Accurate Recovery of Internet Traffic Data: A Tensor Completion Approach

Kun Xie[1], Lele Wang[1], Xin Wang[2], Gaogang Xie[3], Jigang Wen[3], Guangxing Zhang[3]
[1] College of Computer Science and Electronics Engineering, Hunan University, Changsha, China
[2] Department of Electrical and Computer Engineering, State University of New York at Stony Brook, USA
[3] Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China
xiekun@hnu.edu.cn, wanglele2012@hnu.edu.cn, x.wang@stonybrook.edu, xie@ict.ac.cn,
wenjigang@ict.ac.cn, guangxing@ict.ac.cn

***Abstract*—The inference of traffic volume of the whole network from partial traffic measurements becomes increasingly critical for various network engineering tasks, such as traffic prediction, network optimization, and anomaly detection. Previous studies indicate that the matrix completion is a possible solution for this problem. However, as a two-dimension matrix cannot sufficiently capture the spatial-temporal features of traffic data, these approaches fail to work when the data missing ratio is high.**

To fully exploit hidden spatial-temporal structures of the traffic data, this paper models the traffic data as a 3-way traffic tensor and formulates the traffic data recovery problem as a low-rank tensor completion problem. However, the high computation complexity incurred by the conventional tensor completion algorithms prevents its practical application for the traffic data recovery. To reduce the computation cost, we propose a novel Sequential Tensor Completion algorithm (STC) which can efficiently exploit the tensor decomposition result for the previous traffic data to deduce the tensor decomposition for the current data. To the best of our knowledge, we are the first to apply the tensor to model Internet traffic data to well exploit their hidden structures and propose a sequential tensor completion algorithm to significantly speed up the traffic data recovery process. We have done extensive simulations with the real traffic trace as the input. The simulation results demonstrate that our algorithm can achieve significantly better performance compared with the literature tensor and matrix completion algorithms even when the data missing ratio is high.

***Index Terms*—Internet traffic data recovery, Tensor completion**

I. Introduction

Gaining a full knowledge of the traffic data volume between a set of origin and destination (OD) pairs in the networks becomes increasingly critical for the network engineering tasks [1], such as the prediction of future traffic trends, network optimization, protocol design, and anomaly detection.

Although important, it is impractical to collect measurement data from a very large number of points in a large network at the fine time-scales. To reduce the cost, an alternative measurement strategy usually adopted by the network monitoring system is to take random measurement samples from the full traffic data. The actual data collected can be even less due to the unavoidable data loss from the severe communication conditions. As many network engineering tasks require the complete traffic information or they are highly sensitive to the missing data, the accurate reconstruction of missing values from partial traffic measurements becomes a key problem, and we refer this problem as the traffic data recovery problem.

Various studies have been made to handle missing traffic data. As most of the known approaches are designed based on purely spatial [2]–[4] or purely temporal [5], [6] information, their data recovery performance is low. To utilize both spatial and temporal information, several recent studies model the traffic data as traffic matrices and propose matrix-based algorithms to recover the missing traffic data [7]–[12]. Although these approaches present good performance when the data missing ratio is low, their performance suffers when the missing ratio is large, especially in the extreme case when the traffic data on several time intervals are all lost.

To overcome the shortcomings of the matrix-based methods, we propose to model the traffic data based on the multi-way tensor, and design an accurate traffic recovery algorithm. Specifically, our algorithm takes advantage of the tensor pattern to combine and exploit multiple correlation features of the spatial-temporal information, which helps to preserve multiple features of the traffic data and extract the underlying factors in each feature.

Although promising, compared to matrices, tensors have additional data dimensions. Several tensor completion algorithms [13]–[16] have been proposed for the data recovery with their core lying in the tensor decomposition. Requiring a large number of computations, it is difficult to adopt the existing tensor decomposition methods in the traffic data recovery. It is important and challenging to reduce the computation cost and speed up the tensor completion process.

To design an efficient and accurate traffic data recovery algorithm, in this paper, we first analyze a large trace of real traffic data, which reveals that there exist hidden structures in the data. Structure and redundancy in data are often synonymous with the sparsity. To fully exploit theses hidden structures for the data recovery, we model the traffic data as a 3-way traffic tensor and formulate the traffic data recovery problem as a low-rank tensor completion problem. Furthermore, we propose a sequential tensor completion algorithm to quickly solve the problem with a low computation cost. To the best of our knowledge, this is the first time that the tensor pattern is introduced to model the Internet traffic data. Our model helps to preserve the multi-way nature of the traffic data and extract the underlying multi-mode hidden structures in the traffic data. Our contributions are summarized as follows:

- Based on the analysis of real traffic trace, we reveal that traffic data have the features of temporal stability, spatial correlation, and periodicity.
- To fully exploit the hidden structures for the data recovery, we model the traffic data as a 3-way traffic tensor which can combine and utilize the multi-mode correlations (for example, OD pair-mode, time-mode, and day-mode).
- To reduce the computation cost of the traffic recovery, we propose a sequential tensor completion algorithm to deduce the tensor decomposition for the current data based on the tensor decomposition result of the previous traffic data. Our algorithm does not invoke a real tensor decomposition procedure (which incurs a high computation cost) for the current data, so the computation cost can be significantly reduced.
- To evaluate the performance of our proposed algorithm, we have performed extensive simulations based on real traffic trace. Compared with the state of art tensor completion algorithms as well as the matrix-based algorithms, our algorithm can achieve significantly better performance in terms of several metrics, including the ratio of the recovery error, the ratio of the successful recovery, and the computation time.

Although we apply the tensor to capture the traffic volume in this paper, our tensor modeling and the proposed sequential tensor completion approach are useful for the representation of other factors of the network, for instance, delay, jitter, loss, bottleneck-bandwidth, and distance (RTT).

The rest of the paper is organized as follows. We introduce the related work in Section II. The preliminaries of tensor are presented in Section III. We present our analyses on the real traffic data, our system model and problem formulation, and our sequential tensor completion algorithm in Section IV, Section V, and Section VI, respectively. Finally, we evaluate the performance of the proposed algorithm through extensive simulations in Section VII, and conclude the work in Section VIII.

II. Related work

In this section, we review the related work on the recovery of the missing Internet traffic data, and identify the differences of our work from the existing work.

A set of studies have been made to handle the missing traffic data. Designed based on purely spatial [2]–[4] or purely temporal [5], [6] information, most of the known approaches have a low data recovery performance.

To capture more spatial-temporal features in the traffic data, SRMF [7] proposes the first spatio-temporal model of traffic matrices (TMs). It finds sparse approximations to TMs, and recovers the missing data with the spatio-temporal operation and local interpolation. Following SRMF, several other traffic matrix recovery algorithms [8]–[12] are proposed to recover the missing data from partial traffic measurements. Compared with the vector-based recovery approaches, as a matrix could capture more information and correlation among traffic data, matrix-based approaches achieve much better recovery performance. However, a two-dimension matrix is still limited in capturing a large variety of correlation features hidden in the traffic data. For example, although the traffic matrix defined in [7] can represent the traffic flows in different time slots to catch the spatial correlation among flows and the small-scale temporal feature, it can not incorporate other temporal features such as the feature of the traffic periodicity. Therefore, a matrix is still not enough to capture the comprehensive correlations among the traffic data, and the data recovery performance under the matrix-based approaches is still low.

It is promising to apply the emerging higher-order tensors to model the data that intrinsically have many dimensions. Tensor-based missing data recovery methods can capture the global structure of the data via a high-order decomposition (named tensor decomposition), and tensor-based methods prove to be good analytical tools for dealing with the multi-dimensional data. So far, tensor-based data recovery has been utilized in various fields (see an in-depth survey by Kolda and Bader [17]). Several tensor completion algorithms [13]–[16] are proposed for the data recovery.

The core of the tensor completion lies in the tensor decomposition, which commonly takes two forms: CANDECOMP/PARAFAC (CP) decomposition [18], [19] and Tucker decomposition [20]. In multilinear algebra, the tensor decomposition may be regarded as a generalization of the matrix singular value decomposition (SVD) to tensors. In fact, Tucker decomposition is also known as a higher-order SVD (HOSVD) [21]. As the number of elements in a tensor increases exponentially with the number of dimensions, the computational and memory requirements increase quickly, which becomes the main challenge of applying the tensor decomposition in the practical applications.

To the best of our knowledge, we are the first to apply the tensor pattern to model the Internet traffic data to well exploit the hidden structures of the traffic data, and propose an sequential tensor completion algorithm to significantly speed up the traffic data recovering process. We have performed extensive simulations with the real traffic trace as the input. The simulation results show that our sequential tensor completion algorithm can achieve highly accurate recovery performance with a short computation time.

III. Preliminaries of tensor

In this section, we introduce some basic concepts related to the tensor.

Definition 1. ***Tensor:*** *A tensor, also known as Nth-order or N-way tensor, multidimensional array, N-way or N-mode array, is a higher-order generalization of a vector (first-order tensor) and a matrix (second-order tensor), and denoted as* $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ *where N is the order of $\mathcal{A}$, also called* way or mode. The element of $\mathcal{A}$ is denoted by $a_{i_1,i_2,\cdots,i_N}, i_n \in \{1,2,\cdots,I_n\}, 1 \le n \le N$.

Definition 2. ***unfolding*** *[21]: For an Nth-order tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, the matrix unfolding $\mathbf{A}_{(\mathbf{n})} \in \mathbb{R}^{I_n \times (I_{n+1} I_{n+2} \cdots I_N I_1 I_2 \cdots I_{n-1})}$ contains the tensor element $a_{i_1,i_2,\cdots,i_N}$ at the position in the unfolding matrix with its row index i_n and column index equal to* $(i_{n+1}-1) I_{n+2} I_{n+3} \cdots I_N I_1 I_2 \cdots I_{n-1} + (i_{n+2}-1) I_{n+3} I_{n+4} \cdots I_N I_1 I_2 \cdots I_{n-1} + \cdots + (i_N - 1) I_1 I_2 \cdots I_{n-1} + (i_1 - 1) I_2 I_3 \cdots I_{n-1} + (i_2 - 1) I_3 I_4 \cdots I_{n-1} + \cdots + i_{n-1}$.

Fig. 1 shows an unfolding procedure of a 3rd-order tensor, which involves the tensor dimensions I_1, I_2, I_3 in a cyclic way. Fig. 2 shows an example of a tensor $\mathcal{A} \in \mathbb{R}^{3\times 2\times 3}$, in which the matrix unfolding $\mathbf{A}_{(2)}$ is given.

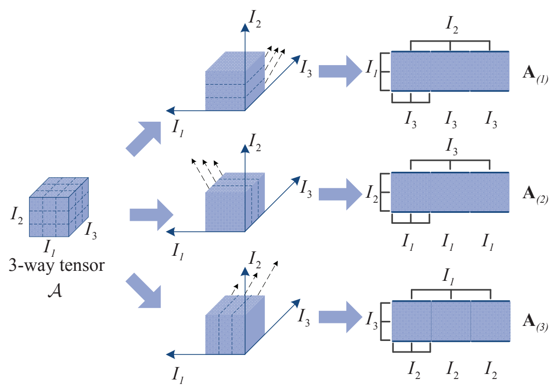


Fig. 1. Unfolding of the $(I_1 \times I_2 \times I_3) -$ tensor $\mathcal{A}$ to the $(I_1 \times I_2 I_3) -$ matrix $\mathbf{A}_{(\mathbf{1})}$, the $(I_2 \times I_3 I_1) -$ matrix $\mathbf{A}_{(\mathbf{2})}$, and the $(I_3 \times I_1 I_2) -$ matrix $\mathbf{A}_{(\mathbf{3})}$

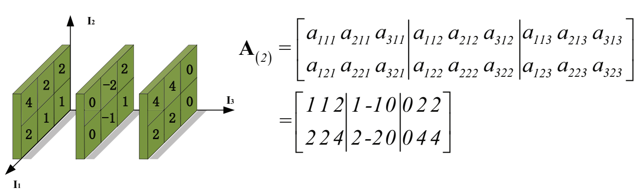


Fig. 2. A tensor $\mathcal{A} \in R^{3\times 2\times 3}$

Definition 3. ***tensor rank*** *or* ***CP-rank*** *[17], [22]: The rank of an arbitrary Nth-order tensor $\mathcal{A}$, denoted by R =rank($\mathcal{A}$), is the minimal number of rank-1 tensors that yield $\mathcal{A}$ in a linear combination. In other words, this is the smallest number of components in an exact CP decomposition [18], [19].*

One major difference between the matrix rank and the tensor CP-rank is that there is no straightforward algorithm to determine the CP-rank of a specific given tensor, which is proven to be NP-hard problem [22].

Definition 4. ***n-rank*** *[17]): The n-rank of an arbitrary Nth-order tensor $\mathcal{A}$, denoted by $R_n = rank_n(\mathcal{A})$, is the tuple of the ranks of the N unfolding matrices, that is, $R_n = \left(rank\left(\mathbf{A}_{(1)}\right), rank\left(\mathbf{A}_{(2)}\right), ..., rank\left(\mathbf{A}_{(N)}\right)\right)$.*

IV. Empirical Study with Real Traffic Data

The literature studies [23], [24] have shown that the similarity is one of the factors that impact the interpolation performance for data recovery. In this section, we perform a set of experiments with the public traffic trace Abilene [25] to investigate and discover the Internet traffic features.

A. Temporal stability

Let Υ denote the non-empty set of all origins and destinations in a network and let $|\Upsilon| = N$. Traffic data are typically measured over some time intervals, and the value reported is an average. Therefore, we can denote $z(i,j,k)$ to be the traffic from origin i to destination j averaged over the time duration $[k, k+\tau)$, where τ denotes the measurement interval.

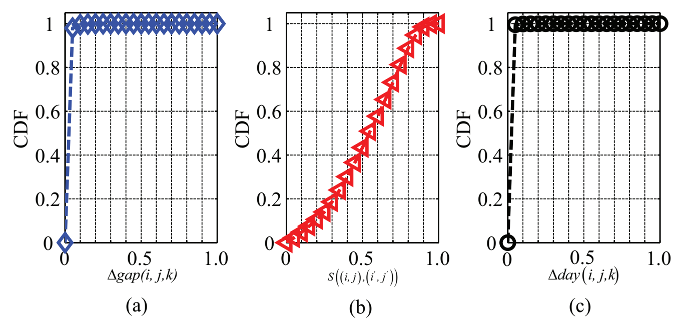


Fig. 3. Empirical study with real traffic data

Traffic data usually change slowly over time. To study the stability of traffic data, we calculate the difference between each pair of adjacent time measurements at a origin-destination (OD) pair. The difference for two consecutive time slots (k, and $k-1$) is equal to

$$gap(i,j,k) = |z(i,j,k) - z(i,j,k-1)| \tag{1}$$

where $1 \leqslant i,j \leqslant N$, $2 \leqslant k \leqslant \Gamma$ and Γ is the number of time intervals of interest. Obviously, $gap(i,j,k) = 0$ if the traffic data of OD pair (i,j) does not change from time slot $k-1$ to k. The smaller the $gap(i,j,k)$, the more stable the traffic data for OD pair (i,j) around time slot k.

By computing the normalized difference values between adjacent time slots, we measure the temporal stability at OD pair (i,j) and time slot k as

$$\Delta gap(i,j,k) = \frac{|z(i,j,k) - z(i,j,k-1)|}{\max\limits_{1\leqslant i,j\leqslant N, 2\leqslant k\leqslant \Gamma} |z(i,j,k) - z(i,j,k-1)|} \tag{2}$$

where $\max\limits_{1\leqslant i,j\leqslant N, 2\leqslant k\leqslant \Gamma} |z(i,j,k) - z(i,j,k-1)|$ is the maximal gap between any two consecutive time slots in the traffic data.

We plot the CDF of $\Delta gap(i,j,k)$ in Fig.3(a). The X-axis represents the normalized difference values between two consecutive time slots, i.e., $\Delta gap(i,j,k)$. The Y-axis represents the cumulative probability. We observe that more than 90% $\Delta gap(i,j,k)$ are very small (< 0.05). These results indicate that the temporal stability exists in the real traffic data.

B. Spatial correlation feature

A correlation coefficient is a quantitative measure of some type of correlation and dependence. Let $z(i,j), z(i',j') \in \mathbb{R}^T$ denote the traffic vectors of OD pair(i,j) and OD pair(i',j'). The spatial correlation between OD pair(i,j) and OD pair(i',j') can be calculated according to

$$\begin{aligned} & S\left((i,j),(i',j')\right) \\ & = \frac{\sum\limits_{k=1}^{\Gamma}\left(|z(i,j,k)-\bar{z}(i,j)|\times\left|z(i',j',k)-\bar{z}(i',j')\right|\right)}{\sqrt{\sum\limits_{k=1}^{\Gamma}(z(i,j,k)-\bar{z}(i,j))^2}\sqrt{\sum\limits_{k=1}^{\Gamma}(z(i',j',k)-\bar{z}(i',j'))^2}} \end{aligned} \tag{3}$$

where $1 \leq i,j,i',j' \leq N$, $\bar{z}(i,j) = \frac{1}{\Gamma}\sum\limits_{k=1}^{\Gamma} z(i,j,k)$, $\bar{z}(i',j') = \frac{1}{\Gamma}\sum\limits_{k=1}^{\Gamma} z(i',j',k)$.

The CDF of $S\left((i,j),(i',j')\right)$ is plotted in Fig. 3(b). The X-axis represents value of $S\left((i,j),(i',j')\right)$, the Y-axis represents the cumulative probability. From the figure, we can see that the value $S\left((i,j),(i',j')\right) < 0.3$ is less than 30%, the value $S\left((i,j),(i',j')\right) > 0.5$ is nearly about 60%, which indicates that real Internet traffic data have strong spatial correlation.

C. Traffic periodic pattern

As we know, users usually have similar Internet visiting behaviors at the same time of different days, such as the similar traffic mode in working hours and sleeping hours. To study the traffic periodic pattern in a day, we calculate the gap between each pair of measurements in two consecutive days at an OD pair. In Abilene [25], traffic measurements are taken every 5 minutes, one day have 288 time intervals. Therefore, the gap between each pair of measurements in adjacent days captured in two time slots (k, and $k+288$) is equal to

$$day(i,j,k) = |z(i,j,k) - z(i,j,k+288)| \tag{4}$$

where $1 \leqslant i,j \leqslant N$ and $1 \leqslant k \leqslant \Gamma - 288$ and Γ is time intervals present. Obviously, the smaller the $day(i,j,k)$, the more similar the traffic data for OD pair (i,j) around the same time slot of adjacent days .

By computing the normalized difference values between adjacent days, we measure the traffic similarity at OD pair (i,j) and time slot k according to

$$\Delta day(i,j,k) = \frac{|z(i,j,k) - z(i,j,k+288)|}{\max\limits_{1\leqslant i,j\leqslant N, 1\leqslant k\leqslant \Gamma-288} |z(i,j,k) - z(i,j,k+288)|} \tag{5}$$

where $\max\limits_{1\leqslant i,j\leqslant N, 1\leqslant k\leqslant \Gamma-288} |z(i,j,k) - z(i,j,k+288)|$ is the maximal gap between any two adjacent days in the traffic data.

We plot the CDF of $\Delta day(i,j,k)$ in Fig.3(c). The X-axis represents the normalized difference values between two adjacent days, i.e., $\Delta day(i,j,k)$. The Y-axis represents the cumulative probability. We observe that more than 90% $\Delta day(i,j,k)$ are very small (< 0.05). These results indicate that traffic periodic pattern exists in real Internet traffic trace.

V. System model and problem formulation

In this section, we first present our traffic tensor model, and then formulate the traffic data recovery problem.

A. Traffic tensor model

Current traffic interpolation approaches usually model the traffic data with a traffic matrix $\mathbf{X} \in \mathbb{R}^{o\times\Gamma}$ ($o = N \times N$), where a column of $\mathbf{X}$ represents the traffic data of all OD pairs at one time slot, while a row of $\mathbf{X}$ represents the time evolution of a single OD pair. As discussed in the introduction, modeling the traffic data in the matrix format cannot sufficiently capture spatial and temporal characteristics of the traffic data. Therefore, although matrix-based approaches work well when the ratio of the missing data is low, their performances degrade significantly when the data missing ratio becomes large.

To address the issues of the matrix-based methods mentioned above, we propose to apply the tensor to model traffic data. As a straightforward way of modeling, traffic tensor may be formed with a 3-way tensor $\mathcal{Z} \in \mathbb{R}^{N\times N\times\Gamma}$, corresponding respectively to the origin, destination and the total number of time intervals to consider. However, such a 3-way tensor model can not fully exploit the similarity structures hidden in the traffic data.

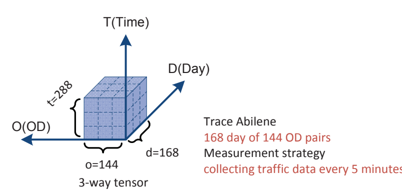


Fig. 4. Traffic tensor model

To fully exploit the traffic features of temporal stability, spatial correlation, as well as the periodicity pattern, we model the traffic data as a 3-way tensor $\mathcal{X} \in \mathbb{R}^{o\times t\times d}$ (as shown in Fig.4), where o corresponds to $N \times N$ OD pairs, and there are d days to consider with each day having t time intervals. Obviously, we have $\Gamma = t \times d$. For the Abilene trace [25], $t = 288$, $o = 144$, and $d = 168$.

B. Problem formulation

Let Ω be the set of indices of the observed entries in $\mathcal{X}$. We define a measurement tensor, $\mathcal{M} \in \mathbb{R}^{o\times t\times d}$, to record the raw measurement data. $\mathcal{M}$ is generally an incomplete tensor due to sample-based traffic monitoring and the unavoidable data loss resulted from severe communication conditions. We define the operation $\mathcal{M}_\Omega = \mathcal{X}_\Omega$ as

$$m_{ijk} = \begin{cases} x_{ijk} & if\ (i,j,k) \in \Omega \\ 0 & otherwise \end{cases} \tag{6}$$

If there are no traffic data between a particular pair of nodes in a given time interval, of course, it leaves the corresponding entry in $\mathcal{M}$ to be empty. In our study, we use zero as a placeholder to replace the empty entry.

In Section IV, the empirical studies reveal that there exist hidden structures (such as temporal stability, spatial correlation feature, and traffic periodic pattern) in the traffic data. Structure and redundancy in data are often synonymous with the data sparsity. As the rank of a tensor is a good metric to indicate its sparsity level, the traffic data recovery problem can be transformed into the mathematical task of finding a low rank tensor that can represent the original data well. Therefore, the traffic data recovery problem can be formulated as a tensor completion problem with the goal of finding its missing entries through the minimization of the tensor rank as

$$\begin{aligned} \min_{\mathcal{X}} \quad & rank\left(\mathcal{X}\right) \\ s.t. \quad & \mathcal{X}_{\Omega} = \mathcal{M}_{\Omega} \end{aligned} \tag{7}$$

According to the definition of n-rank, the n-rank of a given n-way tensor can be analyzed by means of matrix techniques. Therefore, the tensor completion problem defined in (7) can be further transformed to

$$\begin{aligned} \min_{\mathcal{X}} \quad & \sum_{i}^{3} rank\left(\mathbf{X}_{(i)}\right) \\ s.t. \quad & \mathcal{X}_{\Omega} = \mathcal{M}_{\Omega} \end{aligned} \tag{8}$$

VI. Traffic data recovery

The main challenge for the tensor completion is that the tensor decomposition requires a large number of computations. Focusing on reducing the computation cost, in this section, we propose a sequential tensor completion algorithm to quickly recover the traffic data.

Obviously, the transformed problem in (8) considers the tensor as multiple matrices and force the unfolding matrix along each mode of the tensor to be low rank. Therefore, the tensor completion problem is transformed to the low-rank matrix completion problem along each mode, and then the final recovered data can be obtained by folding the recovered data of each mode.

Current studies usually solve the matrix completion task by searching for a matrix with the minimum nuclear norm assuming that the matrix satisfies the incoherence condition and sufficient number of entries are observed. However, it may bring long computation time and even not converge when the sample data is not sufficient. Different from current studies, in this paper, the matrix completion task is accomplished by searching for a column space on the Grassmann manifold that matches the incomplete observations.

For the ith-mode unfolding matrix $\mathbf{X}_{(i)}$, we denote the number of rows and columns of $\mathbf{X}_{(i)}$ as $m_{(i)}$ and $n_{(i)}$. To find a rank-$r_{(i)}$ matrix $\mathbf{X}'_{(i)}$ that is consistent with the observations $(\mathbf{M}_{(i)})_{\Omega}$, the column space searching problem for the matrix completion can be expressed as

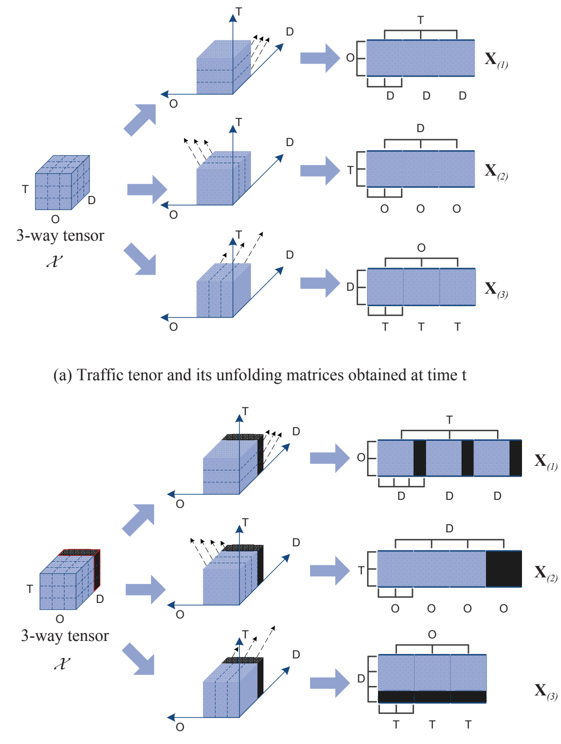
$$\min_{\mathbf{U}_{(i)} \in \mathbb{U}_{m_{(i)} r_{(i)}}} \left\| \left(\mathbf{M}_{(i)} - \mathbf{U}_{(i)} \mathbf{W}_{(i)}^{tr}\right)_{\Omega} \right\|_F^2 \tag{9}$$

where $\mathbf{U}_{(i)}$ is the column orthogonal matrix of matrix $\mathbf{M}_{(i)}$. $\mathbb{U}_{m_{(i)} r_{(i)}}$ denotes the vector space of the matrices $\in \mathbb{R}^{m_{(i)} \times r_{(i)}}$, i.e., $\mathbb{U}_{m_{(i)} r_{(i)}} := \mathbb{R}^{m_{(i)} \times r_{(i)}}$. $\|\cdot\|_F$ denotes the Frobenius norm, and $\mathbf{W}_{(i)}^{tr}$ denotes the transpose of $\mathbf{W}_{(i)}$. Given a $\mathbf{U}_{(i)}$, the $\mathbf{W}_{(i)}$ in Eq.(9) can be calculated through the following function:

$$\mathbf{W}_{(i)} = \arg\min_{\mathbf{W}_{(i)} \in \mathbb{R}^{n_{(i)} \times r_{(i)}}} \left\| \left(\mathbf{M}_{(i)} - \mathbf{U}_{(i)} \mathbf{W}_{(i)}^{tr}\right)_{\Omega} \right\|_F^2 \tag{10}$$

The low-rank matrix completion is transformed to the column space searching problem with the aim of finding a column space consistent with the observed entries. After we obtain the column orthogonal matrix $\mathbf{U}_{(i)}$, through $\mathbf{U}_{(i)} \mathbf{W}_{(i)}^{tr}$, the incomplete ith-mode matrix $\mathbf{M}_{(i)}$ can be recovered and the $\mathbf{X}'_{(i)} = \mathbf{U}_{(i)} \mathbf{W}_{(i)}^{tr}$ is the resulted recovery matrix.

In the following contents, we will further utilize the good feature of the column orthogonal matrix to propose a sequential tensor completion approach to significantly speed up traffic recovery process.



(a) Traffic tenor and its unfolding matrices obtained at time t

(b) Traffic tenor and its unfolding matrices obtained at time t+1

Fig. 5. Sequence tensor completion tasks at time t and $t+1$

Traffic measurement data generally come in sequence. To obtain the complete traffic data for the advanced network management, the tensor completion task will be invoked periodically or upon the request of the network operators. It would involve a large computation cost if we directly solve the problem (9) to find the column space of each unfolding matrix when the tensor completion task is invoked.

Fig. 5 shows the sequence of the tensor completion tasks at time t and time t+1. Comparing the Fig.5 (a) with the Fig.5(b), the major parts of the traffic data (undertone color data) are the same in both figures, where the traffic data are recovered from the previous measurements. The only difference is that the tensor in Fig.5(b) has more traffic data than the tensor in Fig. 5(a), and consequently more columns and rows in the unfolding matrices. The additional data are obtained from the new measurements. This relationship provides us an opportunity to reuse the previous result of tensor decomposition to deduce the tensor decomposition for the current data so the data can be quickly recovered.

For a rank-r matrix $\mathbf{X} = [\mathbf{X}_{\{1,\ldots,n-1\}}, x_{\{n\}}] \in \mathbb{R}^{m \times n}$, where $\mathbf{X}_{\{1,\ldots,n-1\}}$ is the submatrix of $\mathbf{X}$ by removing the last column from $\mathbf{X}$, and $x_{\{n\}}$ is the last column of $\mathbf{X}$ with its observed entry set Ω_n. Let $\mathbf{M}$ be the observation matrix of $\mathbf{X}$, that is, $\mathbf{M}_{\Omega} = \mathbf{X}_{\Omega}$ and $\left(m_{\{n\}}\right)_{\Omega_n} = \left(x_{\{n\}}\right)_{\Omega_n}$. To recover matrix $\mathbf{X}$ from $\mathbf{M}$, before we present our sequential tensor completion algorithm in Algorithm 1, the following theorem will illustrate how to calculate the column orthogonal matrix of $\mathbf{M} = [\mathbf{M}_{\{1,\ldots,n-1\}}, m_{\{n\}}] \in \mathbb{R}^{m \times n}$ based on the obtained column orthogonal matrix of $\mathbf{M}_{\{1,\ldots,n-1\}}$.

Theorem 1. *Let* $\mathbf{U}_1 = \underset{\mathbf{U} \in \mathbb{U}_{mr}}{\arg\min} \|(\mathbf{M} - \mathbf{U}\mathbf{W}^{tr})_\Omega\|_F^2$ *, and define*

$$\mathbf{U}' = \mathbf{U}_0 + \left((cos\,(\sigma\eta) - 1) \frac{p}{\|p\|} + sin\,(\sigma\eta) \frac{\ell}{\|\ell\|} \right) \frac{\omega^{tr}}{\|\omega\|} \quad (11)$$

where $\mathbf{U}_0$ *is an* $m \times r$ *matrix whose orthogonal columns span* $\mathbf{M}_{\{1,\ldots,n-1\}}$*, and* $\eta > 0$ *is a small stepsize,* $\omega = argmin_\omega \left\| (\mathbf{U}_0)_{\Omega_n} \omega - \left(m_{\{n\}}\right)_{\Omega_n} \right\|_2^2$ *is the least-squares weight,* $p = \mathbf{U}_0\omega$*,* $\ell = \left(m_{\{n\}}\right)_{\Omega_\eta} - p$ *is the residual vector, and* $\sigma = \|\ell\|\,\|p\|$*. Then* $\mathbf{U}_1$ *and* $\mathbf{U}'$ *are identical with a specific choice for step size* η*.*

Proof: According to the orthonormal columns of $\mathbf{U}_0$ which spans $\mathbf{M}_{\{1,\ldots,n-1\}}$, we can get

$$\mathbf{U}_0 = \underset{\mathbf{U} \in \mathbb{U}_{mr}}{argmin} \left\| \left(\mathbf{M}_{\{1,\ldots,n-1\}} - \mathbf{U}\mathbf{W}^{tr} \right)_\Omega \right\|_F^2$$

From $m_{\{n\}} = \mathbf{U}_0 w + \ell$, we have

$$m_{\{n\}} = \begin{bmatrix} \mathbf{U}_0 & \frac{\ell}{\|\ell\|} \end{bmatrix} \begin{bmatrix} w \\ \|\ell\| \end{bmatrix}.$$

And then, we can get

$$[\mathbf{M}_{\{1,\ldots,n-1\}}, m_{\{n\}}] = \begin{bmatrix} \mathbf{U}_0 & \frac{\ell}{\|\ell\|} \end{bmatrix} \begin{bmatrix} \mathbf{I} & w \\ 0 & \|\ell\| \end{bmatrix} \begin{bmatrix} \mathbf{W} & 0 \\ 0 & 1 \end{bmatrix}^{tr}.$$

Furthermore, we have

$$\begin{bmatrix} \mathbf{U}_0 & \frac{\ell}{\|\ell\|} \end{bmatrix} = \underset{\mathbf{U} \in \mathbb{U}_{mr+1}}{argmin} \|(\mathbf{M} - \mathbf{U}\mathbf{W}^{tr})_\Omega\|_F^2$$

Taking the SVD of the center matrix to be

$$\begin{bmatrix} \mathbf{I} & w \\ 0 & \|\ell\| \end{bmatrix} = \tilde{\mathbf{U}}\tilde{\mathbf{\Sigma}}\tilde{\mathbf{V}}^{tr}; \tilde{\mathbf{\Sigma}} = \begin{bmatrix} \delta_1 & & & \\ & \ddots & & \\ & & \delta_r & \\ & & & \delta_{r+1} \end{bmatrix}.$$

To find a matrix $\in \mathbb{U}_{mr}$ by catching most energy of the first r singular values of matrix $\mathbf{M}$, set $\mathbf{U}_{t+1} = \left(\begin{bmatrix} \mathbf{U}_0 & \frac{\ell}{\|\ell\|} \end{bmatrix} \tilde{\mathbf{U}} \right)_{\{1,\ldots,r\}}$, $\mathbf{W}_{t+1} = \left(\begin{bmatrix} \mathbf{W} & 0 \\ 0 & 1 \end{bmatrix} \tilde{\mathbf{V}}\tilde{\mathbf{\Sigma}} \right)_{\{1,\ldots,r\}}$, only the top r singular vectors are needed.

According the ISVD algorithm in [26], we can get $\mathbf{U}_{t+1} = \underset{\mathbf{U} \in U_{mr}}{argmin} \left\| \left([\mathbf{M}_{\{1,\ldots,n-1\}}, m_{\{n\}}] - \mathbf{U}\mathbf{W}^{tr} \right)_\Omega \right\|_F^2$

It was shown in [27] that updating $\mathbf{U}_0$ to $\mathbf{U}_{t+1}$ is equivalent to GROUSE for a specific step size η, which performs the gradient descent directly on the Grassmann manifold, that is, $\mathbf{U}_{t+1} = \mathbf{U}' = \mathbf{U}_0 + \left((cos\,(\sigma\eta) - 1) \frac{p}{\|p\|} + sin\,(\sigma\eta) \frac{\ell}{\|\ell\|} \right) \frac{\omega^{tr}}{\|\omega\|}$, which completes the proof.

■

According to Theorem 1, when a new column vector v is appended to the matrix $\mathbf{M}$, we do not need a new column space searching procedure to calculate the orthogonal column matrix $\mathbf{U}'$ for the matrix $[\mathbf{M}, v]$. Instead, $\mathbf{U}'$ can be derived from $\mathbf{U}$ and v only, where $\mathbf{U}$ is the orthogonal column matrix of $\mathbf{M}$ whose orthogonal columns span $\mathbf{M}$. Therefore, Theorem 1 provides a good approach to reuse the column space found for the previous traffic data to quickly recover the current traffic data.

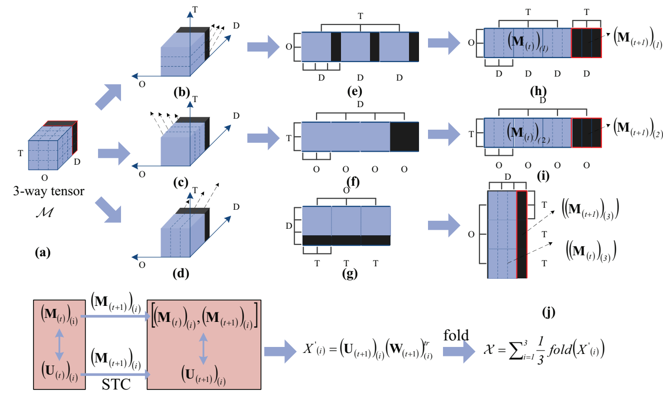


Fig. 6. Sequential tensor completion for traffic data

To design our sequential tensor completion algorithm, we first provide some notations. As shown in Fig.6, the undertone color data are processed in the previous tensor completion procedure, while the dark color data are newly obtained. The three unfolding matrices of the traffic tensor (in Fig. 6 (a)) are shown in Fig. 6 (e), Fig. 6(f), and Fig. 6(g), which can be further transformed into Fig. 6(h), Fig. 6(i), and Fig.6(j), respectively. Note that, the traffic matrices in Fig. 6(h) and (i) are the elementary transformation of matrices in Fig. 6(e) and (f), the traffic matrix in Fig. 6(j) is the transpose of the matrix in Fig. 6(g).

As shown in Fig. 6(h), Fig. 6(i), and Fig. 6(j), we denote undertone color data as $(\mathbf{M}_{(t)})_{(1)}$, $(\mathbf{M}_{(t)})_{(2)}$, and $(\mathbf{M}_{(t)})_{(3)}$, and the remainder dark color data as $(\mathbf{M}_{(t+1)})_{(1)}$, $(\mathbf{M}_{(t+1)})_{(2)}$, and $(\mathbf{M}_{(t+1)})_{(3)}$, respectively. According to Theorem 1, by utilizing the column space of $(\mathbf{M}_{(t)})_{(1)}$, $(\mathbf{M}_{(t)})_{(2)}$, and $(\mathbf{M}_{(t)})_{(3)}$ to calculate the column space of the whole $\left[\left(\mathbf{M}_{(t)}\right)_{(1)}, \left(\mathbf{M}_{(t+1)}\right)_{(1)} \right]$, $\left[\left(\mathbf{M}_{(t)}\right)_{(2)}, \left(\mathbf{M}_{(t+1)}\right)_{(2)} \right]$, and $\left[\left(\mathbf{M}_{(t)}\right)_{(3)}, \left(\mathbf{M}_{(t+1)}\right)_{(3)} \right]$, we design our Sequential Tensor Completion algorithm (SCT), as shown in Algorithm 1.

As shown on lines 2-5 in Algorithm 1, for the newly coming traffic data in $(\mathbf{M}_{(t+1)})_{(i)}$ $(1 \leq i \leq 3)$, we add each column in $(\mathbf{M}_{(t+1)})_{(i)}$ sequentially to existing data, and update the corresponding column space by utilizing the previous $\left(\mathbf{U}_{(t)}\right)_{(i)}$ and the new column to add.

Then according to Eq.(10), calculate the optimal $\left(\mathbf{W}_{(t+1)}\right)_{(i)}$ and set $\mathbf{X}'_{(i)} = \left(\mathbf{U}_{(t+1)}\right)_{(i)} \left(\mathbf{W}_{(t+1)}\right)_{(i)}^{tr}$ as the recovery matrix for this unfolding matrix. After folding each recovered unfolding matrix $\mathbf{X}'_{(1)}$, $\mathbf{X}'_{(2)}$, and $\mathbf{X}'_{(3)}$, the recovered traffic tensor can obtained as shown on line 9.

VII. Performance Evaluations

We evaluate the performance of our proposed algorithm using the public traffic trace data Abilene [25]. The metrics we

Algorithm 1 Sequential Tensor Completion (STC)

Input: The orthogonal matrices $\left(\mathbf{U}_{(t)}\right)_{(1)}$, $\left(\mathbf{U}_{(t)}\right)_{(2)}$, $\left(\mathbf{U}_{(t)}\right)_{(3)}$ for $(\mathbf{M}_{(t)})_{(1)}$, $(\mathbf{M}_{(t)})_{(2)}$, and $(\mathbf{M}_{(t)})_{(3)}$

Output: The recovered traffic tensor $\mathcal{X}$

for $i \leftarrow 1, ..., 3$ **do**

$\left(\mathbf{U}_{(t+1)}\right)_{(i)} = \left(\mathbf{U}_{(t)}\right)_{(i)}$

for each column vector v in $(\mathbf{M}_{(t+1)})_{(i)}$ with its observed entry set Ω_v **do**

Apply theorem 1 to update the column orthogonal matrix

$$\left(\mathbf{U}_{(t+1)}\right)_{(i)} = \left(\mathbf{U}_{(t+1)}\right)_{(i)} + \left(\left(\cos\left(\sigma\eta\right) - 1\right) \frac{p}{\|p\|} + \sin\left(\sigma\eta\right) \frac{\ell}{\|\ell\|} \right) \frac{\omega^{tr}}{\|\omega\|} \tag{12}$$

where $\omega = argmin_{\omega} \left\| \left(\mathbf{U}_{(t+1)}\right)_{(i)\Omega_v} \omega - (v)_{\Omega_v} \right\|_2^2$, $p = \left(\mathbf{U}_{(t+1)}\right)_{(i)} \omega$, residual $\ell = (v)_{\Omega_v} - p$, and $\sigma = \|\ell\| \, \|p\|$.

end for

According to Eq.(10), $\left(\mathbf{W}_{(t+1)}\right)_{(i)}$ can be calculated from

$$\left(\mathbf{W}_{(t+1)}\right)_{(i)} = \underset{\mathbf{W} \in \mathbb{R}^{n_{(i)} \times r_{(i)}}}{\arg\min} \left\| \left(\begin{array}{c} \left[\left(\mathbf{M}_{(t)}\right)_{(i)}, \left(\mathbf{M}_{(t+1)}\right)_{(i)} \right] \\ - \left(\mathbf{U}_{(t+1)}\right)_{(i)} \mathbf{W}^{tr} \end{array} \right)_{\Omega} \right\|_F^2 \tag{13}$$

$\mathbf{X}'_{(i)} = \left(\mathbf{U}_{(t+1)}\right)_{(i)} \left(\mathbf{W}_{(t+1)}\right)_{(i)}^{tr}$

end for

$\mathcal{X} = \sum_{i=1}^{3} \frac{1}{3} fold\left(\mathbf{X}'_{(i)}\right)$

Return traffic tensor $\mathcal{X}$.

consider include: Error Ratio and the Recovery Computation Time.

As mentioned in Section VI, traffic measurement data generally come in sequence. In the simulation, in each sequential step, we add one more day measurement data. Then we apply the tensor completion to the measurement data to recover the full data. Finally, we calculate the error ratio by comparing the recovered data with the raw data trace. In this paper, one sequence recovery step in the simulations includes the above three operations.

Definition 5. ***Error Ratio:*** *a metric for measuring the recovery error of entries in the tensor after the interpolation, which can be calculated as*

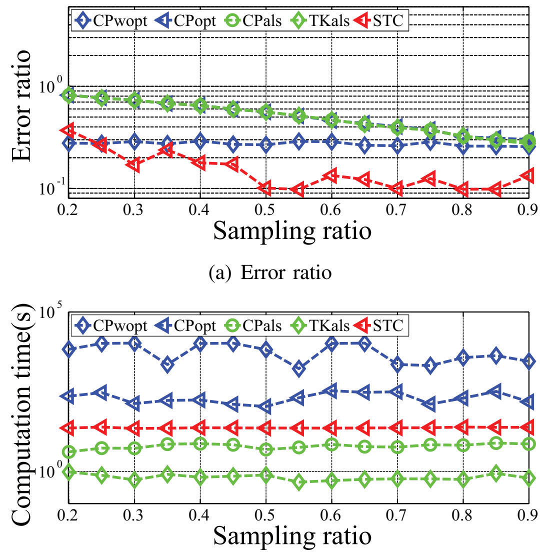
$$\frac{\sqrt{\sum_{i,j,k=d} \left(x_{ijk} - \hat{x}_{ijk}\right)^2}}{\sqrt{\sum_{i,j,k=d} x_{ijk}^2}}, \tag{14}$$

where $1 \le i \le o$, $1 \le j \le t$ *and* $k = d$. x_{ijk} *and* $\hat{x}_{ijk}$ *in (14) denote the raw data and the recovered data at* (i, j, k)*-th element of* $\mathcal{X}$*, respectively.*

Note that $k = d$ in Eq.(14), that is, only the last measurement data in the last day is counted into the performance metric calculation.

Definition 6. ***Recovery Computation Time:*** *a metric for measuring the average number of seconds of one sequence recovery step.*

All simulations are run on a Microstar workstation, which is equipped with two Intel (R) Xeon (R) E5-2620 CPUs (2.00GHz) (totaliy 24 Cores) and 32.00GB RAM. To measure the recovery computation time, we insert a timer to all the implemented approaches.



(a) Error ratio

(b) Computation time

Fig. 7. Sample ratio

A. Comparison with other tensor completion algorithms

Besides our STC, we implement other four tensor completion algorithms.

1) CP_{wopt} [14]: CP_{wopt} (CP Weighted Optimization) addresses the problem of fitting the CP model to incomplete data sets by solving a weighted least squares problem ([14] $Eq.(2)$) with a first-order optimization approach.

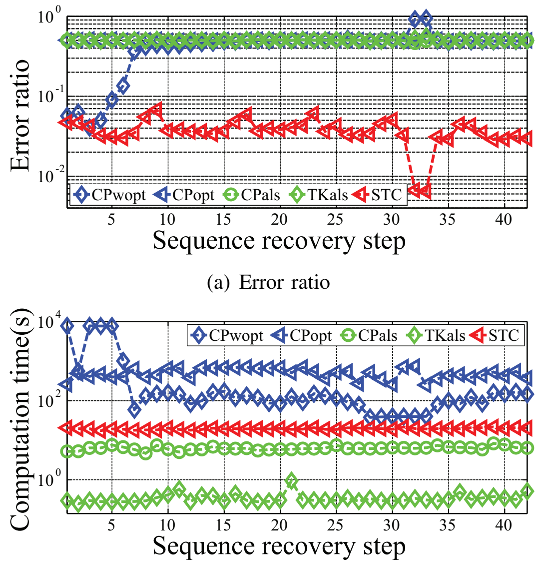
2) CP_{opt} [15]: different from CP_{wopt}, CP_{opt} addresses the problem of fitting the CP model to incomplete data sets by solving a least-square (ALS) optimization problem ([15] $Eq.(2)$) with a gradient-based optimization approach.

3)CP_{als}: CP_{als} addresses the problem of fitting the CP model to incomplete data sets by solving an alternating least-square problem. It is implemented using the Tensor Toolbox [28].

4) TK_{als}: TK_{als} addresses the problem of fitting the Tucker model to incomplete data sets by solving an alternating least-squares problem. It is implemented using the Tensor Toolbox [28].

Among the four peer tensor completion algorithms, the first three CP_{wopt}, CP_{opt}, and CP_{als} are designed based on CP model, the last TK_{als} is designed based on the Tucker model.

Fig.7 shows the performance results under different sampling ratio for one sequence step. Obviously, Fig.7(a) shows that our algorithm can achieve the highest recovery performance with the least error ratio. This good performance demonstrates that our STC algorithm has the good ability of capturing the global information in the traffic data to recover the missing data with a high accuracy.



(a) Error ratio

(b) Computation time

Fig. 8. Multiple sequence steps

From Fig.7(a), among all the implemented tensor completion algorithms, STC and CP_{wopt} are the top two algorithms that can recover data with low error ratio. While as shown in Fig.7(b), the computation time under CP_{wopt} is much larger than that under STC. Although the computation time under CP_{als} and TK_{als} is smaller than our STC (Fig.7(b)), CP_{als} and TK_{als} fail to recover data with high error ratio as shown in Fig. 7(a).

Fig. 8 shows the recovery performance under multiple sequence steps by fixing the sampling ratio to 50%. The results are consistent with those in Fig.7. It is worth noticing that, the computation time of the recovery under our STC remains the same for each sequential step, while the computation time is not stable under CP_{wopt}. Therefore CP_{wopt} is sensitive to the data added in each step. This simulation results demonstrate that our STC is a good tensor recovery algorithm that is not sensitive to the data needed to recover.

B. Comparison with matrix completion algorithms

Among all the current traffic inferring studies, the matrix-completion-based recovery algorithm is proven to achieve the best performance. In this part, we further implement other seven matrix completion algorithms for the performance comparison.

1) NMF [29]: NMF performs non-negative matrix factorization, where the non-negative matrix factorization is a recently developed technique for finding part-based, linear representations of non-negative data. Given a non-negative matrix $\mathbf{V}$, the goal of NMF is to find the non-negative matrix factors $\mathbf{W}$ and $\mathbf{H}$ such that $\mathbf{V} = \mathbf{W}\mathbf{H}^{tr}$.

2) $SRMF$ [7]: $SRMF$ is a matrix interpolation technique which uses an alternating least squares procedure to find the global sparse, low-rank approximation of the traffic matrix that accounts for the spatial and temporal properties.

3) $SRSVD$ [7]: $SRSVD$ is a matrix interpolation technique which uses an alternating least squares procedure to find the sparse, low-rank approximation of the traffic matrix.

4) SVT [30]: SVT approximates the matrix with the minimum nuclear norm obeying a set of convex constraints. SVT has two remarkable features: one is that the soft-thresholding operation is applied to a sparse matrix, and the other is that the rank of the matrix obtained in the iterates is empirically non decreasing.

5) $OptSpace$ [31]: $OptSpace$ is designed based on the singular value decomposition followed by the local manifold optimization, for solving the low-rank matrix completion problem.

6) SET [32]: SET is proposed for solving the consistent matrix completion problem. The SET algorithm consists of two parts, subspace evolution and subspace transferring.

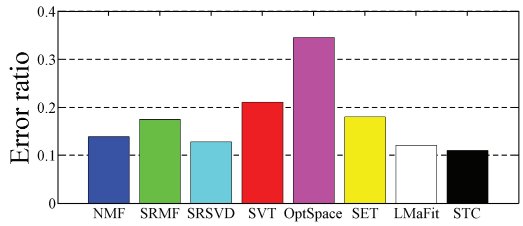
7) $LMaFit$ [33]: LMaFit is based on a nonlinear successive over-relaxation (SOR) method that only requires solving a linear least squares problem per iteration. Following the idea of the nonlinear SOR technique, LMaFit uses a weighted average between the current updated data and data from the previous iteration to achieve a faster convergence.

All the seven matrix completion algorithms are applied to the traffic matrix which is defined in $SRMF$ [7].

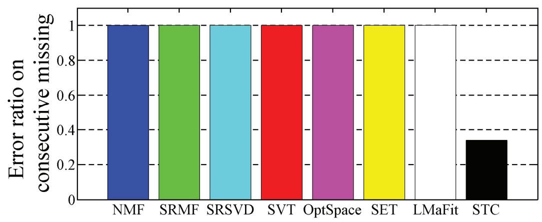
Fig.9(a) shows the error ratio under sample ratio=50% for a sequence of recovery step. Obviously, our STC achieves the best recovery performance among all the algorithms studied. Moreover, among the one day measurement data in the sequential step, we let consecutive measurements over 50 minutes all lost, and then calculate the error ratio on the 50 minutes data, as shown in Fig.9(b). The consecutive data missing, obviously, results in the consecutive column missing in the traffic matrix. From the literature work, we know that the conventional matrix completion algorithms can only recover data if there is no row or column to be completely empty. If a row or a column is missing, matrix completion algorithms do not have effect on these missing entries. Because we use zero as a placeholder to replace the empty entry, the error ratio on this kind of consecutive missing is 1 under all the matrix completion algorithms. While the error ratio on the consecutive missing data is only 0.3 under our STC. STC utilizes the information along three dimensions, while the matrix completion only considers the constraints along two particular dimensions. This is the key reason why the our STC outperforms the matrix completion-based algorithms.

VIII. Conclusion

In this paper, we apply the emerging concept of tensor completion to the recovery of the missing Internet traffic data. To well capture the spatial-temporal features inherent in the traffic data, we first analyze a large trace of real traffic data, and our studies reveal that the traffic data have the features of temporal stability, the spatial correlation, and the periodicity. To fully exploit theses hidden structures for the data recovery, we model the traffic data as a traffic tensor which can combine and utilize the multi-mode correlations.



(a) Error ratio (sampling ratio 50%)



(b) Error ratio on consecutive data missing

Fig. 9. Performance comparison with matrix completion algorithms

To reduce the computation cost for the the tensor completion, we propose a novel Sequential Tensor Completion algorithm (STC) to quickly recover the missing traffic data. We have done extensive simulations to evaluate the performance of our proposed STC algorithm, and the simulation results demonstrate that our algorithm can achieve significantly better performance compared with current of state tensor and matrix completion algorithms.

Acknowledgment

The work is supported by the National Natural Science Foundation of China under Grant Nos.61572184, 61472131, the Prospective Research Project on Future Networks (Jiangsu Future Networks Innovation Institute) under Grant No.BY2013095-4-06, the National High Technology Research and Development Program of China (863 Program) under Grant No.2015AA010201, the National Basic Research Program (973 Program) under Grant No.2012CB315805, U.S. National Science Foundation under Grant Nos. ECCS-1231800 and CNS 1247924.

References

[1] M. Roughan, M. Thorup, and Y. Zhang, "Traffic engineering with estimated traffic matrices," in *ACM IMC*, 2003.

[2] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. D. Kolaczyk, and N. Taft, "Structural analysis of network traffic flows," in *ACM SIGMETRICS*, 2003.

[3] Y. Zhang, M. Roughan, C. Lund, and D. Donoho, "Estimating point-to-point and point-to-multipoint traffic matrices: An information-theoretic approach," in *IEEE/ACM Trans. Netw*, 2005, pp. 947–960.

[4] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing network-wide traffic anomalies," *Acm Sigcomm Computer Communication Review*, vol. 34, no. 4, pp. 219–230, 2004.

[5] Y.Vardi, "Network tomography," *J. Amer. Statist. Assoc.*, vol. vol. 91, no. 433,, p. pp. 365377, 1996.

[6] P. Barford, J. Kline, D. Plonka, and A. Ron, "A signal analysis of network traffic anomalies," *ACM IMW*, 2002.

[7] M. Roughan, Y. Zhang, W. Willinger, and L. Qiu, "Spatio-temporal compressive sensing and internet traffic matrices (extended version)," *Networking IEEE/ACM Transactions on*, vol. 20, no. 3, pp. 662 – 676, 2012.

[8] M. Mardani and G. Giannakis, "Robust network traffic estimation via sparsity and low rank," in *IEEE ICASSP*, 2013.

[9] R. Du, C. Chen, B. Yang, and X. Guan, "Vanet based traffic estimation: A matrix completion approach," in *IEEE GLOBECOM*, 2013.

[10] G. Gürsun and M. Crovella, "On traffic matrix completion in the internet," in *ACM IMC 2012*.

[11] Y.-C. Chen, L. Qiu, Y. Zhang, G. Xue, and Z. Hu, "Robust network compressive sensing," in *ACM MobiCom*, 2014.

[12] K. Xie, L. Wang, X. Wang, G. Xie, G. Zhang, D. Xie, and J. Wen, "Sequential and adaptive sampling for matrix completion in network monitoring systems," in *IEEE INFOCOM*, 2015.

[13] J. Liu, P. Musialski, P. Wonka, and J. Ye, "Tensor completion for estimating missing values in visual data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 208–220, 2013.

[14] E. Acar, D. M. Dunlavy, T. G. Kolda, and M. Mørup, "Scalable tensor factorizations for incomplete data," *Chemometrics and Intelligent Laboratory Systems*, vol. 106, no. 1, pp. 41–56, 2011.

[15] E. Acar and T. G. Dunlavy, Daniel M.and Kolda, "A scalable optimization approach for fitting canonical tensor decompositions," *Journal of Chemometrics*, vol. 25, no. 2, p. 6786, 2011.

[16] S. Gandy, B. Recht, and I. Yamada, "Tensor completion and low-n-rank tensor recovery via convex optimization," *Inverse Problems*, vol. 27, no. 2, p. 025010, 2011.

[17] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications." *Siam Review*, vol. 51, no. 3, pp. 455–500, 2009.

[18] J. Carroll and J.-J. Chang, "Analysis of individual differences in multidimensional scaling via an n-way generalization of eckart-young decomposition," *Psychometrika*, vol. 35, no. 3, pp. 283–319, 1970.

[19] R. A. Harshman, "Foundations of the parafac procedure: Models and conditions for an" explanatory" multi-modal factor analysis," *UCLA Working Papers in Phonetics*, vol. 16, no. 1, p. 84, 1970.

[20] L. Tucker, "Some mathematical notes on three-mode factor analysis," *Psychometrika*, vol. 31, no. 3, pp. 279–311, 1966.

[21] L. D. Lathauwer, B. D. Moor, and J. Vandewalle, "A multilinear singular value decomposition," in *SIAM J. Matrix Anal. Appl*, 2000, pp. 1253–1278.

[22] J. Håstad, "Tensor rank is np-complete," *Journal of Algorithms*, vol. 11, no. 4, pp. 644–654, 1990.

[23] L. Qu, L. Li, Y. Zhang, and J. Hu, "Ppca-based missing data imputation for traffic flow volume: a systematical approach," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 10, no. 3, pp. 512–522, 2009.

[24] H. Tan, G. Feng, J. Feng, W. Wang, Y.-J. Zhang, and F. Li, "A tensor-based method for missing traffic data completion," *Transportation Research Part C: Emerging Technologies*, vol. 28, pp. 15–27, 2013.

[25] "The abilene observatory data collections. http://abilene. internet2.edu /observatory/data-collections.html."

[26] M. Brand, "Incremental singular value decomposition of uncertain data with missing values," in *Computer VisionECCV*. Springer, 2002.

[27] L. Balzano and S. J. Wright, "On grouse and incremental svd," in *IEEE CAMSAP*, 2013.

[28] B. W.Bader, T. G.Kolda *et al.*, "Matlab tensor toolbox version 2.5," Available online, January 2012. [Online]. Available: http://www.sandia.gov/ tgkolda/TensorToolbox/

[29] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in neural information processing systems*, 2001, pp. 556–562.

[30] J.-F. Cai, E. J. Candès, and Z. Shen, "A singular value thresholding algorithm for matrix completion," *SIAM Journal on Optimization*, vol. 20, no. 4, pp. 1956–1982, 2010.

[31] S. Keshavan, Raghunandan H.and Oh and A. Montanari, "Matrix completion from a few entries," *Information Theory IEEE Transactions on*, vol. 56, no. 6, pp. 2980–2998, 2009.

[32] W. Dai and E. Milenkovic, O.and Kerman, "Subspace evolution and transfer(set)for low-rank matrix completion," *Signal Processing IEEE Transactions on*, vol. 59, no. 7, pp. 3120–3132, 2011.

[33] Z. Wen, W. Yin, and Y. Zhang, "Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm," *Mathematical Programming Computation*, vol. 4, no. 4, pp. 333–361, 2012.