This copy is the authors' edition of our paper "Effective Prediction of Missing Data on Apache Spark over Multivariable Time Series" published by IEEE Transactions on Big Data. The final version should be retrived from IEEE library.

Effective Prediction of Missing Data on Apache Spark over Multivariable Time Series

Weiwei Shi[1], Yongxin Zhu[1,*], Philip S. Yu[2], Jiawei Zhang[2], Tian Huang[3], Chang Wang[1], and Yufeng Chen[4]
[1]School of Microelectronics, Shanghai Jiao Tong University, China
[2]Department of Computer Science, University of Illinois at Chicago, USA
[3]Cavendish Laboratory, University of Cambridge
[4]Shandong Power Supply Company of State Grid, China
{iamweiweishi, zhuyongxin}@sjtu.edu.cn, {psyu, jzhan9}@uic.edu,
th523@cam.ac.uk, willy@sjtu.edu.cn, chenyufeng@sina.com

More massive volume of data are generated in many areas than ever before. However, the missing of some values in collected data always occurs in practice and challenges extracting maximal value from these large scale data sets. Nevertheless, in multivariable time series, most of the existing methods either might be infeasible or could be inefficient to predict the missing data. In this paper, we have taken up the challenge of missing data prediction in multivariable time series by employing improved matrix factorization techniques. Our approaches are optimally designed to largely utilize both the internal patterns of each time series and the information of time series across multiple sources. Based on the idea, we have imposed three different regularization terms to constrain the objective functions of matrix factorization and built five corresponding models. Extensive experiments on real-world data sets and synthetic data set demonstrate that the proposed approaches can effectively improve the performance of missing data prediction in multivariable time series. Furthermore, we have also demonstrated how to take advantage of the high processing power of Apache Spark to perform missing data prediction in large scale multivariable time series.

***Index Terms*—matrix factorization, missing data prediction, time series, Big Data, Apache Spark**

I. Introduction

The sophistication of instruments to collect data from multiple sources and the resulting volume of data have grown to an unprecedented level since the era of big data started in recent years [1]. Multivariable time series, one common data format, are ubiquitous in many real-world applications, like electric equipment monitoring, weather or economic forecasting, environment state monitoring, security surveillance and many more [2], [3], [4]. In most applications, multiple sensors are employed to generate time series data, and they usually share one common goal. For example, in the power grid system, various diagnostic gases sensors are deployed to monitor the status of the main power transformers and generate multivariable time series by measuring the content of the diagnostic gases over time [5]. In the "Internet of Things", a large number of sensors are used to produce multivariable time series of the external environment, e.g., the air or water quality. In the medical and healthcare systems, multiple sensors can also be equipped within living spaces to monitor the health and general well-being of senior citizens, while also ensuring that proper treatment is being administered and assisting people regaining lost mobility via therapy as well [6]. In this paper, the sensors sharing one common goal are treated as a sensor network.

Unfortunately, due to the harsh working conditions or uncontrollable factors, such as the extreme weather, equipment failure or the unstable communication signal, the raw time series in a sensor network usually involve missing values. For instance, while in-service, missing values in power grid surveillance systems can occur for various reasons, such as quick evaporation of acetylene, the existence of contamination on the surface of the platinum alloy of a gas meter, etc. Yet, in practice, sensors and communication failures are more common factors that produce missing values in many applications. And still worse, immediate fixation of these practical problems is rarely plausible and might cost too much.

The inevitable data missing necessitates integrated analysis of observed data sets. A large collection of data mining and statistical methods have been proposed to predict the missing values of time series [7]. One simplest solution might be linear interpolation. But it is only feasible to be applied to the case where only a low ratio of collected data are missing and the time series vary very steadily. Modeling methods, more commonly used solutions, try to discover the underlying patterns and seasonality to predict the missing values using some common sense [8]. Representative modeling approaches include deterministic models, stochastic models, and state space models [7]. For example, Frasconi et al. employed a seasonal kernel to measure the similarity between time-series instances and proposed the seasonal autoregressive integrated moving average model coupled with a Kalman filter achieved excellent performance of missing data prediction [9]. Baraldi et al. [10] proposed a fuzzy method for missing data reconstruction, which involves fuzzy similarity measurement and shows superiority to an auto associative kernel regression method. Besides, Song et al. used matrix factorization to predict traffic matrices and their method showed more effective performance than traditional methods [11].

Nevertheless, these methods either focus on predicting the missing data in the time series from one single source or could not effectively handle the missing data prediction problem of the time series from multiple sources. For instance, the fuzzy method might lose their effectiveness when the missing ratio is too high as the method mainly based on the quality of

Corresponding author: Yongxin (email: zhuyongxin@sjtu.edu.cn) similar time series. As a consequence, this method largely depends on the quality of the raw observed data. The SARMIA method aims at predicting the missing values with underlying seasonality, and thus the method might have trouble modelling data that have no strict internal seasonality. Especially, when the amount of instances becomes too large, the seasonal kernel computation would cost too much time. The common matrix factorization method usually needs to incorporate the internal specific characteristic of the time series data, such as the spatial information, which might restrict the method to one specific application.

In this paper, aiming at solving the above problems, we propose to fuse the temporal smoothness of time series and the information across multiple sources into matrix factorization in order to improve the accuracy of missing data prediction in multivariable time series. First, as each time series rarely fluctuate wildly over time, i.e., time series usually host internal and tangible pattern of temporal smoothness. Thus, we try to take advantage of the characteristic to reduce the prediction error of the missing data prediction in multivariable time series. Concretely, a selective penalty term is employed in the matrix factorization objective function to smooth the time series, i.e., we aim at minimizing the fluctuation of each time series with time. Second, as there exists valuable correlation information across multiple sources in a sensor network, we also attempt to fuse that information into matrix factorization to obtain higher performance. Specifically, the correlation information is incorporated in designing two sensor network regularization terms, i.e., the correlated sensors based regularization (CSR) term and the uncorrelated sensors based regularization (USR) term, to constrain the matrix factorization objective function. We take aim at minimizing the difference between a sensor and its correlated sensors or maximizing the difference between a sensor and its uncorrelated sensors based on the sensor network regularization. Moreover, to treat the correlated or uncorrelated sensors differently, we further improve the sensor network regularization terms of the objective function by incorporating similarity functions. By taking advantage of the internal characteristic of each time series and the knowledge of time series across multiple sources, five models are built in the paper:

1) MFS: Matrix Factorization with Smoothness constraints;
2) CSM: Correlated Sensors based Matrix factorization;
3) USM: Uncorrelated Sensors based Matrix factorization;
4) CSMS: Correlated Sensors based Matrix factorization with Smoothness constraints;
5) USMS: Uncorrelated Sensors based Matrix factorization with Smoothness constraints;

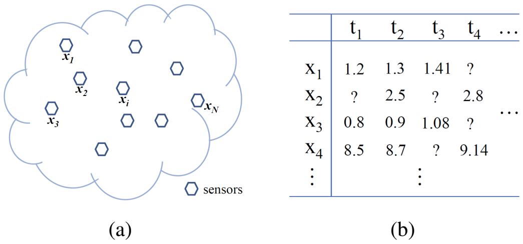
In the era of big data, more massive volume of time series data are generated nowadays than ever before. Besides, analyzing big data is a complex and time-consuming task, which needs more efficient and specific analysis tool than traditional ones. Thus parallel versions of matrix factorization have become of great interest. Apache Spark is a large-scale distributed data processing environment that builds on the principles of scalability and fault tolerance that are instrumental in the success of Hadoop and MapReduce [12], [13]. Apache Spark has already implemented a fundamental version of matrix factorization for recommendation. Here, we implement our proposed approaches using the Apache Spark platform.

The experimental results reveal that our proposed methods show superior performance to the traditional and state-of-the-art algorithms.

The contributions of this paper are summarized as follows:

1) We propose novel methods to constrain the matrix factorization by fusing both the temporal smoothness of each time series and the information across multiple sources to improve the performance of missing data prediction in multivariable time series. These constraints are leveraged to largely utilize interior characteristic of time series data.
2) We elaborate how the smoothness constraints are carefully designed and how the correlation information across the different sources in a sensor network can contribute to the missing data prediction in multivariable time series. And we incorporate smoothness constraints and two sensor network regularization terms to constrain the matrix factorization respectively. Also, we systematically illustrate how to design matrix factorization objective functions with the carefully designed regularization terms.
3) We implement and verify the proposed methods with three data sets from real world and one synthetic data set. And for big data analysis, we also implement and verify the proposed methods on Apache Spark platform.

II. Problem Formulation



	t_1	t_2	t_3	t_4	…
x_1	1.2	1.3	1.41	?	
x_2	?	2.5	?	2.8	…
x_3	0.8	0.9	1.08	?	
x_4	8.5	8.7	?	9.14	
⋮		⋮			

Fig. 1. (a) Illustration of a sensor network; (b)multivariable time series.

In this paper, we focus on the missing data prediction problem of the time series in multiple sources. To illustrate this problem more clearly, we show an example in Fig. 1. Fig. 1(a) presents a simplified example of a sensor network. Fig. 1(b) is a sensors-time matrix, i.e., a multivariable time series, collected from the sensors in (a)'s network.

Table I lists the main symbols we use throughout this paper. Let $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3, \ldots, \boldsymbol{x}_N\}$ be the multivariable time series collected from N different data sources, and the jth entity in time series data from the ith source can be denoted as X_{ij} for $i = \{1, 2, 3, \ldots, N\}$, $j = \{1, 2, 3, \ldots, M\}$ and $X_{ij} \geq 0$. In our case, when X_{ij} in multivariable time series is missing, it will be denoted as '?'. In addition, we use a matrix $W \in \mathbb{R}^{N*M}$ to indicate whether the value in X is missing or observed. The entries of W can be defined as:

$$w_{ij} = \begin{cases} 0 \; if \; X_{ij} \; is \; missing \\ 1 \; otherwise. \end{cases} \tag{1}$$

TABLE I
SYMBOLS AND DEFINITION.

Symbols	Definition and Description
X	multivariable time series
N	number of data sources
M	length of each time series
x_i	the time series from ith data source
W	indicator matrix
S, V	latent factors
L	dimension of latent factors
H	similarity functions
B_{ij}	the element at the ith row and jth column of matrix B
B_i	the ith row of matrix B
B^T	transpose of matrix B
$\circ$	Hadamard product

for all $i = \{1, 2, 3, \ldots, N\}, j = \{1, 2, 3, \ldots, M\}$. The problem of the missing data prediction in multivariable time series can be defined as follows:

Problem 1: Missing data prediction in multivariable time series.

Given: a multivariable time series $X \in \mathbb{R}^{N*M}$; an indicator matrix W;

Prediction: the predicted values of the missing entries indicated by W.

III. PROPOSED METHODS

In this section, we describe the details of the proposed methods for missing data prediction in multivariable time series. We begin with the discussion about the baseline solution for the problem. Next, we elaborate the key idea of the proposed methods. We present how to take advantage of the smoothness characteristic to reduce the error of the missing data prediction in multivariable time series. Besides, we also elaborate why and how to utilize the valuable correlation information across multiple sources in time series data collected from one sensor network to improve the prediction performance. Given the main idea, we carefully design three regularization terms to constrain the matrix factorization and then build five different models. In the process of regularization terms design, five similarity functions are introduced, which are also key components of the proposed method. Finally, we give the detailed realization of the proposed methods based on Apache Spark.

A. Low Rank Matrix Factorization

The singular value decomposition (SVD) is a popular and effective factorization of a real or complex matrix. SVD approach focuses on discovering linear correlations among time series and on applying these correlations for further data analysis [14]. Given $X \in \mathbb{R}^{N*M}$, the singular value decomposition of X is given by

$$X = U\Sigma Q^T, \tag{2}$$

where $U \in \mathbb{R}^{N*L}$ is a matrix with orthogonal columns and $Q \in \mathbb{R}^{M*L}$, and $\Sigma \in \mathbb{R}^{L*L}$ is a diagonal matrix containing the singular values of X along its main diagonal.

The most popular low-rank factorization is obtained when the SVD is rearranged as

$$X = (U\Sigma^{\frac{1}{2}})(\Sigma^{\frac{1}{2}}Q^T) = SV^T, \tag{3}$$

where $S = (U\Sigma^{\frac{1}{2}}) \in \mathbb{R}^{N*L}$ and $V = (\Sigma^{\frac{1}{2}}Q^T) \in \mathbb{R}^{M*L}$ with $L < min(N, M)$.

A standard formulation of the problem is to determine S and V with respect to the existing components:

$$\min_{S,V} \frac{1}{2}\|X - SV^T\|_F^2. \tag{4}$$

As the original matrix X might contain a great number of missing values, we only need to factorize the observed entities in X. Hence, we have a modified optimization problem:

$$\min_{S,V} \frac{1}{2}\|W \circ (X - SV^T)\|_F^2 + \frac{\lambda_1}{2}\|S\|_F^2 + \frac{\lambda_2}{2}\|V\|_F^2, \tag{5}$$

where $\lambda_1, \lambda_2 > 0$ and $\circ$ denotes the Hadamard product. Two regularization terms $\|S\|_F^2$ and $\|V\|_F^2$ are added in order to avoid overfitting [15]. Gradient based approaches can be applied to find a local minimum in Equation (5) due to their effectiveness and simplicity [16]. Equation (5) also contains a nice probabilistic interpretation with Gaussian observation noise, which is detailed in [17]. The product of S and V^T is the reconstructed X and is denoted as $\hat{X}$ in the paper.

In general, from the aforementioned formulation, we can provide a practical configuration of our methods:

$$\begin{aligned}\min_{S,V} \frac{1}{2}&\|W \circ (X - SV^T)\|_F^2 + \frac{\lambda_1}{2}\|S\|_F^2 \\ &+ \frac{\lambda_2}{2}\|V\|_F^2 + \alpha_S J_S(S) + \alpha_V J_V(V),\end{aligned} \tag{6}$$

where α_S and α_V are nonnegative regularization parameters and the terms $J_S(S)$ and $J_V(V)$ are chosen to enforce the desired properties of the time series data. In the following subsections, we will show how to design effective regularization terms according to the properties of multivariable time series in the process of matrix factorization.

B. Fusion of Temporal Smoothness of Time Series

In the real world, a large amount of time series usually do not fluctuate wildly. For example, the room temperature, the gas concentrations in electric equipments, the energy consumption in cities, and product prices in the market rarely change drastically. In order to fuse the temporal smoothness of each time series, as V denotes the latent matrix with time dimension, optimization problem is improved as:

$$\begin{aligned}\min_{S,V} \mathcal{L}_V(X, S, V) = \frac{1}{2}&\|W \circ (X - SV^T)\|_F^2 + \frac{\lambda_1}{2}\|S\|_F^2 \\ &+ \frac{\lambda_2}{2}\|V\|_F^2 + \frac{\beta}{2}\|GV^T\|_F^2,\end{aligned} \tag{7}$$

where typical examples of the matrix G are the 1st derivative approximation $G_1 \in \mathbb{R}^{(L-1)\times L}$ and the 2nd derivative approximation $G_2 \in \mathbb{R}^{(L-2)\times L}$ [18], given by

$$G_1 = \begin{pmatrix} 1 & -1 & & & 0 \\ & 1 & -1 & & 0 \\ \vdots & & \ddots & \ddots & 0 \\ 0 & & & 1 & -1 \end{pmatrix}, \tag{8}$$

$$G_2 = \begin{pmatrix} -1 & 2 & -1 & & 0 \\ & -1 & 2 & -1 & 0 \\ \vdots & & \ddots & \ddots & 0 \\ 0 & & -1 & 2 & -1 \end{pmatrix}. \tag{9}$$

We denote this first model as Matrix factorization with Smoothness constraints (MFS). As Alternating Least Squares (ALS) can be done effectively, the key idea of which is to find the local optimum solution of S and V alternatively, where the gradients of $\mathcal{L}_V(X, S, V)$ with respect to S_i and V_i could be calculated as:

$$\frac{\partial \mathcal{L}_V}{\partial S_i} = \sum_{j=1}^{M} W_{ij}(S_i V_j^T - X_{ij})V_j + \lambda_1 S_i \tag{10}$$

for all i $\in \{1, 2, \ldots, N\}$,

$$\frac{\partial \mathcal{L}_V}{\partial V_j} = \sum_{i=1}^{N} W_{ij}(S_i V_j^T - X_{ij})S_i + \lambda_2 V_j + \beta V_j G^T G \tag{11}$$

for all j $\in \{1, 2, \ldots, M\}$.

C. Fusion of Information across Multiple Sources

Multiple sources provide access to the insight of the nearby raw data. First, we endeavor to fuse the valuable information across multiple sources by integrating correlated sources. Then, from the opposite view, uncorrelated sources also bring us significant insight into the distant raw data.

1) Correlated Sensors based Regularization

In a sensor network, in spite of the fact that the different sensors are assigned different tasks, they usually share one common goal and there might exist a strong correlation among some of the sensors. For example, in a smart building equipped with various sensors in humid areas, the humidity might go up with the temperature. So the humidity sensor may have a strong correlation with the temperature sensor. In environmental monitoring systems, there also might be a high correlation between the chemical and biological sensors as their detection values may possibly change simultaneously. As for the personal medical care, the blood pressure usually increases with the heartbeat, thus the corresponding sensors may have a strong correlation. If one sensor has a strong correlation with another one, we call the two sensors are $correlated$.

As S denotes the latent sensor matrix and there might be strong correlation among correlated sensors, we propose the second missing data prediction model based on matrix factorization technique, i.e., Correlated Sensors based Matrix factorization (CSM), with the following optimization problem:

$$\begin{aligned} \min_{S,V} \mathcal{L}_S(X, S, V) =& \frac{1}{2}\|W \circ (X - SV^T)\|_F^2 \\ &+ \frac{\lambda_1}{2}\|S\|_F^2 + \frac{\lambda_2}{2}\|V\|_F^2 \\ &+ \frac{\alpha}{2}\sum_{i=1}^{N}\|S_i - \frac{1}{|C(i)|}\sum_{c \in C(i)} \rho_{i,c} S_c\|_F^2, \end{aligned} \tag{12}$$

where α is the penalty factor and $\alpha > 0$, $C(i)$ denotes the set of the correlated sensors of the ith sensor and $|C(i)|$ is the total number of these correlated sensors. The included scaling factor $\rho_{i,c}$ aims at matching the scale difference between the ith sensor and the cth sensor. In this model, we incorporate one sensor network regularization term, i.e., the Correlated Sensors based Regularization (CSR) term

$$\frac{\alpha}{2}\sum_{i=1}^{N}\|S_i - \frac{1}{|C(i)|}\sum_{c \in C(i)} \rho_{i,c} S_c\|_F^2, \tag{13}$$

in order to minimize the distance between the ith sensor and its correlated sensors. Concretely, if the correlated sensors are $C(i)$, then we deduce that the state of the ith sensor is correlated to the average state of $C(i)$.

The above sensor network regularization imposes a hypothesis that the state between the ith sensor and the average state of $C(i)$ is very close, after scale adjustment. However, this hypothesis is usually invalid in the real world. For instance, there is one temperature sensor, one humidity sensor and one light sensor in a smart room. The temperature sensor might have a stronger correlation with the light sensor than the humidity sensor. Thus, a more practical model should treat the correlated sensors in $C(i)$ differently based on how correlated they are with the ith sensor. As a consequence, the optimization problem in Equation (12) is improved as:

$$\begin{aligned} \min_{S,V} \mathcal{L}_S(X, S, V) =& \frac{1}{2}\|W \circ (X - SV^T)\|_F^2 + \frac{\lambda_1}{2}\|S\|_F^2 \\ + \frac{\lambda_2}{2}\|V\|_F^2 &+ \frac{\alpha}{2}\sum_{i=1}^{N}\|S_i - \frac{\sum\limits_{c \in C(i)} H(i,c) * \rho_{i,c} S_c}{\sum\limits_{c \in C(i)} H(i,c)}\|_F^2. \end{aligned} \tag{14}$$

The sensor network regularization item CSR in Equation (14) is designed to treat each sensor in $C(i)$ differently. The function $H(i, c)$ measures the similarity between the ith sensor and the cth sensor. From this improved regularization item, we know that if the cth sensor is very correlated to the ith sensor, the value of $H(i, c)$ will be large, i.e, it contributes more to the state of the ith sensor. Similarly, the gradients of $\mathcal{L}_S(X, S, V)$ with respect to S_i and V_i could be calculated as:

$$\begin{aligned} \frac{\partial \mathcal{L}_S}{\partial S_i} =& \sum_{j=1}^{M} W_{ij}(S_i V_j^T - X_{ij})V_j + \lambda_1 S_i \\ &+ \alpha(S_i - \frac{\sum_{c \in C(i)} H(i,c) * \rho_{i,c} S_c}{\sum_{c \in C(i)} H(i,c)}), \end{aligned} \tag{15}$$

$$\frac{\partial \mathcal{L}_S}{\partial V_j} = \sum_{i=1}^{N} W_{ij}(S_i V_j^T - X_{ij})S_i + \lambda_2 V_j, \tag{16}$$

for all i $\in \{1, 2, \ldots, N\}$ and j $\in \{1, 2, \ldots, M\}$.

2) Uncorrelated Sensors based Regularization

The CSM model we propose imposes a regularization term based on correlated sensors to constrain the matrix factorization. From the opposite view, if one sensor has a weak correlation with another one, we call the two sensors are $uncorrelated$. And we also employ another sensor network regularization term, i.e., the uncorrelated sensors based regularization term, to build the Uncorrelated Sensors based Matrix factorization (USM) model. Since uncorrelated sensors share weak correlation, we attempt to add one regularization term to maximize the distance between the ith sensor and its uncorrelated sensors. Consequently, the optimization problem in Equation (14) is updated as:

$$\begin{aligned}\min_{S,V} \mathcal{L}'_S(X, S, V) &= \frac{1}{2}\|W \circ (X - SV^T)\|_F^2 \\ &+ \frac{\lambda_1}{2}\|S\|_F^2 + \frac{\lambda_2}{2}\|V\|_F^2 \\ &- \frac{\alpha'}{2}\sum_{i=1}^{N}\|S_i - \frac{\sum\limits_{c' \in C'(i)} H(i,c') * \rho_{i,c'} S_{c'}}{\sum\limits_{c' \in C'(i)} H(i,c')}\|_F^2.\end{aligned} \tag{17}$$

where α' is the penalty factor and $\alpha' > 0$, $C'(i)$ denotes the set of the uncorrelated sensors of the ith sensor. In contrast to the CSM model, we incorporate the other sensor network regularization term, i.e., the Uncorrelated Sensors based Regularization (USR) term. Similarly, the gradients of $\mathcal{L}'_S(X, S, V)$ with respect to S_i and V_i could be calculated as:

$$\begin{aligned}\frac{\partial \mathcal{L}'_S}{\partial S_i} &= \sum_{j=1}^{M} W_{ij}(S_i V_j^T - X_{ij})V_j + \lambda_1 S_i \\ &- \alpha'(S_i - \frac{\sum\limits_{c' \in C'(i)} H(i,c') * \rho_{i,c'} S_{c'}}{\sum\limits_{c' \in C'(i)} H(i,c')})\end{aligned} \tag{18}$$

$$\frac{\partial \mathcal{L}'_S}{\partial V_j} = \sum_{i=1}^{N} W_{ij}(S_i V_j^T - X_{ij})S_i + \lambda_2 V_j, \tag{19}$$

for all i $\in \{1, 2, \ldots, N\}$ and j $\in \{1, 2, \ldots, M\}$.

3) Similarity Function

The proposed regularization terms in Equation (14) and Equation (17) require a function H to measure the similarity between two sensors, which is a key component of the proposed method. In this paper, we incorporate five similarity functions, which include Vector Space Similarity (VSS), Gaussian Kernel (GK), Pearson Correlation Coefficient (PCC), Dynamic Time Warping (DTW) and Constant Function (CF).

VSS is applied to measure the similarity between two sensors i and c:

$$H_{VSS}(i,c) = \frac{\sum\limits_{j \in \boldsymbol{o}_i \cap \boldsymbol{o}_c} X_{ij} \cdot X_{cj}}{\sqrt{\sum\limits_{j \in \boldsymbol{o}_i \cap \boldsymbol{o}_c} X_{ij}^2}\sqrt{\sum\limits_{j \in O_i \cap O_c} X_{cj}^2}}, \tag{20}$$

where $\boldsymbol{o}_i$ and $\boldsymbol{o}_c$ is the subset of $\boldsymbol{x}_i$ and $\boldsymbol{x}_c$. The entities in $\boldsymbol{o}_i$ and $\boldsymbol{o}_c$ are observed. From Equation (20), we know that a larger value of H_{VSS} means that sensors i and c are more similar.

Another way to measure the similarity between two sensors i and c is based on Gaussian Kernel:

$$H_{GK}(i,c) = exp(-\frac{\sum\limits_{j \in \boldsymbol{o}_i \cap \boldsymbol{o}_c} (X_{ij} - X_{cj})^2}{2\sigma^2}). \tag{21}$$

Similarly, the more similar two sensors are, the larger the value of H_{GK} will be.

However, the above two functions do not take the different scales between two sensors into consideration. For example, the value detected by the light sensor might be much larger than that of the humidity sensor. Thus, another commonly used function PCC is employed to solve the problem, which is calculated as follows:

$$H_{PCC}(i,c) = \frac{\sum\limits_{j \in \boldsymbol{o}_i \cap \boldsymbol{o}_c} (X_{ij} - \bar{X}_i) \cdot (X_{cj} - \bar{X}_c)}{\sqrt{\sum\limits_{j \in \boldsymbol{o}_i \cap \boldsymbol{o}_c} (X_{ij} - \bar{X}_i)^2}\sqrt{\sum\limits_{j \in \boldsymbol{o}_i \cap \boldsymbol{o}_c} (X_{cj} - \bar{X}_c)^2}}, \tag{22}$$

where $\bar{X}_i$ and $\bar{X}_c$ are the average values of $\boldsymbol{o}_i$ and $\boldsymbol{o}_c$ respectively.

Actually, the proposed three similarity functions usually require that the length of $\boldsymbol{o}_i$ is equal to that of $\boldsymbol{o}_c$. Thus the functions only take the samples observed in both $\boldsymbol{x}_i$ and $\boldsymbol{x}_c$ into computing. As a consequence, they did not make full use of the observed entities in the time series and might lose important information of the raw data set. DTW is a well-known technique to compare two time series with different length. It aims at aligning two time series by warping the time axis iteratively until an optimal match between the two sequences is found. The strategy is to find a warping path W that minimize the warping cost:

$$DTW(\boldsymbol{o}_i, \boldsymbol{o}_c) = \min\sqrt{\sum_{p=1}^{p=P} w_p}, \tag{23}$$

where $w_1, w_2, \ldots, w_P = W$. This path can be found using dynamic programming to evaluate the following recurrence which defines the cumulative distance $\gamma(i,c)$ as the distance $d(o_{i,j_i}, o_{i,j_c})$ and the minimum of the cumulative distances of the adjacent elements:

$$\begin{aligned}\gamma(i,c) &= d(o_{i,j_i}, o_{i,j_c}) \\ &+ min\{\gamma(i-1,c), \gamma(i,c-1), \gamma(i-1,c-1)\},\end{aligned} \tag{24}$$

where o_{i,j_i} and o_{c,j_c} denotes the j_ith and j_cth elements in $\boldsymbol{o}_i$ and $\boldsymbol{o}_c$ respectively. This review of DTW is necessarily brief, and the details could be found in [19]. To make it consistent that a larger value of H means that sensors i and c are more correlated, the reciprocal of DTW is employed as the similarity function:

$$H_{DTW}(i,c) = \frac{1}{DTW(\boldsymbol{o}_i, \boldsymbol{o}_c)}. \tag{25}$$

Furthermore, to better reveal the necessity of incorporating similarity functions, a constant function

$$H_{CF}(i,c) = C \tag{26}$$

is also employed as the baseline function in the paper.

D. Integration of Temporal Smoothness of Time Series and Information across Multiple Sources

The above proposed CSM, USM and MFS models aim at taking advantage of either the information across multiple sources or the temporal smoothness of time series. Naturally, it is convincing that the combined fusion of the two characteristics of the multivariable time series can also contribute to improving the performance of missing data prediction. So, we also propose the following two models: Correlated Sensors based Matrix factorization with Smoothness constraints (CSMS) and Uncorrelated Sensors based Matrix factorization with Smoothness constraints (USMS).

The objective function of CSMS is:

$$\begin{aligned}\min_{S,V} \mathcal{L}_{SV}(X,S,V) &= \frac{1}{2}\|W \circ (X - SV^T)\|_F^2 \\ &+ \frac{\lambda_1}{2}\|S\|_F^2 + \frac{\lambda_2}{2}\|V\|_F^2 \\ &+ \frac{\alpha}{2}\sum_{i=1}^{N}\|S_i - \frac{1}{|C(i)|}\sum_{c \in C(i)} \rho_{i,c} S_c\|_F^2 + \frac{\beta}{2}\|GV^T\|_F^2,\end{aligned} \tag{27}$$

The objective function of USMS is:

$$\begin{aligned}\min_{S,V} \mathcal{L}'_{SV}(X,S,V) &= \frac{1}{2}\|W \circ (X - SV^T)\|_F^2 \\ &+ \frac{\lambda_1}{2}\|S\|_F^2 + \frac{\lambda_2}{2}\|V\|_F^2 \\ &- \frac{\alpha'}{2}\sum_{i=1}^{N}\|S_i - \frac{\sum\limits_{c' \in C'(i)} H(i,c') * \rho_{i,c'} S_{c'}}{\sum\limits_{c' \in C'(i)} H(i,c')}\|_F^2 + \frac{\beta}{2}\|GV^T\|_F^2.\end{aligned} \tag{28}$$

The solution of the above two objective functions is similar to that of MFS and is not included here due to the limited space.

E. Implementation of the Proposed Methods on Apache Spark

The scale of modern time series data sets is rapidly growing. And there is an imperative need to develop solutions to harness this wealth of data using statistical methods. Spark is a distributed computing framework developed at UC Berkeley AMPLab. Spark's in-memory parallel execution model in which all data will be loaded into memory to avoid the I/O bottleneck benefits the iterative computation [20]. Spark also provides very flexible DAG-based (directed acyclic graph) data flows, which can significantly speedup the computation of the iterative algorithms. The two features of Spark bring performance up to 100 times faster compared to Hadoop's two-stage MapReduce paradigm.

Here, we implement our proposed methods on Apache Spark platform. To make the solutions more adaptable to the platform, the gradients of the objective functions are rewritten in matrix form. Taking the MFS model for an example, the gradients of $\mathcal{L}_V(X,S,V)$ with respect to S and V could be calculated as:

$$\frac{\partial \mathcal{L}_V}{\partial S} = W \circ (SV^T - X)V + \lambda_1 S, \tag{29}$$

$$\frac{\partial \mathcal{L}_V}{\partial V} = W \circ (SV^T - X)^T S + \lambda_1 V + \beta V G^T G \tag{30}$$

As the CSR and USR regularization terms are hardly implemented in matrix form, the solution of CSMS's and USMS's objective functions is slightly different from that of MFS. Taking the USMS model for an example, given that the correlated sensors based regularization term only exerts effect on the gradient of $\mathcal{L}'_{SV}(X,S,V)$ with respect to S_i, we divide the gradient computation into two steps. First, the matrix product is computed according to Equation (29). Then, the gradient could be simply summed by:

$$\frac{\partial \mathcal{L}'_{SV}}{\partial S_i} = [\frac{\partial \mathcal{L}_V}{\partial S}]_i - \alpha'(S_i - \frac{\sum\limits_{c' \in C'(i)} H(i,c') * \rho_{i,c'} S_{c'}}{\sum\limits_{c' \in C'(i)} H(i,c')}), \tag{31}$$

for all i $\in \{1, 2, \ldots, N\}$, where $[\cdot]_i$ represents the ith row of the matrix.

F. Overall Algorithm

Algorithm 1 : MFS for missing data prediction in multivariable time series

```
Input:  multivariable time series $X$, indicator matrix $W$;
        dimension of latent factors $L$;
        parameters $\alpha$, $\lambda$, $|C(i)|$;
Output: $\hat{X}$: the predicted values of $X$
repeat
    $\gamma$ = computing the best step size;
    for $i = 1$ to $N$ do
        $S_i$ = $S_i$ - $\gamma$ $\frac{\partial \mathcal{L}_V}{\partial S_i}$          ▷ based on Equation (10)
    end for
    for $j = 1$ to $M$ do
        $V_j$ = $V_j$ - $\gamma$ $\frac{\partial \mathcal{L}_V}{\partial V_j}$          ▷ based on Equation (11)
    end for
until Convergence
Predicted $\hat{X} = SV^T$
```

Putting everything together, we have the overall algorithm based on MFS for solving the problem illustrated in Equation (14). As Algorithm 1 shows, given multivariable time series X, the dimension of latent factor L, the parameters α, λ, and $|C(i)|$, the algorithm is designed to obtain a more accurate solution of the factors S and V. The algorithm updates S and V until convergence, and the step size γ is updated in each iteration based on the line search strategy. The missing values

Algorithm 2 : USMS implemented on Apache Spark

```
Input: data path $dataPath$;
Output: $\hat{X}$: the predicted values of $X$
repeat
    $rddX, rddW \leftarrow$ SparkContext.textFile($dataPath$)
                                                    ▷ X and W
    initialize the parameters;        ▷ S, V, L, $\lambda$, $\beta$, and $\gamma$
    $rowMatrixX \leftarrow$ new RowMatrix($rddX$)
    calculate $\frac{\partial \mathcal{L}_V}{\partial S}$ and $\frac{\partial \mathcal{L}_V}{\partial V}$    ▷ based on Equation (29)
  and Equation (30)          ▷ using RowMatrix.multiply
    $S$ = $S$.map{ $S_i$ - $\gamma$ $\frac{\partial \mathcal{L}'_{SV}}{\partial S_i}$ }          ▷ updating S
    $V$ = $V$ - $\gamma$ $\frac{\partial \mathcal{L}'_{SV}}{\partial V}$ = $V$ - $\gamma$ $\frac{\partial \mathcal{L}_V}{\partial V}$          ▷ updating V
until Convergence
Predicted
$rowMatrixS \leftarrow$ new RowMatrix($rddS$)
$\hat{X} = rowMatrixS$.multiply($V^T$).collected()
```

could be obtained from the predicted $\hat{X}$. The algorithms of CSM, USM, CSMS, and USMS are similar with Algorithm 1, so they are not included here due to the limited space.

Next, we also show one representative algorithm, i.e. USMS, on Apache Spark. As Algorithm 2 shows, the input data X and W are first transformed to resilient distributed data set(RDD), i.e. $rddX$ and $rddW$, respectively, which is a new distributed memory abstraction in Spark. Then, to implement the matrix multiplication in Spark, the $rddX$ is transformed as RowMatrix so that it could be multiplied by a local matrix, such as V^T. Likewise, the matrix product in Equation (29) and Equation (30) could be obtained. Next, S_i and V are updated by the calculated gradients until convergence. Finally, the predicted $\hat{X}$ is obtained by performing RowMatrix multiplication one more time.

IV. Experiments

In this section, to demonstrate the effectiveness of the proposed methods, we conduct extensive experiments on three real-world data sets and one synthetic data set.

A. Data Set Description

The details about the four data sets are summarized in Table II. The data sets consists of two small scale data sets and two large scale data sets.

Motes Data Set: The motes data set contains temperature time series collected from 54 sensors deployed in the Intel Berkeley Research lab in about one month [21]. Each time series are collected once every 31 seconds. In the experiment, the length of each time series is 14000.

Sea-Surface Temperature Data Set: The Sea-Surface Temperature (SST) data set consists of hourly temperature measurements from Tropical Atmosphere Ocean Project [22]. In the experiment, the length of each time series is 18000.

Gas Sensor Array Data Set: Gas Sensor Array under dynamic gas mixtures (GSA) data set, the other large scale data set, was collected in a gas delivery platform facility at the ChemoSignals Laboratory in the BioCircuits Institute, University of California San Diego [23]. GSA contains the acquired time series from 16 chemical sensors exposed to Ethylene in air at varying concentration levels. Each measurement was constructed by the continuous acquisition of the 16-sensor array signals for a duration of about 12 hours without interruption. In the experiment, the length of each time series is 1.5E6.

Synthetic Data Set: Synthetic (SYN) data set, a large scale data set, is generated by $Asin(\omega y) + cons + noise$, where $A > 0$ denotes the amplitude of sinusoidal function, ω is the angular frequency, $cons$ represents a non-zero center amplitude and $noise \sim N(0, 1)$ is an additive Gaussian noise. In the experiment, the parameters are set as $A \in \{2, 2.5, 3\}$, $con \in \{2, 3, 5\}$, $\omega \in \{1, \pi, 2\pi\}$ and the length of SYN is 1E6.

B. Experimental Setup

As Table II shows, the samples are partitioned over 10 folds: 9 folds as the training set and the remaining 1 fold as the test set. As the time series show special temporal characteristic, we randomly split the data sets. For fair comparison with the baseline methods, the experiments are conducted with the same parameters when we evaluate the performance of the proposed method with different missing ratios. In addition, when we conduct the experiments on one specific parameter, the other parameters remain unchanged and the missing ratio is equal to 0.1.

To evaluate all the methods fairly, we incrementally simulate the data missing of the four data sets with an increasing missing ratio. For example, to increase the missing ratio from 0.80 to 0.90, we randomly move 10% of the total data from the observed data set to the missing data set. In this way, the subsequent missing data set always contains the missing data of the previous one. The missing data prediction of testing data sets is based on the known 10% of available values in the training set.

From Equation (14) and (17), we know that the constant value C will not change the value of the equations. Thus, C could be simply set as 1. Besides, the parameters λ_1 and λ_2 are both set equal to λ in this paper. For the MFS, CSMS, and USMS models, which incorporate the temporal smoothness constraints, the 2nd derivative approximation matrix G_2 is employed in the regularization terms.

The algorithm stops when the change of the cost of two latest iterations is lower than a threshold value (1E-7). The line search strategy involved in our methods selects the step size and the step direction simultaneously, which provides values that help to converge to the absolute minimum of the loss function.

Parallel computing experiments are carried out in a cluster of four working machines based on the same experimental setup given above. As there is no reasonably significance in implementing parallel experiments with small or medium data sets, these experiments are only conducted based on the two large scale data sets, i.e., GSA and SYN data sets. The working nodes are virtual machines (VM) and each of them has two cores with an Intel 2400 CPU and 4G memory. The operating system for the cluster is CentOS 7, while the version of Apache Spark platform is 1.4.0 and the Hadoop platform is version 2.6.0.

TABLE II
STATISTICS FOR THE FOUR DATA SETS IN THE EXPERIMENT, SHOWING THE TOTAL NUMBER OF SENSORS, THE TOTAL LENGTH OF SAMPLES IN THE TIME SERIES, AND THE NUMBER OF TRAINING AND TEST DATA SET.

	Property				Experiment	
Data set	#sensors	#time	mean	standard variance	#Train	#Test
MO	54	14000	21.24	10.78	12600	1400
SST	11	18000	19.1	8.97	16200	1800
GSA	16	1.5E6	4.46	2.37	1.35E6	1.5E5
SYN	27	1E6	3.4	3.2	9E5	1E5

MO: Motes data set. SST: Sea-Surface Temperature Data Set. GSA: Gas Sensor Array Data Set. SYN: Synthetic data set.

1) Comparison Methods

The baseline methods are selectively chosen based on their popularity and effectiveness in building prediction models. We compare the proposed methods with these baseline methods in predicting the values of the missing samples in multivariable time series. The comparison methods used in the experiment include:

- **Linear Interpolation**: LI uses the mean value of two nearest neighbors of the missing entries to predict the missing values.
- **AutoregRessive Integrated Moving Average**: An ARIMA model is fitted to time series data either to better understand the data in the time series, which could be applied to estimate the missing values [24].
- **Non-negative Matrix Factorization**: NMF is originally proposed for image processing. However, it is commonly used in collaborative filtering recently, which is an alternative method to address the problem of missing data prediction [25].
- **Probabilistic Matrix Factorization**: PMF is another method to address the missing data prediction problem of multivariable time series [26].
- **Bayesian PMF**: BPMF is a fully Bayesian treatment of PMF, which is more appropriate for predicting the missing data with large missing ratios [27].
- **Support Vector Machine**: SVM approach builds regression models based on each source in the sensor network respectively [28].

2) Evaluation Method

To evaluate the performance of the proposed method, root mean squared error (RMSE) is used to measure the prediction quality. RMSE is defined as follows:

$$RMSE = \sqrt{\frac{\sum\limits_{i,j}(1 - W_{ij})(X_{ij} - \hat{X}_{ij})^2}{\sum\limits_{i,j}(1 - W_{ij})}}, \qquad (32)$$

where X_{ij} is the raw time series matrix and $\hat{X}_{ij}$ is the corresponding predicted value. W is the indicator value which is defined in Equation (1).

C. Experimental Results

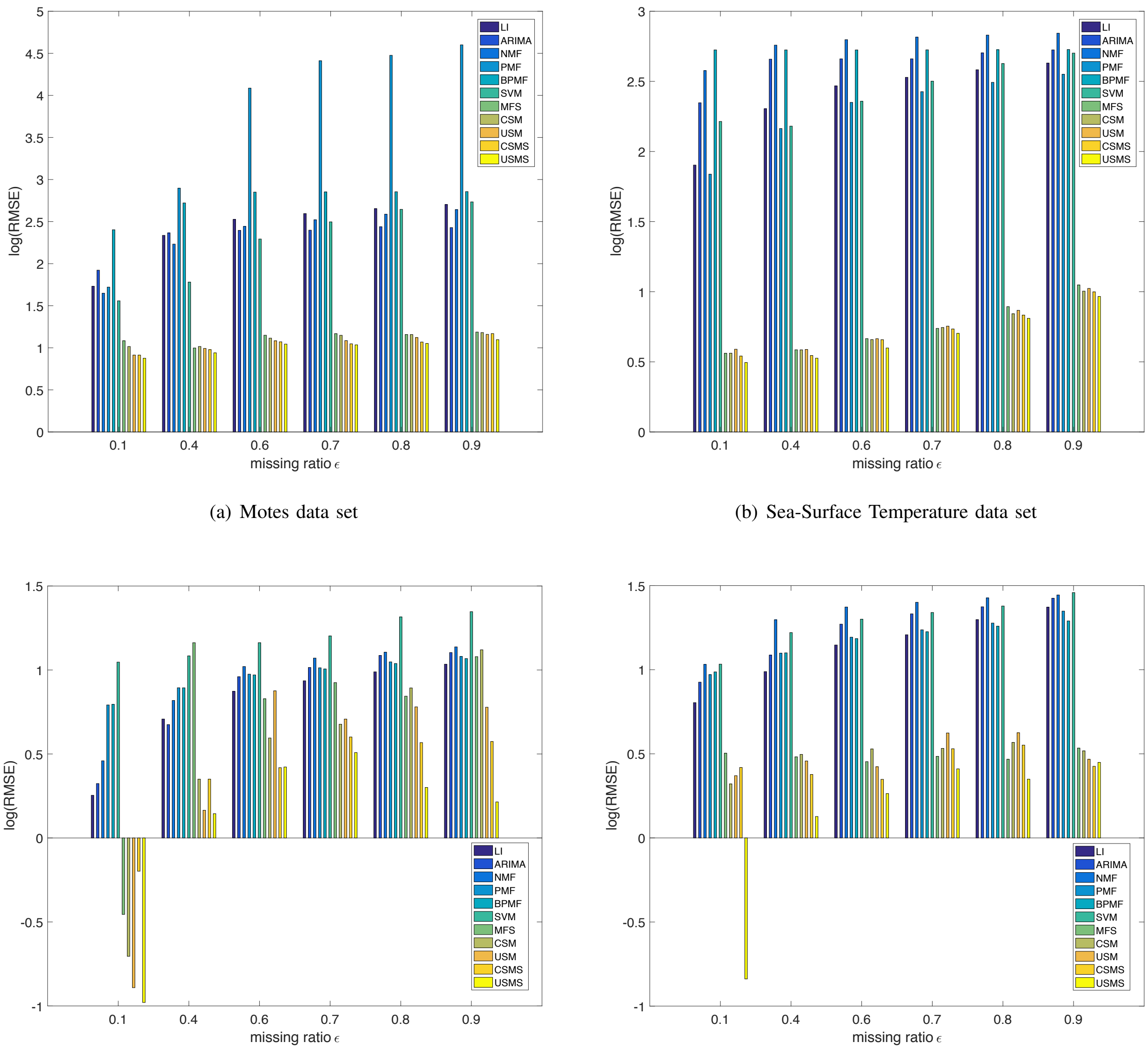
Fig. 2 shows the experimental results of the proposed methods and baseline methods on the MO, SST, GSA and SYN data sets. The logarithm of RMSE is shown in vertical axis to present the experimental results more clearly.

First, in general, the proposed five models show much better performance than the baseline methods, which demonstrates that the proposed matrix factorization methods based on fusing the temporal smoothness of time series and the information across multiple sources are suitable and effective to predict the missing values in multivariable time series. Concretely, let's consider the first model MFS. We can see that MFS consistently outperforms the other baseline methods. Quantitatively, as for the Motes data set, when the missing ratio ϵ is equal to 0.4, MFS achieves the lowest RMSE 2.76, which is about 84% lower than PMF. Even when the missing ratio exceeds 0.6, the RMSE of MFS is still within a reasonable range. As for the other three data sets, the proposed method MFS also outperforms other baseline methods, which show barely satisfactory results even when the missing ratio is as low as 0.1.

Moreover, given the models CSM and USM, we can observe that the RMSE of USM is generally a little bit lower than CSM for the Motes and SYN data sets. However, as for the SST data set, the performance of USM is not as good as that of CSM. As Table II shows, the Motes and SYN data set generates from 54 and 27 sensors respectively. As a result, these two data sets have a much higher chance of containing uncorrelated sensors. Thus USM shows better performance as it employs an USR constraint, i.e., uncorrelated sensors based regularization term. The experimental results of GSA data set in Fig 2(c) further demonstrate the rationality of choosing USM when the number of sensors is relatively high. Contrarily, the SST data set is collected from only 11 sensors. So it is much more important for the SST data set to find the correlated sensors, thus CSM shows superior performance. Nevertheless, as CSM and USM show better performance than the baseline methods, they are both alternative models in the proposed methods.

The prominent superiority of MFS, CSM and USM to NMF, PMF and BPMF reveals that the smoothness, CSR and USR constraints can largely contribute to the latent factors extraction in the process of matrix factorization, which further demonstrates the effectiveness of the integration of both information across multiple sources and temporal smoothness of time series.

Finally, as expected, the CSMS and USMS models generally show even better performance than MFS, CSM and USM, as both of them are built by integrating the information across



(a) Motes data set

(b) Sea-Surface Temperature data set

(c) Gas Sensor Array under dynamic gas mixtures data set

(d) Synthetic data set

Fig. 2. Missing data prediction performance comparison between proposed methods and baseline methods.

multiple sources and temporal smoothness of time series together. Note that USMS is consistently superior to CSMS. For example, when the missing ratio reaches as high as 0.9, the RMSE of USMS is only 1.24, which is roughly 58% lower than the proposed MFS model, 60% lower than CSM, and about 8.8% than CSMS for the SYN data set. We deduce that, when the smoothness constraint is combined with CSR and USR in the USMS model, the USR constraint can be more effective than CSR in the process of matrix factorization by removing uncorrelated information, while CSMS aims at retaining the most valuable information of the raw data set.

D. Similarity Functions Impact Discussion

The similarity function H aims at finding the set of correlated sensors $C(i)$ or the set of uncorrelated sensors $C'(i)$. H directly determines which sensors are correlated or uncorrelated with the ith sensor and the weights of the sensor network regularization terms. Thus, we mainly focus on the analysis of the similarity functions in this subsection. Due to the lack of space, we only give the impact discussion of the similarity functions in the fifth model USMS as it shows the best performance in the five models. Similar results are observed for the other models.

As Table III shows, when the missing ratio is below 0.6, PCC obtains lower RMSE for all of the four data sets. As Equation (22) reveals, PCC takes the different scales among various sensors into consideration, which might contribute to its better performance. Moreover, DTW achieves far superior performance to the other functions when the missing ratio exceeds 0.6. We deduce that DTW can better measure the similarity between two time series when the missing ratio is high, as it utilizes all the observed entities in the raw

TABLE III
PERFORMANCE OF USMS WITH DIFFERENT SIMILARITY FUNCTIONS AND MISSING RATIO ϵ.

	ϵ	VSS	GK	PCC	DTW	CF
MO	0.1	2.98	2.99	**2.40**	2.42	2.48
	0.4	2.56	2.62	**2.54**	2.57	2.61
	0.6	2.98	2.89	**2.84**	2.94	2.92
	0.7	2.94	2.99	2.92	**2.81**	3.05
	0.8	3.09	3.07	2.88	**2.86**	2.99
	0.9	3.03	3.10	2.98	**2.94**	3.02
SST	0.1	1.81	1.91	**1.74**	1.79	1.90
	0.4	1.93	1.88	**1.79**	1.85	1.84
	0.6	1.94	1.93	**1.91**	1.95	1.97
	0.7	2.19	2.12	2.15	**2.05**	2.13
	0.8	2.42	2.35	2.35	**2.28**	2.40
	0.9	2.85	2.71	2.73	**2.62**	2.75
GSA	0.1	0.79	0.83	**0.63**	-	0.78
	0.4	1.17	1.18	**1.14**	-	1.17
	0.6	1.37	1.39	**1.30**	-	1.39
	0.7	1.68	1.66	**1.51**	-	1.68
	0.8	1.59	1.53	**1.42**	-	1.54
	0.9	1.57	1.58	**1.63**	-	1.59
SYN	0.1	3.78	3.02	**0.37**	-	2.16
	0.4	1.26	2.45	**1.15**	-	1.38
	0.6	1.65	1.67	**1.52**	-	1.59
	0.7	1.92	2.65	**1.66**	-	1.82
	0.8	1.95	2.07	**1.35**	-	2.18
	0.9	2.27	4.11	**1.23**	-	2.40

MO: Motes data set. SST: Sea-Surface Temperature Data Set. GSA: Gas Sensor Array under dynamic gas mixtures data set. SYN: Synthetic data set.

time series. However, for the large scale data sets GSA and SYN, DTW runs out of memory in the system. Nevertheless, both PCC and DTW are alternative similarity functions in the proposed method. In addition, we observe that the constant function CF shows barely satisfactory results, which further demonstrates the necessity and importance of employing an appropriate similarity function.

E. Parameters Impact Discussion

In this subsection, we also only give the analyses of the parameters of the fifth model USMS. Fig. 3 shows the impact of the parameters on the performance of USMS.

In general, the RMSE of USMS with different parameters is universally below a reasonable value and shows acceptable stability, despite the fact that there is a little bit variation with various parameter as shown in the figure.

Concretely, first, the parameter $|C(i)|$ denotes the total number of the correlated sensors with the ith sensor, which plays a very important role in the proposed method. Taking the Motes data set for an example, when $|C(i)|$ is set as 4, the RMSE is equal to 2.68. However, when $|C(i)|$ is equal to 7, the method achieves the lowest RMSE 2.40, which is reduced by about 11%. We deduce that an oversized $|C(i)|$ will bring in more noise while too small a $|C(i)|$ will be not enough to constrain the matrix factorization. Thus, an appropriate value of $|C(i)|$ is of great importance in the proposed method.

Then, the impact of the dimension of the latent factors L on the performance is also shown in the figure. On the whole, the optimal RMSE is consistently very small. Specifically, take the SST data set for instance, the RMSE is equal to 2.03 when L is set as 10, while the lowest RMSE 1.74 is obtained when L is equal to 6. Nevertheless, based on the experimental results, we may safely set $L = 11$, $L = 6$, $L = 4$ and $L = 4$ for the four data sets respectively. Hence, the dimension of the latent factors L also plays an important part in the proposed method.

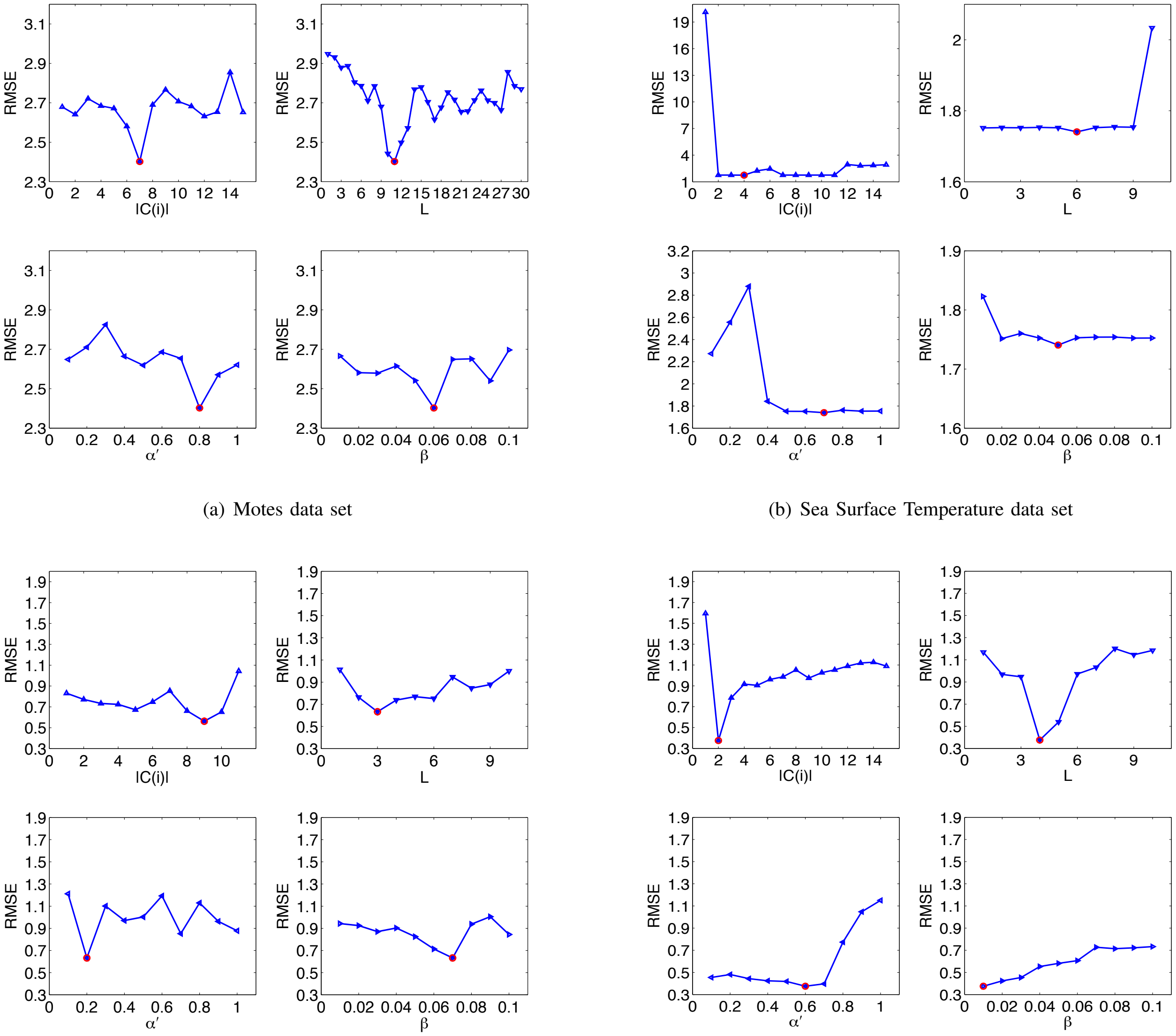
Next, the impact of α' on the performance is presented. α' controls how much information of the sensor network should be incorporated into the optimization problem. In general, as the Fig. 3 shows, the RMSE not only is consistently very low but also shows little variation for most of the different α' values. We can observe that the best performance is achieved when α' is equal to 0.8, 0.7, 0.6 and 0.2 for the four data sets respectively. We deduce that too small an α' would greatly decrease the influence of the sensor regularization term on the matrix factorization. On the other hand, if we employ too large an α', the sensor regularization term would dominate the learning processes. So, an appropriate coefficient α' could further improve the performance of the proposed method.

Finally, the coefficient β is optimized. As the figure shows, it suffices to say that the model USMS generally achieves good stability with β, although Fig. 3(a) shows a slight fluctuation with the parameter β. Based on the experimental results, we can reasonably set $\beta = 0.06$, $\beta = 0.05$ $\beta = 0.01$and $\beta = 0.07$ for the four data sets respectively.

F. Evaluation under Parallel Environment

Since the superior performance of the proposed methods is obtained, which means that the proposed models can effectively predict the missing values in multivariable time series, we now turn to evaluate performance of the methods when it comes to dealing with the case of big data. As the USMS model shows best performance among the proposed five models, we focus on the evaluation of USMS under parallel environment. Similar results are observed for the other proposed models. In order to show the scalability of USMS, we also perform experiments by a stand-alone computer on Spark platform, which has four cores from an Intel i7 and 8G memory. All the baseline methods are conducted under Matlab version 2012b by the same stand-alone computer.

First, Fig. 4(a) and Fig. 4(c) shows the execution time comparison between different methods, which includes both the training time and the testing time. Here, we conduct experiments on the two large scale data sets GSA and SYN. And the size of the two data sets is 1.5E6 and 1E6 respectively. As the SVM baseline method takes more than 12 hours, which is much longer than the other methods, it is not included in the figure. We can observe that LI obtains the least execution time due to its simple computation method. However, the prediction accuracy of LI is actually unsatisfactory. Nevertheless, among the other methods, the proposed model USMS takes relatively very little time for both of data sets while ensuring high prediction performance.



(a) Motes data set

(b) Sea Surface Temperature data set

(c) Gas Sensor Array under dynamic gas mixtures data set

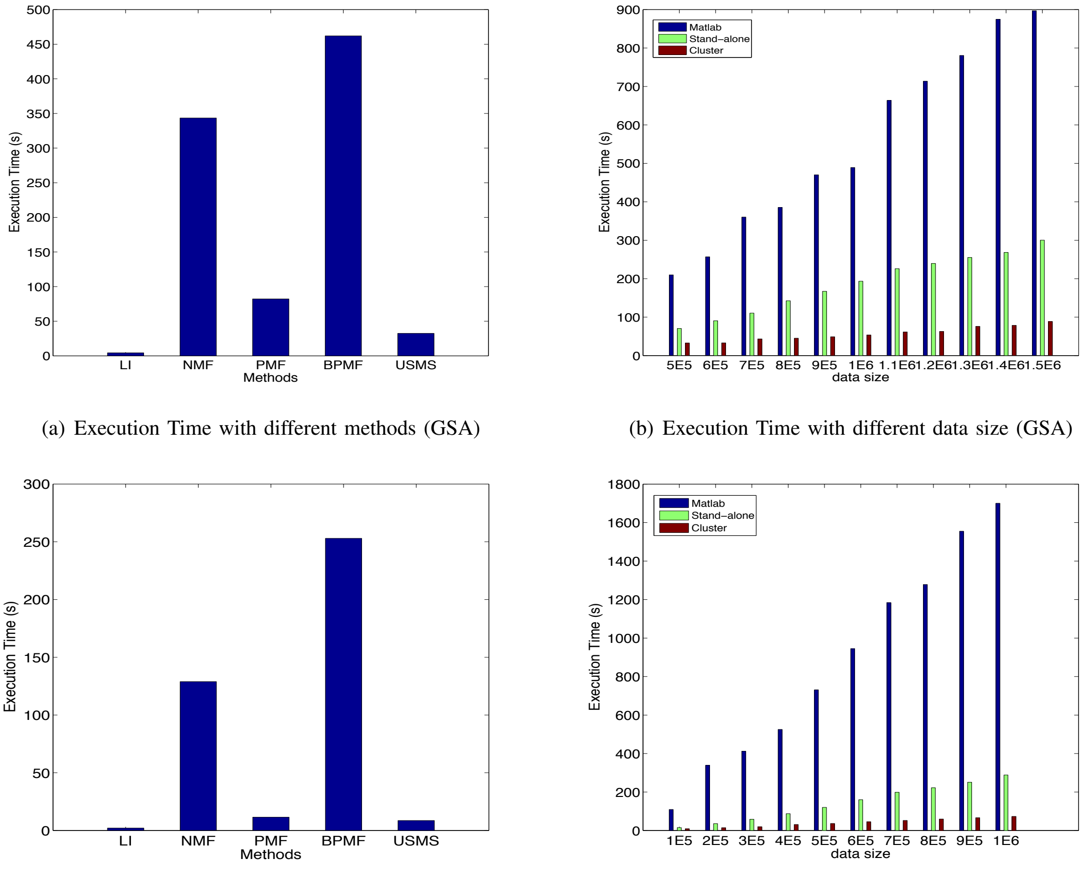
(d) Synthetic data set

Fig. 3. Impact of Parameters.

Moreover, we also conduct the experiments on the execution time of USMS with different size of GSA and SYN. Fig 4(d) and Fig 4(b) shows the execution time of USMS under different operating environment, which includes Matlab, Apache Spark with a stand-alone computer, Apache Spark cluster. We can observe that the USMS under parallel environment shows satisfactory scalability as it can deeply reduce the execution time. For example, when the size of GSA data set is equal to one million, USMS only takes 53 seconds, which is roughly 16 times faster than Matlab and 2.3 times faster than stand-alone computer. Thus we may conclude that the propose methods can be effective models for predicting the missing values in large scale multivariable time series.

V. Related Works

Missing data prediction: The prediction of missing data are pervasive problems in machine learning and statistical data analysis. Salakhutdinov et al propose a PMF (Probabilistic Matrix Factorization) method [26]. The method is aimed at improving the prediction accuracy of the recommender system. As the multivariable systems hold many internal characteristics with the recommender system, PMF could not be effectively applied in our scenario. Ma et al. propose a missing data prediction algorithm for collaborative filtering. Their approach determines whether to predict the missing data and how to predict the missing data by using information of users and items by judging whether a user (an item) has other correlated users (items) [29]. The problem is similar to ours, but the proposed method is mainly focused on incorporating the information of social network, which is very different from our sensor networks. Asif et al. [30] propose methods which can construct low-dimensional representation of large and diverse networks, in presence of missing historical and neighboring data to overcome data missing problems in an



(a) Execution Time with different methods (GSA)

(b) Execution Time with different data size (GSA)

(c) Execution Time with different methods (SYN)

(d) Execution Time with different data size (SYN)

Fig. 4. Execution Time Evaluation.

urban road network. The key idea of this method is to extract important information from large amount of data. However, the proposed method also cannot be directly applied to our scenario. Faloutsos et al propose Dynamic Bayesian Network. Their main idea is to simultaneously exploit smoothness and correlation. Their method yields results with satisfactory reconstruction error. But they solve it using probabilitic graph model, which might be inefficient when the data size is large [31]. In our preliminary work [28], we simply propose an optimized linear regression (OLR) method to predict the missing values. However, when the missing ratio is too high, OLR might lose its effectiveness. Besides, since OLR is an improve method of linear regression, it aims at dealing with the time series data sets with great smoothness.

Time Series Mining: There has been a great deal of research work in time series mining in various areas [32], [4], [33]. In the economic domain, the economic time series could be utilized to discover the nature of economic [34]. Energy time series and climate time series analysis shows profound significance in constructing sustainable development of the natural environment [35]. In the study of genetics, time series mining is also a powerful tool to discover the principles of gene [36]. In industrial production, chemical plant time series are used to monitor an entire manufacture process of a chemical plant [37].

VI. Conclusion

In this paper, we have proposed novel methods to constrain the matrix factorization for predicting the missing data in the time series from multiple sources, which achieve satisfactory performance of missing data prediction and high computing efficiency. The methods aim at fusing the smoothness characteristic of each time series and valuable correlation information across multiple sources in a sensor network into matrix factorization. Correspondingly, the methods incorporate smoothness, CSR and USR constraints to optimize the solution of matrix factorization. Based on the idea, we proposed five effective models. The prominent superiority of MFS, CSM and USM reveals the effectiveness of latent factors extraction in the process of matrix factorization after incorporating the constraints. Furthermore, the combination of information extraction across multiple sources and temporal smoothness of each time series demonstrate the effectiveness of the proposed methods. Even when the missing ratio is as high as 90%, the RMSE of the proposed methods is still within reasonable range. Finally, the experiments under parallel environment reveal that the USMS model can be executed effectively. We conclude that the proposed methods are alternative models for predicting the missing values in large scale multivariable time series.

Acknowledgment

This paper is sponsored in part by the National High Technology and Research Development Program of China (863 Program, 2015AA050204), State Grid Science and Technology Project (520626140020, 14H100000552, SGC-QDK00PJJS1400020), State Grid Corporation of China, National Natural Science Foundation of China (No.61373032), NSF through grants IIS-1526499, CNS-1626432, and NSFC 61672313.

References

[1] H. Nguyen, W. Liu, F. Chen, Discovering congestion propagation patterns in spatio-temporal traffic data, IEEE Transactions on Big Data PP (99) (2016) 1–1.

[2] Y. Cai, H. Tong, W. Fan, P. Ji, Fast mining of a network of coevolving time series, in: Proceedings of the 2015 SIAM International Conference on Data Mining, pp. 298–306. arXiv:http://epubs.siam.org/doi/pdf/10.1137/1.9781611974010.34, doi:10.1137/1.9781611974010.34.

[3] H.-V. Nguyen, J. Vreeken, Linear-time detection of non-linear changes in massively high dimensional time series, in: Proceedings of the SIAM International Conference on Data Mining (SDM'16), 2016.

[4] N. Méger, C. Rigotti, C. Pothier, Swap randomization of bases of sequences for mining satellite image times series, in: Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD), Springer, 2015, pp. 190–205.

[5] W. Shi, Y. Zhu, T. Huang, G. Sheng, Y. Lian, G. Wang, Y. Chen, An integrated data preprocessing framework based on apache spark for fault diagnosis of power grid equipment, Journal of Signal Processing Systems 82 (2016) 1–16.

[6] R. Istepanian, S. Hu, N. Philip, A. Sungoor, The potential of internet of m-health things "miot" for non-invasive glucose level sensing, in: 2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2011, pp. 5264–5266. doi:10.1109/IEMBS.2011.6091302.

[7] S.-F. Wu, C.-Y. Chang, S.-J. Lee, Time series forecasting with missing values, in: 2015 1st International Conference on Industrial Networks and Intelligent Systems (INISCom), 2015, pp. 151–156.

[8] W. Lao, Y. Wang, C. Peng, C. Ye, Y. Zhang, Time series forecasting via weighted combination of trend and seasonality respectively with linearly declining increments and multiple sine functions, in: 2014 International Joint Conference on Neural Networks (IJCNN), 2014, pp. 832–837.

[9] M. Lippi, M. Bertini, P. Frasconi, Short-term traffic flow forecasting: An experimental comparison of time-series analysis and supervised learning, IEEE Transactions on Intelligent Transportation Systems 14 (2) (2013) 871–882.

[10] P. Baraldi, F. D. Maio, D. Genini, E. Zio, Reconstruction of missing data in multidimensional time series by fuzzy similarity, Applied Soft Computing 26 (215) 1 –9. doi:http://dx.doi.org/10.1016/j.asoc.2014.09.038.

[11] Y. Song, M. Liu, S. Tang, X. Mao, Time series matrix factorization prediction of internet traffic matrices, in: 2012 IEEE 37th Conference on Local Computer Networks (LCN), 2012, pp. 284–287.

[12] Z. Zhang, K. Barbary, F. A. Nothaft, E. R. Sparks, O. Zahn, M. J. Franklin, D. A. Patterson, S. Perlmutter, Kira: Processing astronomy imagery using big data technology, IEEE Transactions on Big Data PP (99) (2016) 1–1.

[13] M. Winlaw, M. B. Hynes, A. L. Caterini, H. D. Sterck, Algorithmic acceleration of parallel ALS for collaborative filtering: Speeding up distributed big data recommendation in spark, in: 21st IEEE International Conference on Parallel and Distributed Systems, ICPADS 2015, Melbourne, Australia, December 14-17, 2015, 2015, pp. 682–691.

[14] S. Papadimitriou, J. Sun, C. Faloutos, P. S. Yu, Dimensionality Reduction and Filtering on Time Series Sensor Streams, Springer US, Boston, MA, 2013, pp. 103–141.

[15] Y. Zhang, M. Chen, D. Huang, D. Wu, Y. Li, idoctor: Personalized and professionalized medical recommendations based on hybrid matrix factorization, Future Generation Computer Systems 66 (2017) 30 – 35. doi:https://doi.org/10.1016/j.future.2015.12.001.

[16] H. Ma, D. Zhou, C. Liu, M. R. Lyu, I. King, Recommender systems with social regularization, in: Proceedings of the Fourth ACM International Conference on Web Search and Data Mining, WSDM '11, ACM, 2011, pp. 287–296. doi:10.1145/1935826.1935877.

[17] R. Salakhutdinov, A. Mnih, Bayesian probabilistic matrix factorization using markov chain monte carlo, in: Proceedings of the 25th International Conference on Machine Learning, ICML '08, ACM, New York, NY, USA, 2008, pp. 880–887. doi:10.1145/1390156.1390267.

[18] A. Cichocki, R. Zdunek, A. H. Phan, S.-i. Amari, Nonnegative matrix and tensor factorizations: applications to exploratory multi-way data analysis and blind source separation, John Wiley & Sons, 2009.

[19] M. Müller, Dynamic time warping, Information retrieval for music and motion (2007) 69–84.

[20] N. Bharill, A. Tiwari, A. Malviya, Fuzzy based scalable clustering algorithms for handling big data using apache spark, IEEE Transactions on Big Data PP (99) (2016) 1–1.

[21] M. Samuel, Intel lab data, http://db.csail.mit.edu (2004).

[22] Noaa/pacific marine environmental laboratory (2014).

[23] J. Fonollosa, S. Sheik, R. Huerta, S. Marco, Reservoir computing compensates slow response of chemosensor arrays exposed to fast varying gas concentrations in continuous monitoring, Sensors and Actuators B: Chemical 215 (2015) 618 – 629. doi:http://dx.doi.org/10.1016/j.snb.2015.03.028.

[24] G. R. Newsham, B. J. Birt, Building-level occupancy data to improve arima-based electricity use forecasts, in: Proceedings of the 2nd ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Building, ACM, New York, NY, USA, 2010, pp. 13–18.

[25] D. D. Lee, H. S. Seung, Learning the parts of objects by non-negative matrix factorization, Nature 401 (1999) 788–791.

[26] A. Mnih, R. R. Salakhutdinov, Probabilistic matrix factorization, in: Advances in Neural Information Processing Systems, Curran Associates, Inc., 2008, pp. 1257–1264.

[27] R. Salakhutdinov, A. Mnih, Bayesian probabilistic matrix factorization using markov chain monte carlo, in: Proceedings of the 25th International Conference on Machine Learning, ICML '08, ACM, 2008, pp. 880–887. doi:10.1145/1390156.1390267.

[28] W. Shi, Y. Zhu, J. Zhang, X. Tao, G. Sheng, Y. Lian, G. Wang, Y. Chen, Improving power grid monitoring data quality: An efficient machine learning framework for missing data prediction, in: 2015 IEEE 17th International Conference on High Performance Computing and Communications, IEEE, 2015, pp. 417–422.

[29] H. Ma, I. King, M. R. Lyu, Effective missing data prediction for collaborative filtering, in: Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '07, ACM, 2007, pp. 39–46. doi:10.1145/1277741.1277751.

[30] M. Asif, N. Mitrovic, L. Garg, J. Dauwels, P. Jaillet, Low-dimensional models for missing data imputation in road networks, in: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2013, pp. 3527–3531. doi:10.1109/ICASSP.2013.6638314.

[31] L. Li, C. Faloutsos, Fast algorithms for time series mining, 2010 IEEE 26th International Conference on Data Engineering Workshops (ICDEW 2010) 00 (2010) 341–344.

[32] D. Gao, Y. Kinouchi, K. Ito, X. Zhao, Neural networks for event extraction from time series: a back propagation algorithm approach, Future Generation Computer Systems 21 (7) (2005) 1096 – 1105. doi:http://dx.doi.org/10.1016/j.future.2004.03.009.

[33] M.-Y. Chen, A high-order fuzzy time series forecasting model for internet stock trading, Future Generation Computer Systems 37 (2014) 461 – 467. doi:http://dx.doi.org/10.1016/j.future.2013.09.025.

[34] E. J. Ruiz, V. Hristidis, C. Castillo, A. Gionis, A. Jaimes, Correlating financial time series with micro-blogging activity, in: Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, WSDM '12, ACM, 2012, pp. 513–522. doi:10.1145/2124295.2124358.

[35] M. Chaouch, Clustering-based improvement of nonparametric functional time series forecasting: Application to intra-day household-level load curves, IEEE Transactions on Smart Grid 5 (1) (2014) 411–419. doi:10.1109/TSG.2013.2277171.

[36] A. Passerini, M. Lippi, P. Frasconi, Predicting metal-binding sites from protein sequence, Trans. Comput. Biol. Bioinformatics 9 (2012) 203–213.

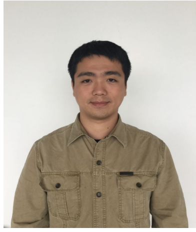
[37] X. Bian, X. Ning, G. Jiang, Hierarchical sparse dictionary learning, in: Machine Learning and Knowledge Discovery in Databases, Springer, 2015, pp. 687–700.

Weiwei Shi received his Bachelor and his M.Sc degree in College of Opto-Electronic Engineering from Nanjing University of Posts and Telecommunications, China, in 2010 and 2013, respectively. He is pursuing his Ph.D degree in the School of Electronic Information and Electrical Engineering at Shanghai Jiao Tong University. His current research interests are focused on data mining and big data.

Yongxin Zhu is an Associate Professor at Shanghai Jiao Tong University, China. He is also a visiting Associate Professor with National University of Singapore. He is a senior member of IEEE and China Computer Federation. He received his B.Eng. in EE from Hefei University of Technology, the M. Eng. in CS from Shanghai Jiao Tong University, and his Ph.D. in CS from National University of Singapore. His research interest is in computer architectures, embedded systems and system-on-chip. He has published over 100 English journal and conference papers and 30 Chinese journal papers. He has 20 China patents approved.

Philip S. Yu received the BS degree in electrical engineering from National Taiwan University, the MS and PhD degrees in electrical engineering from Stanford University, and the MBA degree from New York University. He is a distinguished professor of computer science at the University of Illinois at Chicago and holds the Wexler chair in information technology. His research interest is on big data, including data mining, data stream, database, and privacy. He has published more than 830 papers in refereed journals and conferences. He holds or has applied for more than 300 US patents. He is a fellow of the ACM and the IEEE.

Jiawei Zhang received the bachelor degrees in computer science from Nanjing University, China, in 2012. He is pursuing a PhD degree in the Department of Computer Science at the University of Illinois at Chicago. His main research areas are data mining and machine learning, especially multiple aligned social networks studies.



Tian Huang is a Research Associate with Astrophysics Group, Cavendish Laboratory, University of Cambridge. He joined the University of Cambrige in July 2016. In March 2016, he received his Ph.D. in Computer Science and Engineering from School of Microelectronics, Shanghai Jiao Tong University. In June 2008, he received his B.S. degree in Electronics Science and Technology from Shanghai Jiao Tong University. His main research interest is Data Mining for time series, including time series big data indexing, anomaly detecting, computer architecture for time series data mining and statistical models for time series data.

Chang Wang received his B.S. (2010) in computer science and technology school from Xi'an Polytechnic University, M.S. (2012) in computer science school from Wu Han University. Since September 2012, he has been pursuing his ph.d degree in the school of Electronic Information and Electrical Engineering from Shanghai Jiao Tong University, Shanghai city, China. His research interests are in the area of high performance computing focused on micro-architecture of future many-core processor and big data storage modeling.

Yufeng Chen is a senior engineer. He received his Bachelor degree from Shanghai Jiao Tong University in 1992. Now, he is the director of Evaluation Center.

View publication stats