings, conclusion and way forward. Int J Forecasting. 2018;34(4):802–8.
23. Makridakis S, Spiliotis E, Assimakopoulos V. The M5 accuracy competition: results, findings and conclusions. Int J Forecasting. 2022. https://doi.org/10.1016/j.ijforecast.2021.11.013.
24. Hyndman RJ, Khandakar Y. Automatic time series forecasting: The `forecast` package for `R`. J Stat Softw. 2008;27(3):1–22.
25. Salinas D, Flunkert V, Gasthaus J, Januschowski T. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. Int J Forecasting. 2020;36(3):1181–91.
26. Gastinger J, Nicolas S, Stepić D, Schmidt M, Schülke A. A study on ensemble learning for time series forecasting and the need for meta-learning. arXiv:2104.11475; 2021.
27. Cichocki A, Zdunek R, Phan AH, Amari S. Nonnegative matrix and tensor factorizations: Applications to exploratory multiway data analysis and blind source separation. Chichester: Wiley; 2009.
28. Kolda TG, Bader BW. Tensor decompositions and applications. SIAM Rev. 2009;51(3):455–500.
29. Papalexakis EE, Faloutsos C, Sidiropoulos ND. Tensors for data mining and data fusion: Models, applications, and scalable algorithms. ACM T Intel Syst Tec. 2016;8(2):e16.
30. Rabanser S, Shchur O, Günnemann S. Introduction to tensor decompositions and their applications in machine learning. arXiv:1711.10781; 2017.
31. Kiers HAL. Towards a standardized notation and terminology in multiway analysis. J Chemometr. 2000;14(3):105–22.
32. Hitchcock FL. The expression of a tensor or polyadic as a sum of products. J Math Phys. 1927;6(1):164–89.
33. Carroll JD, Chang JJ. Analysis of individual preferences in multidimensional scaling via an N-way generalization of 'Eckart-Young' decomposition. Psychometrika. 1970;35(3):283–319.
34. Harshman RA. Foundations of the PARAFAC procedure: Models and conditions for an 'explanatory' multimodal factor analysis. UCLA Working Papers Phonetics. 1970;16:1–84.
35. Hyndman RJ, Koehler AB, Snyder RD, Grose S. A state space framework for automatic forecasting using exponential smoothing. Int J Forecasting. 2002;18(3):439–54.
36. Fildes R. Evaluation of aggregate and individual forecast method selection rules. Manage Sci. 1989;35(9):1056–65.
37. Petropoulos F, Makridakis S, Assimakopoulos V, Nikolopoulos K. 'Horses for Courses' in demand forecasting. Eur J Oper Res. 2014;237(1):152–63.
38. Kang Y, Hyndman RJ, Li F. GRATIS: GeneRAting TIme Series with diverse and controllable characteristics. Stat Anal Data Min. 2020;13(4):354–76.
39. Schouten RM, Lugtig P, Vink G. Generating missing values for simulation purposes: A multivariate amputation procedure. J Stat Comput Sim. 2018;88(15):2909–30.
40. Rubin DB. Inference and missing data. Biometrika. 1976;63(3):581–92.
41. Little RJA, Rubin DB. Statistical analysis with missing data. Hoboken: Wiley; 2002.
42. Hyndman RJ, Koehler AB. Another look at measures of forecast accuracy. Int J Forecasting. 2006;22(4):679–88.
43. Kossaifi J, Panagakis Y, Anandkumar A, Pantic M. `TensorLy`: Tensor learning in `Python`. J Mach Learn Res. 2019;20(26):1–6.

44. Seabold S, Perktold J. `Statsmodels`: Econometric and statistical modeling with `Python`. In: Proceedings of the 9th Python in Science Conference (SCIPY2010); 2010. p. 57–61
45. Januschowski T, Gasthaus J, Wang Y, Salinas D, Flunkert V, Bohlke-Schneider M, et al. Criteria for classifying forecasting methods. Int J Forecasting. 2020;36(1):167–77.
46. Karlsson Rosenblad A. Accuracy of automatic forecasting methods for univariate time series data: A case study predicting the results of the 2018 Swedish general election using decades-long series. Commun Stat Case Stud. 2021;7(3):475–93.
47. Bates JM, Granger CW. The combination of forecasts. J Oper Res Soc. 1969;20(4):451–68.
48. Emmanuel T, Maupong T, Mpoeleng D, Semong T, Mphago B, Tabona O. A survey on missing data in machine learning. J Big Data. 2021;8(1): e140.

Publisher's Note