HTF: An Effective Algorithm for Time Series to Recover Missing Blocks

Haijun Zhang(✉), Hong Gao(✉), and Dailiang Jin

Harbin Institute of Technology, Harbin 150001, Heilongjiang, China
1160300327@stu.hit.edu.cn, {honggao,jdl}@hit.edu.cn

Abstract. With the popularity of time series analysis, failure during data recording, transmission, and storage makes missing blocks in time series a problem to be solved. Therefore, it is of great significance to study effective methods to recover missing blocks in time series for better analysis and mining. In this paper, we focus on the situation of continuous missing blocks in multivariate time series. Aiming at the blackout missing block pattern, we propose a method called hankelized tensor factorization (HTF), based on singular spectrum analysis (SSA). After the hankelization of the time series, this method decomposes the intermediate result into the product of time-evolving embedding, time delaying embedding, and hidden variables embedding of multivariate variables in the low-dimensional space, to learn the essence of time series. In an experimental benchmark containing 5 data sets, the recovery effect of HTF and other baseline methods in three missing block patterns are compared to evaluate the performance of HTF. Results show that when the missing block pattern is blackout, the HTF method achieves the best recovery effect, and it can also have good results for other missing patterns.

Keywords: Multivariate time series · Missing block pattern · Missing value recovery · Tensor factorization

1 Introduction

With the rapid development of 5G, big data, and the internet of things, time series data from various sensors, financial markets, meteorological centers, industry monitoring system and the internet is growing at an unprecedented rate. People expect to exploit the huge value that can reveal the development trends in the field of interest behind the data, making the analysis, mining and forecasting of time series becoming a popular topic. Unfortunately, all of this requires time series Completeness as a prerequisite that data in real scenarios often lacks due to network failure, storage equipment malfunction and other situations from time to time. Data missing in a period of time, that is, the existence of missing

The National Key Research and Development Program of China (Grant No. 2019YFB2101902), Joint Funds of the National Natural Science Foundation of China under Grant No. U19A2059.

C. S. Jensen et al. (Eds.): DASFAA 2021 Workshops, LNCS 12680, pp. 29–44, 2021.
https://doi.org/10.1007/978-3-030-73216-5_3

blocks in time series has already been a common and urgent problem waiting to be solved.

Data Quality is a very popular topic recently, and there are a lot of good study result, i.e. [4]. However, when facing the data quality problem with time series, things are different.

In the course of practice, people gradually reach a consensus that traditional statistical method (such as interpolation, multiple imputation [15], etc.) to fill in missing blocks will obscure the hidden pattern of the original data, destroying the dynamic trend of data, followed by the recovery result not conducive to subsequent analysis and prediction [1]. As a result, new methods are gradually emerging.

In view of positions where missing blocks appear, Khayati et al. [9] put forward three kinds of missing patterns: disjoint, overlap and blackout, and then a detailed comparison of the previous work in these three patterns was presented. Their work pointed out that the recovery effect of the same method on the same time series in different missing patterns can have a huge difference, which previous study did not pay attention to. As a result of their research, they suggested the study of missing block recovery in time series data should focus on the performance in different missing patterns, especially in the case of blackout where there is a lack of good methods.

Based on the work of Khayati et al. [9], in particular, there is a gap in the topic of missing block recovery in the blackout missing pattern. An effective algorithm named hankelized tensor factorization (HTF) is proposed, to solve the situation that existing works usually have a bad performance in the case of blackout. The Inspiration of HTF is from the singular spectrum analysis (SSA), a powerful time series analysis technique, decomposes the time data to the weighted sum of a series of independent and explainable components, so as to reconstruct the original sequence well. However, singular spectrum analysis is not suitable for the case of large missing blocks. This paper drawing on the idea of singular spectrum analysis, decomposes the sequence into the product of time-evolving embedding, time delaying embedding and the hidden variables embedding of multi-dimensions in the low-dimensional space. And then a reconstruction of the original sequence is displayed, filling missing values by their reconstruction estimation. In order to obtain the recovery effect of the method, we conducted experiments under a benchmark of 5 datasets, evaluating the performance of HTF and other baseline methods. In conclusion, the contribution of this paper are listed below:

- HTF, an effective algorithm to recover the missing blocks in time series is proposed. With inspiration from SSA, we solve the problem of missing block recovery by an approach named tensor factorization that decomposes high-dimensional data to low-dimensional embeddings to learning high order temporal correlations among the sequential data.
- Based on the work of Khayati et al., the recovery effect of HTF is compared with other baseline methods in three different missing patterns on the

benchmark of 5 datasets inherited from their study, including deep learning methods which was not contained before.

2 Related Work

Study of recovery effect under different missing blocks patterns has not been widely paid attention to. The work of Simeng Wu et al. [17] is most similar to this paper, they proposed the HKMF method which focuses on how to fill missing values of blackout in time series, but their work only suits the case that the time series is univariate. One contribution of this paper can be regarded as extending their work to a multivariate case by another technique. More importantly, we find the theoretical foundation that our work may be the best result we can achieve when the time series satisfies linear recursive formula by explaining the fact that HTF and singular spectrum analysis is the same thing in some sense.

Apart from the work mentioned above, Li et al. [10] suggested Dynammo, a method based on the linear dynamic system which makes use of kalman filter to model the time series with the assumption that the observed time sequence is generated by the linear evolution of hidden variables. Kevin et al. [16] recommended a sequence matching method using the most similar subsequence to generate an imputation, by selecting k complete sequence most similar to the sequence just before the missing blocks. Haiang-Fu Yu et al. [18] put forward a matrix factorization method named TRMF to handle the case when the dimension of time series is high. The main contribution of their work is a new kind of regularization that can be explained as a constraint over a temporal graph. Wei Cao et al. [3] proposed a new deep learning model based on the original bidirectional LTSM, modifying the structure to make it more adaptive to the characteristic of time series. A new loss function that minimizes the error both in forward imputation and backward imputation was presented at the same time.

The following sections are arranged as below: Sect. 3 introduces the problem definition and missing block patterns. Section 4 introduces the HTF method and its relation to SSA. At last, in Sect. 5, we evaluate the recovery effect and performance of our method in the benchmark.

3 Backgroud

As a type of data, time series can have various auxiliary data associated with it, and timestamp is the most common among them. In this paper, we do not discuss the situation this additional information are contained. Only considering observed multivariate time series sampled evenly can help us simplify the problem and find the essence of this type of data.

3.1 Definition

The general problem of missing block recovery for time series is defined as follows:

Given: A time series of length T, $\boldsymbol{X} = \langle \boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_T \rangle$, where $\boldsymbol{x}_t \in \mathbb{R}^D$ is a D-dimensional column vector for data at time t; an indicating matrix $\boldsymbol{W} = \{0, 1\}^{D \times T}$, with $\boldsymbol{W}_{i,t} = 0$ if the i-th dimension of $\boldsymbol{x}_t$ is missing, and $\boldsymbol{W}_{i,t} = 1$ if the data is present.

Solve: Estimate the missing values in $\boldsymbol{X}$ indicated by $\boldsymbol{W}$.

We say $\boldsymbol{W}_{i,j}.\boldsymbol{W}_{i,j+1}, ..., \boldsymbol{W}_{j+l-1}$ is a missing block of length l, if $\boldsymbol{W}_{i,j}.\boldsymbol{W}_{i,j+1}, ..., \boldsymbol{W}_{j+l-1} = 0$. If $\boldsymbol{W}_{1,j}, \boldsymbol{W}_{2,j}, ..., \boldsymbol{W}_{D,j} = 0$, we state there is a blackout in the j-th column. The main topic this paper study is when the length $l \geq 20$. The missing pattern of data is defined by the indicating matrix $\boldsymbol{W}$, we notate the max length over all missing blocks as l_M, and h_M is the notation of the max height over all the columns of missing blocks that $\boldsymbol{W}_{i,j}, \boldsymbol{W}_{i+1,j}, ..., \boldsymbol{W}_{i+h-1,j} = 0$.

3.2 Patterns of Missing Blocks

The pattern of missing blocks is determined by factors like the relative position, the length and the height of missing blocks, and so on. Khayati et al. argued the following three possible pattern types of missing, as shown in Fig. 1, X_1, X_2, X_3 are the univariate time series of same length column by column, the missing blocks in every sequence is marked as grey:

1) Disjoint: In this case, missing blocks do not intersect on each other in the same time span. That is to say for each missing block, the sequences from other variables at the same time is complete, equivalent to $h_M = 1$. It is the simplest case in all three types.
2) Overlap: In this case, missing blocks intersect each other partly, and there would be at most $D - 1$ variables missing at the same time, equivalent to $2 \leq h_M < D$. It is a relatively complex case.
3) Blackout: In this case, missing blocks coincide with each other at some time interval, equivalent to $h_M = D$. Generally speaking, this case is very hard to recover when the missing length is large. So in this paper, we constrain $20 \leq l_M \leq 100$. It is the most challenging case.

In the real scenarios, these three patterns of missing seldom appear alone, a problem that all three types mix together usually happens when data transmission in a bad network.

4 Hankelized Tensor Factorization

The Hankelized Tensor Factorization algorithm we proposed in this paper, mainly focuses on the Blackout missing pattern. In this case, the indicating matrix $\boldsymbol{W}$ contains some continuous columns all filled with zero, like $\boldsymbol{W}_{:,j}, \boldsymbol{W}_{:,j+1}, ..., \boldsymbol{W}_{:,j+l-1} = \boldsymbol{0}$.

The details of HTF are listed below. First, apply hankelization on the input time series matrix $\boldsymbol{X} \in \mathbb{R}^{D \times T}$ carrying the missing position. Then, a novel tensor factorization approach inspired by gradient descent will be carried out on the

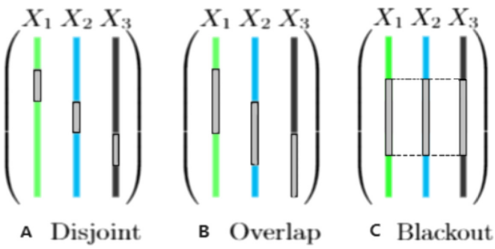


Fig. 1. Different types of missing block pattern (the missing marked as grey)

result of the first step, causing the high dimensional data with missing transformed to some low dimensional embeddings. Thirdly, a reconstruction based on the embedding obtained just before will be multiplied to make a reconstruction of the hankelized time series with missing imputed. In the end, we will do an inverse version of hankelization to turn the reconstruction to an estimation matrix, with each missing value in the original data matrix being imputed.

4.1 Methodology

In the following subsections, the detail of the procedures listed above will be explained. For the simplicity of mathematical expression, we give the following arithmetic definition:

Definition 1. $\boldsymbol{A}, \boldsymbol{B}$ *are matrices of size* $m \times n$,
$(\boldsymbol{A} \odot \boldsymbol{B})_{ij} \triangleq \boldsymbol{A}_{ij} \times \boldsymbol{B}_{ij}$

Definition 2. $\boldsymbol{x}, \boldsymbol{y}$ *are vectors of length* m *and* n,

$$\boldsymbol{x} \otimes \boldsymbol{y} \triangleq \begin{bmatrix} x_1y_1 & x_1y_2 & \dots & x_1y_n \\ x_2y_1 & x_2y_2 & \dots & x_2y_n \\ \cdot & \cdot & \dots & \cdot \\ x_my_1 & x_my_2 & \dots & x_my_n \end{bmatrix}$$

A. Tensor Hankelization
The first step of HTF is to hankelize the input time series matrix $\boldsymbol{X}$. Compared to the way Hassani et al. [7] proposed that does hankelization for the sequence $\boldsymbol{X}_{i,:}$ of each variable i, then merge to a compound matrix by put all the results in the row order or in the column order, generating a much larger matrix compared with the original $\boldsymbol{X}$. We give a way that makes the matrix to a tensor, just like what Hassani et al. do when it hankelizes the univariate sequence as defined. The difference between these two ways is shown in Fig. 2. We believe the result tensor can be represented by more explainable, effective and independent components in our approach, so the essential can be learned better and the reconstruction result has a better recovery effect. The comparison experiments in Sect. 5.2.C support our point of view.

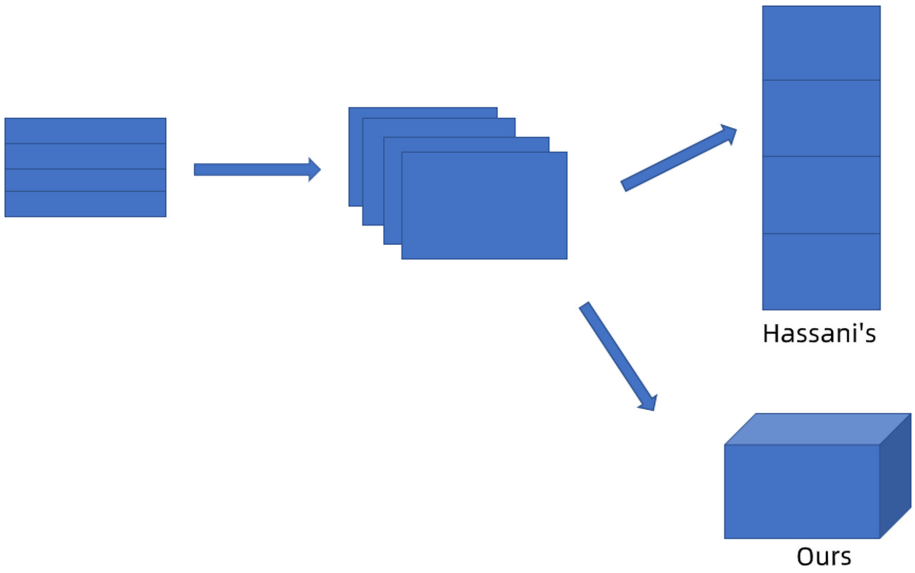


Fig. 2. Different ways of hankelization (Hassani's and ours)

Tensor hankelization can be thought of as doing the process of formula (1) for the sequence vector $\boldsymbol{X}_{i,:}$ of each variable in time series matrix $\boldsymbol{X}$, and then put the result in the third dimension *height*, vertical to the dimension of time-evolving (row) and multi-variables (column). We notate $z_i, z_{i+1}, ..., z_{i+L-1}$ as time evolving vector list of length L, and $z_j, z_{j+1}, ..., z_{j+K-1}$ as time delaying vector list of length K. The value of K must satisfy $K \geq l_M + 1$, which l_M has been defined in the Sect. 3.1. For the recovery effect of our method, the value of K should be appropriately large. The result of this step can be seen as a map from all matrix elements to all of the tensor, which is shown in Eq. (2).

$$H(\boldsymbol{z}) = H([\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_3]) = \begin{bmatrix} z_1 & z_2 & \cdots & z_L \\ z_2 & z_3 & \cdots & z_{L+1} \\ \cdot & \cdot & \cdots & \cdot \\ z_K & z_{K+1} & \cdots & z_T \end{bmatrix} \tag{1}$$

$$H_K(\boldsymbol{X})_{i,j,k} = \boldsymbol{X}_{i,j+k-1}, H_K(\boldsymbol{X}) \in \mathbb{R}^{D \times K \times L} \tag{2}$$

Tensor Factorization

The next step of HTF is to decompose the tensor coming from the tensor hankelization. By decomposing the result to time-evolving embedding, time delaying embedding and multi-variate embedding, the essential of time series can be represented independently and explainably. Another important advantage of factorization is that the embeddings which map the components of time series to low dimensional space are dense despite the original in the high dimension is sparse, the principle behind similar to that the product $\boldsymbol{Z} = \boldsymbol{x}\boldsymbol{y}^T$ of vector $\boldsymbol{x}$,

$\boldsymbol{y}$ of length m and n generates a matrix of $m \times n$ elements, while solving $\boldsymbol{x}, \boldsymbol{y}$ does not need as many as $m \times n$ elements of $\boldsymbol{Z}$. This paper takes the following formula to do tensor factorization,

$$H_K(\boldsymbol{X}) = Y \approx \sum_{r=1}^{R} \boldsymbol{A}_{:,r} \otimes \boldsymbol{B}_{:,r} \otimes \boldsymbol{C}_{:,r} \tag{3}$$

which $\boldsymbol{A}, \boldsymbol{B}$ and $\boldsymbol{C}$ represent the time-evolving trend, time delaying trend and the hidden variables of multi-variables in the low R-dimension space.

In order to solve $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$, we select the method based on the stochastic gradient descent from machine learning, compared with other approaches to solve tensor factorization problem, such as alternating least squares [8]. The reason is that other methods may be harmful to the result because of their default initialization of zero when dealing with missing values which is no difference between a tensor with no missing values but is filled with zero and the tensor which is missing at the same position and initialized with zero. We propose the following function:

$$\mathcal{L}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}(\boldsymbol{Y})} = \sum_{(i,j,k)\in\Omega} (\boldsymbol{Y}_{i,j,k} - \sum_{r=1}^{R} \boldsymbol{A}_{i,r}\boldsymbol{B}_{j,r}\boldsymbol{C}_{k,r})^2 + \lambda_s R_T(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}) + \lambda_r R(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}) \tag{4}$$

where Ω is the set of indexes correspond to the observed elements in $\boldsymbol{Y}$, which is generated from $\boldsymbol{W}$. $\boldsymbol{Y}_{i,j,k}$ is the (i,j,k)th element of $\boldsymbol{Y}$. $R_T(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}), R(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C})$ are two regularizers defined in Eq. (6) and (7), respectively, with λ_r, λ_s being the coefficients. Given Eq. (4), the task of learning $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ is achieved by solving:

$$\langle \boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C} \rangle = argmin_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}} \mathcal{L}_{\boldsymbol{A},\boldsymbol{B},\boldsymbol{C}}(\boldsymbol{Y}) \tag{5}$$

More specifically, the objective function in Eq. (4) contains four components as follows:

First, $\sum_{(i,j,k)\in\Omega}(\boldsymbol{Y}_{i,j,k} - \sum_{r=1}^{R} \boldsymbol{A}_{i,r}\boldsymbol{B}_{j,r}\boldsymbol{C}_{k,r})^2$ quantifies the error of $Y \approx \sum_{r=1}^{R} \boldsymbol{A}_{:,r} \otimes \boldsymbol{B}_{:,r} \otimes \boldsymbol{C}_{:,r}$.

Second, $R_T(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C})$ is the temporal regularizer defined as

$$R_T(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}) = \sum_{i=1}^{D}\sum_{j=1}^{K}\sum_{k=1}^{L}(\sum_{r=1}^{R} A_{i,r}B_{j,r}(C_{k,r} - C_{k-1,r}))^2 \tag{6}$$

which restricts the adjacent element of the solution of $\hat{Y} \triangleq \sum_{r=1}^{R} \boldsymbol{A}_{:,r} \otimes \boldsymbol{B}_{:,r} \otimes \boldsymbol{C}_{:,r}$ should be close.

Third, $R(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C})$ is a L_2 regularizer defined as:

$$R(\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}) = ||\boldsymbol{A}||_F^2 + ||\boldsymbol{B}||_F^2 + ||\boldsymbol{C}||_F^2 \tag{7}$$

which solve the overfitting problem of machine learning.

Solving Rule

To find the optimized $\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{C}$ that minimizes the objective function as shown in Eq. (4), we adopt the stochastic gradient descent approach with the following update rules:

$$\begin{aligned} \boldsymbol{A}_{i,:}^{new} = \boldsymbol{A}_{i,:} + \eta[\boldsymbol{err}_{i,j,k}(\boldsymbol{B}_{j,:} \odot \boldsymbol{C}_{k,:}) - \lambda_r \boldsymbol{A}_{i,:} \\ -\lambda_s(\Gamma \odot (\Delta \odot \boldsymbol{C}_{k,:})) \\ -\lambda_s(\Gamma' \odot (\Delta' \odot \boldsymbol{C}_{k,:}))] \end{aligned} \tag{8}$$

$$\begin{aligned} \boldsymbol{B}_{j,:}^{new} = \boldsymbol{B}_{j,:} + \eta[\boldsymbol{err}_{i,j,k}(\boldsymbol{A}_{i,:} \odot \boldsymbol{C}_{k,:}) - \lambda_r \boldsymbol{B}_{j,:} \\ -\lambda_s(\Gamma \odot (\boldsymbol{A}_{i,:} \odot \boldsymbol{C}_{k,:})) \\ -\lambda_s(\Gamma' \odot (\boldsymbol{A}_{i,:} \odot \boldsymbol{C}_{k,:}))] \end{aligned} \tag{9}$$

$$\begin{aligned} \boldsymbol{C}_{k,:}^{new} = \boldsymbol{C}_{k,:} + \eta[\boldsymbol{err}_{i,j,k}(\boldsymbol{A}_{i,:} \odot \boldsymbol{B}_{j,:}) - \lambda_r \boldsymbol{C}_{k,:} \\ -\lambda_s(\Gamma \odot (\boldsymbol{A}_{i,:} \odot \Delta)) \\ -\lambda_s(\Gamma' \odot (\boldsymbol{A}_{i,:} \odot \Delta'))] \end{aligned} \tag{10}$$

which:

$$\begin{aligned} \Delta &= \boldsymbol{B}_{j,:} - \boldsymbol{B}_{j-1,:} \\ \Gamma &= \boldsymbol{A}_{i,:}(\Delta \odot C_{k,:}) \\ \Delta' &= \boldsymbol{B}_{j+1,:} - \boldsymbol{B}_{j,:} \\ \Gamma' &= \boldsymbol{A}_{i,:}(\Delta' \odot C_{k,:}) \end{aligned}$$

$$\boldsymbol{err}_{i,j,k} = \boldsymbol{Y}_{i,j,k} - \sum_{r=1}^{R} \boldsymbol{A}_{i,r} \boldsymbol{B}_{j,r} \boldsymbol{C}_{k,r}$$

4.2 Relation to Singular Spectrum Analysis

The HTF algorithm this paper put forward can be thought of as the tensor factorization version of singular spectrum analysis. Hassani et al. [6] pointed out, as a novel and powerful time series analysis technique, which can be used to process the time series from dynamic system, signal processing, economy and many other spheres, singular spectrum analysis decomposes the original data into the sum of a series of independent and explainable components, such as smooth trend, period components, quasi-period components and non-structural noise.

It generally consists of two steps: decomposition and reconstruction. The first step is to hankelize the time series, then do singular value decomposition: $H(\boldsymbol{X}) \approx \lambda_1 \boldsymbol{U}_1 \boldsymbol{V}_1 + \lambda_2 \boldsymbol{U}_2 \boldsymbol{V}_2 + ... + \lambda_n \boldsymbol{U}_n \boldsymbol{V}_n$. The second step is to select eigenvalues that have large impact on the reconstruction by $\sum_{l=1}^{r} \lambda_l / \sum_{l=1}^{n} \lambda_l \geq threshold$, with the purpose of denoising data. So the reconstruction of $H(\boldsymbol{X})$ is $r < n, \hat{H}(\boldsymbol{X}) = \lambda_1 \boldsymbol{U}_1 \boldsymbol{V}_1 + ... + \lambda_r \boldsymbol{U}_r \boldsymbol{V}_r$. At the last, get the estimation of $\boldsymbol{X}$ by diagonal average approach.

Golyandina et al. [5] suggested singular spectrum analysis is the optimal reconstruction for the time series satisfying the Eq. (11):

$$x_n = c_1 x_{n-1} + c_2 x_{n-2} + ... + c_k x_{n-k} \tag{11}$$

This property makes it powerful for lots of time series since many time-evolving processes in real scenarios can be represented or approximately represented by this formula.

Due to the characteristic of singular value decomposition (SVD) that treats the missing value of matrix to decompose as zero or some constant, singular spectrum analysis always takes this default initialization of missing as part of the trend in data, resulting in the lack of capacity of the method to recover missing blocks. Although Mahmoudvand et al. [13] proposed a kind of improvement based on singular spectrum analysis, it has a strict demand on the shape of missing, which can not deal with arbitrary shapes of missing blocks. However, the tensor factorization method proposed by this paper which uses stochastic gradient descent does not take missing into consideration, making it adapt to the problem.

Since apart from the approach to decompose the original matrix in the procedure of decomposition, the rest steps of the SSA and HTF are almost the same, there is no doubt that the HTF has the same capacity as SSA if tensor factorization technique can reach the effect of SVD which is shown in [12].

5 Experiments

5.1 Experiment Setting

In this section, we conducted experiments using real-world data sets to evaluate the performance of our approach based on the benchmark of Khayati et al., appending the comparison of deep learning approach to get a more exhaustive evaluation.

The environment of the experiments is a 4-core Intel i5-7300HQ CPU, NVIDIA GTX 1050 GPU. The implementation of all methods mentioned in experiments are coded with Pytorch1.5 and Python3.6.

A. Data Sets and Experiment Methodology

Five real-world data sets are used for the experiments[1]:

- Air Quality: contains air quality data in a city of Italy from 2004 to 2005. There is a periodic trend and jumping changes in the data. We cut it to 10 variables, 1000 time points.
- Electricity: contains family electricity usage in France per minute from 2006 to 2010. There is a strong time varying property in it. We cut it to 20 variables, 2000 time points.

[1] All 5 datasets can be found on https://archive.ics.uci.edu/ml/datasets.php.

- Temperature: contains temperature data sampled in weather stations across China from 1960 to 2012. There is a high correlation between variables. We cut it to 50 variables, 5000 time points.
- Gas: contains gas data collected by a chemical laboratory in the United States in a gas extraction platform from 2007 to 2011. There is a huge difference in correlation between variables. We cut it to 100 variables, 1000 time points.
- Chlorine: contains chlorine data from 166 intersections in a water system, sampled every 5 min during 15 days. There is a very smooth trend in data, with a strong periodicity. We cut it to 50 variates, 1000 time points.

The evaluation metric we used in this paper is:

$$NRMSE(\boldsymbol{X}, \hat{\boldsymbol{X}}) = \sqrt{\frac{1}{|\Omega_{test}|} \sum_{(i,j)\in\Omega_{test}} (\boldsymbol{X}_{i,j} - \hat{\boldsymbol{X}}_{i,j})^2 / \left(\frac{1}{|\Omega_{test}|} \sum_{(i,j)\in\Omega_{test}} |\boldsymbol{X}_{i,j}|\right)}$$

where $\hat{\boldsymbol{X}}$ is the result that the missing in the original matrix is filled with estimation. Ω_{test} is the set of all indices that value is missing. The smaller the NRMSE of the algorithm is, the better the recovery effect is.

The reason why use NRMSE is that not only the recovery effect of different methods on the same data set can be compared, but also the effect of one method on different datasets can be evaluated. It has wide use in papers like [11,14,18], becoming a popular metric to evaluate the effect of a method to recover missing values.

To demonstrate the effectiveness of HTF, we compare its performance against the following baseline approaches: 1) TKCM [16]; 2) Dynammo; [10]; 3) TRMF [18]; 4) BRITS [3]. The brief statement of their work has been introduced in Sect. 2.

The original dataset in the benchmark is complete, we simulate the different missing patterns by generating the related indicating matrix $\boldsymbol{W}$, and the missing blocks only appear in the middle of the series to avoid the end case which is similar to forecasting tasks that the methods we experiment usually perform badly. Because of the randomness of result caused by the position of missing generated randomly (There is a large difference between the smooth sequence piece and steep sequence piece in the recovery effect.), we conduct a duplicate experiment way to reduce the randomness.

B. Empirical Parameters Setting

$K, L = T + 1 - K$ in the step of hankelization, $\lambda_r, \lambda_s, \eta$ in the step of update rule, and the number of iterations *iter_num* that applying update rule are all the hyper-parameters that we need to set empirically.

Before we compare our method with others, we carried out several trials to select these hyper-parameters of HTF to avoid bad performance of our method. So in the end, we set $\lambda_s = \lambda_r \in 0.0001, 0.001, 0.01$ for different datasets, $R =$

$\lceil 1.2 * D \rceil$, $K = \lceil 1.5 * l_M \rceil$, and the number of iterations of gradient descent is $iter\_num = 100$. One process that we did experiments to find a good parameter is shown in Sect. 5.2.D. Also, we do the same process for each baseline method if it has selective parameters.

5.2 Recovery Effect

A. Performance on Disjoint and Overlap

In this set of experiments, when generating indicating matrix $\boldsymbol{W}$, the number of involved variables is at least 4, at most about 40% of total. In the pattern of Overlap, $h_M = 2$.

From Table 1 and Table 2, it can be concluded that overlap case is more difficult to recover effectively than disjoint, and almost all methods have the a degree of performance reduction on all data sets. Many methods can achieve good results when complete sequences of variables exists. The reason is that the complex dependence of variables in time series at each timestamp can be learned based on the complete sequence of certain variables as a reference. Among them, the BRITS algorithm based on deep learning achieved the best performance, and HTF did not get a good enough result. The essential of this phenomenon is that HTF focuses on learning the autocorrelation of variables while other methods more focus on the correlations between variables.

Table 1. Performance on disjoint pattern

Dataset	TKCM	Dynammo	TRMF	BRITS	HTF
Air quality	0.916	0.707	0.494	**0.018**	0.620
Electricity	0.838	0.867	0.811	**0.683**	0.704
Chlorine	0.323	0.034	**0.015**	0.047	0.129
Gas	1.740	0.384	0.081	**0.068**	0.467
Temperature	0.426	0.133	0.111	**0.099**	0.281

Table 2. Performance on overlap pattern

Dataset	TKCM	Dynammo	TRMF	BRITS	HTF
Air quality	1.244	0.770	0.672	**0.578**	0.714
Electricity	0.863	1.347	0.873	**0.489**	0.674
Chlorine	0.335	0.095	**0.061**	0.081	0.165
Gas	1.726	0.476	0.383	**0.073**	0.452
Temperature	0.442	0.481	0.430	**0.140**	0.288

B. Performance on Blackout

In this set of experiments, generating indicating matrix $\boldsymbol{W}$, the number of involved variables is about 40% of total. the length of each missing blocks is 100.

From Table 3, we can find the HTF algorithm we proposed achieved the best NRMSE in all datasets. All other methods perform really bad, they lost the ability to recover the missing, which can be confirmed both in nrmse result and Fig. 3. In [2], authors discussed the reason why dynammo fails in the blackout pattern. Due to only extracting the most similar subsequence to fill the missing, TKCM can not adapt the local dynamic well, so the shifting phenomenon is very serious. The matrix factorization technique TRMF takes can not handle the situation that there is a column in the matrix that is missing, which caused one column of embedding not having information to update its value. So the imputation may look like white noise with little variance with random initialization. For BRITS, long intermediate process missing, a kind of gradient vanishing is hard to solve for neural networks, which is the inherent defect of deep learning, and there may only be fluctuation values near the end of missing blocks, while the middle of the block is smooth.

Figure 3 shows the picture of what the recovery result looks like compared with the original data in the blackout missing pattern on the Electricity dataset, which is evidence of the analysis we give before. The time points of the missing block are from 325 to 425, and the red one in the figure is the original data, while the blue is the image of recovery. It is not hard to see only our method return the result looks reflecting the characteristic of original time series and there is no much difference between the recovery and truth.

Table 3. Performance on blackout pattern

Dataset	TKCM	Dynammo	TRMF	BRITS	HTF
Air quality	1.254	1.070	1.252	1.797	**0.763**
Electricity	1.235	1.165	1.273	1.604	**0.652**
Chlorine	0.521	0.691	0.517	0.516	**0.275**
Gas	1.459	1.235	1.202	1.737	**0.606**
Temperature	1.108	0.542	1.123	0.593	**0.219**

C. Influence of Different Ways of Hankelization

The two ways of hankelization introduced in 4.1.A is valid in theory, and the effect can hardly be analyzed by mathematical. So we compared these two ways in experiments, we name the way we use tensor hankelization HTF, and the way we use the matrix hankelization HTF-M. Which can be seen from Table 4, HTF outperforms HTF-M in almost all cases, though the advantage is not obvious, this kind of difference is enough in Statistical.

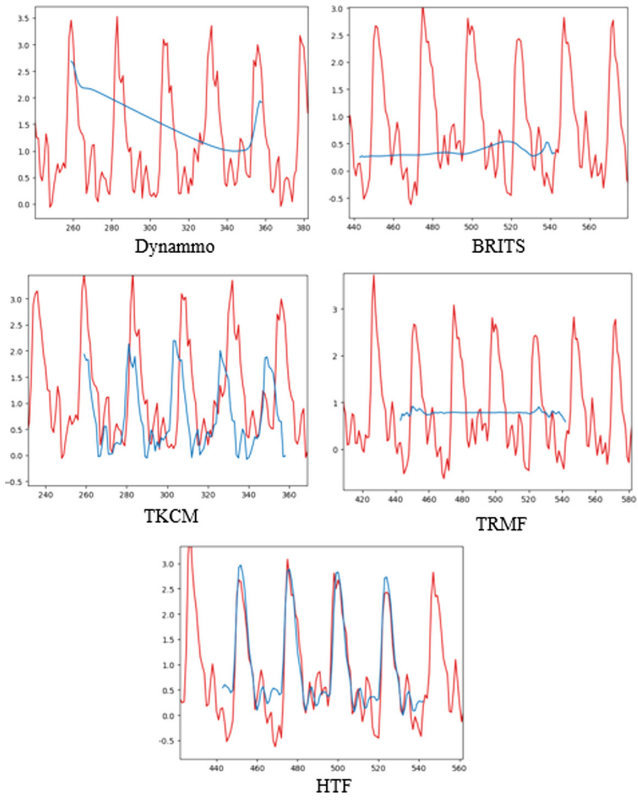


Fig. 3. Recovery effect of different methods on Electricity dataset

Table 4. Influence of different ways of hankelization

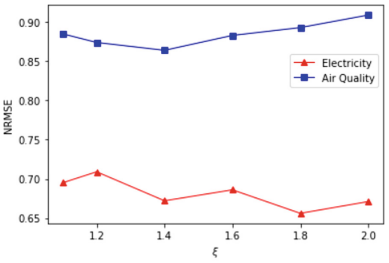
Dataset	HTF	HTF-M
Air quality	**0.763**	0.798
Electricity	**0.652**	0.658
Chlorine	**0.275**	0.293
Gas	**0.606**	0.632
Temperature	**0.219**	0.239

D. Influence of parameter K

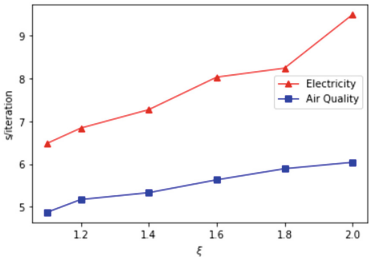
The selection of K may have a big impact in performance , and this set of experiments aims to find a good K. Because of K should satisfy $K \geq l_M + 1$, we set $K = \lceil \xi l_M \rceil (\xi > 1)$. By adjust the value of ξ, we can see the impact of different K.

Figure 4 shows the result of the influence of different ξ on NRMSE and time cost every iteration in Air Quality and Electricity data with parameters fixed

with $\lambda_s = \lambda_r = 0.0001, iter\_num = 100.$ along with the increase of ξ, time cost each iteration shows a monotonous increase trend, while the NRMSE decreases when $\xi \in (1, 1.6]$, and goes smooth when $\xi > 1.6$. A good value of k should balance both recovery effect and time cost the program does, so $\xi = 1.5, K = 1.5 * l_M$ is a good selection.



(a) NRMSE of HTF



(b) time cost per iteration of HTF

Fig. 4. The recovery effect and time cost of HTF related to ξ

5.3 Result Conclusion

In conclusion, the HTF algorithm achieved an excellent result in blackout pattern compared to other baseline methods, while also performs well in other missing patterns. It is an effective algorithm for time series to recover missing blocks.

6 Conclusion

This paper presents a novel tensor factorization-based approach called HTF to address the challenging problem of estimating the values of missing blocks in time series, especially the blackout missing pattern by decomposing multivariate data sequence into time-evolving embedding, time delaying embedding and multivariate embedding. Following this idea, the method first transforms a time series matrix into a hankelized version. Through the experiments on the benchmark inherited from the previous work, we demonstrate the effectiveness of HTF by comparing its performance against state-of-art baseline approaches.

For future work, we plan: 1) design a deep learning method to handle the Blackout missing pattern Specifically, aiming to deal with the case that time series does not satisfy linear recursive formula. 2) Solving the problem when data is sampled unevenly.

References

1. Aydilek, I.B., Arslan, A.: A hybrid method for imputation of missing values using optimized fuzzy c-means with support vector regression and a genetic algorithm. Inf. Sci. (Ny) **233**, 25–35 (2013)
2. Cai, Y., Tong, H., Fan, W., Ji, P.: Fast mining of a network of coevolving time series, pp. 298–306 (2015). https://doi.org/10.1137/1.9781611974010.34
3. Cao, W., Wang, D., Li, J., Zhou, H., Li, Y., Li, L.: Brits: bidirectional recurrent imputation for time series. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems, NIPS 2018, pp. 6776–6786. Curran Associates Inc., Red Hook (2018)
4. Miao, D., Cai, Z., Li, J., Gao, X., Liu, X.: The computation of optimal subset repairs. Proc. VLDB Endow. **13**, 2061–2074 (2020)
5. Golyandina, N., Korobeynikov, A.: Basic singular spectrum analysis and forecasting with R. Comput. Stat. Data Anal. **71**, 934–954 (2014). https://doi.org/10.1016/j.csda.2013.04.009
6. Hassani, H.: Singular spectrum analysis: Methodology and comparison. University Library of Munich, Germany, MPRA Paper 5 (2007)
7. Hassani, H., Mahmoudvand, R.: Multivariate singular spectrum analysis: a general view and new vector forecasting approach. Int. J. Energy Stat. **01**, 55–83 (2013). https://doi.org/10.1142/S2335680413500051
8. Hidasi, B., Tikk, D.: Fast ALS-based tensor factorization for context-aware recommendation from implicit feedback. In: Flach, P.A., De Bie, T., Cristianini, N. (eds.) ECML PKDD 2012. LNCS (LNAI), vol. 7524, pp. 67–82. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33486-3_5
9. Khayati, M., Lerner, A., Tymchenko, Z., Cudre-Mauroux, P.: Mind the gap: an experimental evaluation of imputation of missing values techniques in time series. Proc. VLDB Endow. **13**, 768–782 (2020). https://doi.org/10.14778/3377369.3377383
10. Li, L., Mccann, J., Pollard, N., Faloutsos, C.: DynaMMo : mining and summarization of coevolving sequences with missing values. In: KDD 2009 Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 507–516 (2009). https://doi.org/10.1145/1557019.1557078
11. Li, Z., Ye, L., Zhao, Y., Song, X., Teng, J., Jin, J.: Short-term wind power prediction based on extreme learning machine with error correction. Prot. Control Modern Power Syst. **1** (2016). https://doi.org/10.1186/s41601-016-0016-y
12. Maehara, T., Hayashi, K., Kawarabayashi, K.T.: Expected tensor decomposition with stochastic gradient descent. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, AAAI 2016, pp. 1919–1925. AAAI Press (2016)
13. Mahmoudvand, R., Rodrigues, P.: Missing value imputation in time series using singular spectrum analysis. Int. J. Energy Stat. **04**, 1650005 (2016). https://doi.org/10.1142/S2335680416500058
14. Pennekamp, F., et al.: The intrinsic predictability of ecological time series and its potential to guide forecasting. Ecol. Monogr. **89**, e01359 (2019)

15. Sterne, J.A.C., et al.: Multiple imputation for missing data in epidemiological and clinical research: potential and pitfalls. BMJ **338** (2009). https://doi.org/10.1136/bmj.b2393. https://www.bmj.com/content/338/bmj.b2393
16. Wellenzohn, K., Böhlen, M.H., Dignös, A., Gamper, J., Mitterer, H.: Continuous imputation of missing values in streams of pattern-determining time series. In: EDBT (2017)
17. Wu, S., Wang, L., Wu, T., Tao, X., Lu, J.: Hankel matrix factorization for tagged time series to recover missing values during blackouts. In: 2019 IEEE 35th International Conference on Data Engineering (ICDE), pp. 1654–1657 (2019). https://doi.org/10.1109/ICDE.2019.00165
18. Yu, H.F., Rao, N., Dhillon, I.S.: Temporal regularized matrix factorization for high-dimensional time series prediction. In: Lee, D.D., Sugiyama, M., Luxburg, U.V., Guyon, I., Garnett, R. (eds.) Advances in Neural Information Processing Systems, vol. 29, pp. 847–855. Curran Associates, Inc. (2016). http://papers.nips.cc/paper/6160-temporal-regularized-matrix-factorization-for-high-dimensional-time-series-prediction.pdf