Robust Tensor Recovery with Fiber Outliers for Traffic Events

YUE HU and DANIEL B. WORK, Vanderbilt University

Event detection is gaining increasing attention in smart cities research. Large-scale mobility data serves as an important tool to uncover the dynamics of urban transportation systems, and more often than not the dataset is incomplete. In this article, we develop a method to detect extreme events in large traffic datasets, and to impute missing data during regular conditions. Specifically, we propose a robust tensor recovery problem to recover low-rank tensors under fiber-sparse corruptions with partial observations, and use it to identify events, and impute missing data under typical conditions. Our approach is scalable to large urban areas, taking full advantage of the spatio-temporal correlations in traffic patterns. We develop an efficient algorithm to solve the tensor recovery problem based on the *alternating direction method of multipliers* (ADMM) framework. Compared with existing l_1 norm regularized tensor decomposition methods, our algorithm can exactly recover the values of uncorrupted fibers of a low-rank tensor and find the positions of corrupted fibers under mild conditions. Numerical experiments illustrate that our algorithm can achieve exact recovery and outlier detection even with missing data rates as high as 40% under 5% gross corruption, depending on the tensor size and the Tucker rank of the low rank tensor. Finally, we apply our method on a real traffic dataset corresponding to downtown Nashville, TN and successfully detect the events like severe car crashes, construction lane closures, and other large events that cause significant traffic disruptions.

CCS Concepts: • **Information systems** → **Spatial-temporal systems**; *Data mining*; • **Computing methodologies** → **Anomaly detection**; **Factorization methods**;

Additional Key Words and Phrases: Robust tensor recovery, tensor factorization, multilinear analysis, outlier detection, traffic events, urban computing

ACM Reference format:
Yue Hu and Daniel B. Work. 2020. Robust Tensor Recovery with Fiber Outliers for Traffic Events. *ACM Trans. Knowl. Discov. Data* 15, 1, Article 6 (December 2020), 27 pages.
https://doi.org/10.1145/3417337

1 INTRODUCTION

1.1 Motivation

Event detection is an increasing interest in urban studies [33, 51]. Efficiently analyzing the impact of large events can help us assess the performance of urban infrastructure and aid urban management. Nowadays, with the development of intelligent transportation systems, large scale traffic

This material is based upon the work supported by the National Science Foundation under Grant No. CMMI-1727785.
Authors addresses: Y. Hu and D. B. Work, Vanderbilt University, 1025 16th Ave S, Nashville, Nashville; emails: {yue.hu, dan.work}@vanderbilt.edu.

1556-4681/2020/12-ART6 $15.00
https://doi.org/10.1145/3417337

data are accumulating from loop detectors, Global Positioning System devices, and high-resolution cameras. The large amount of data provides us insight into the dynamics of urban environments in the presence of large scale events. The main objective of this article is to develop a method to detect extreme events in traffic datasets describing large urban areas, and to impute missing data during regular conditions.

There are two major challenges in detecting extreme events in traffic datasets. First, most large traffic datasets are incomplete [9, 14, 45], meaning there are a large number of entries for which the current traffic condition is not known. The missing data can be caused by the lack of measurements (e.g., no instrumented vehicles recently drove over the road segment), or due to senor failure (e.g., a traffic sensor which loses communication, power, or is physically damaged). Missing data can heavily influence the performance of traffic estimation [9, 10, 45, 52], especially as the missing data rate increases. Naive imputation of the missing entries to create a complete dataset is problematic, because without a clear understanding of the overall pattern, an incorrect value can be imputed that will later degrade the performance of an outlier detection algorithm. Consequently, missing data should be carefully handled.

The second challenge is to fully capture and utilize the pattern of regular traffic, in order to correctly impute missing data and separate the outliers out from regular traffic. Studies have suggested that for regular traffic patterns, there exist systematic correlations in time and space [2, 16, 20, 58]. For example, due to daily commute patterns, traffic conditions during Monday morning rush hour are generally repeated but with small variation from one week to the next (e.g., the rush hour might start a little earlier or last a little longer). Also, traffic conditions are spatially structured due to the network connectivity, ending up in global patterns. For example, the traffic volume on one road segment should influence and be influenced by its downstream and upstream traffic volumes, respectively.

Most existing research has not fully addressed these two challenges. On the one hand, missing data and outlier detection tend to be dealt with separately. Either it is assumed that the dataset is complete for the purpose of outlier detection [19, 24], or it is assumed that the dataset is clean, for the purpose of missing data imputation [1, 41]. Yet in reality, missing data and outliers often exist at the same time. On the other hand, currently many studies on traffic outliers consider only a single monitoring spot or a small region [11, 18, 40, 49], not fully exploiting the spatio-temporal correlations. Only a few studies scale up to large regions to capture urban-scale correlations. Examples here include the work of Yang et al. [58], which proposes a coupled Bayesian *robust principal component analysis* (RPCA) approach to detect road traffic events, and Liu et al. [34], which constructs a spatio-temporal outlier tree to discover the causal interactions among outliers.

In this article, we tackle the traffic outlier event detection problem from a different perspective, taking into account missing data and spatio-temporal correlations. Specifically, we model a robust tensor decomposition problem, as illustrated in Section 1.2. Furthermore, we develop an efficient algorithm to solve it. We note that the application of our method is not limited to traffic event detection, but can also be applied to general pattern recognition and anomaly detection, where there exist multi-way correlations in datasets that are either completely or partially observed.

1.2 Solution Approach

In this subsection, we develop a robust tensor decomposition model for the traffic extreme event detection problem, and develop a robust tensor completion problem considering partial observations.

To exploit these temporal and spatial structures, a tensor [17, 25] is introduced to represent the traffic data over time and space. We form a three-way tensor, as shown in Figure 1. The first dimension is the road segment, the second dimension is the time of the week, (Monday midnight 1 am all

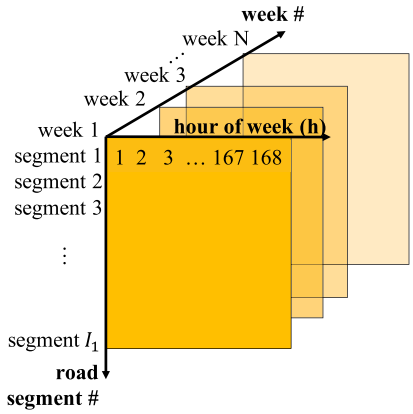

Fig. 1. Traffic data are arranged in three-way tensor, with the dimensions corresponding to *(i)* the hour of the week, *(ii)* the road segment number, and *(iii)* the week number.

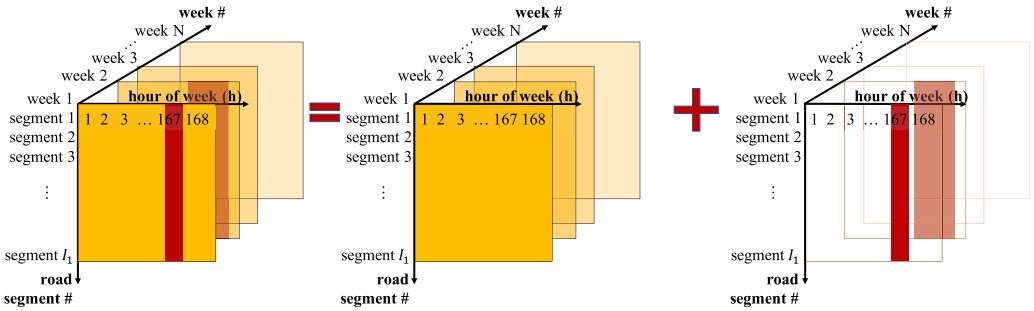

Fig. 2. Observation data decomposed into low-rank tensor for regular traffic and fiber-sparse tensor for outlier events.

the way to Sunday 11 pm midnight, $24 \times 7 = 168$ entries in total), and the third dimension is the week in the dataset. In this way, the temporal and spacial patterns along different dimensions are naturally encoded. One effective way to quantify this multi-dimensional correlation is the Tucker rank of the tensor [17, 25, 48], which is the generalization of matrix rank to higher dimensions.

As for the extreme events, we expect them to occur relatively infrequently. We encode the outliers in a sparse tensor, which is organized in the same way as the traffic data tensor. Furthermore, extreme events tend to affect the overall traffic of an urban area. That is, we assume that at the time when extreme event occurs, the traffic data of all road segments deviates from the normal pattern. Thus, the outliers occur sparsely as fibers along the road segment dimension with hour of week fixed. This sets it apart from random noises, which appear scattered over the whole tensor entries and are unstructured. This fiber-wise sparsity problem is studied in the two-dimensional matrix case [56], where the $l_{2,1}$ norm is used to control the column sparsity, and we adapt it for higher dimensions.

Putting these together, we organize the traffic data into a tensor $\mathcal{B}$, then decompose it into two parts,

$$\mathcal{B} = \mathcal{X} + \mathcal{E},$$

where tensor $\mathcal{X}$ contains the data describing the regular traffic patterns, and tensor $\mathcal{E}$ denotes the outliers, as illustrated in Figure 2. Because the normal traffic pattern is assumed to have strong

correlation in time and space, it is approximated by a low-rank tensor $\mathcal{X}$. Similarly, because outliers are infrequent, we expect the tensor containing outliers, $\mathcal{E}$, to be sparse. With these ideas in mind, it is possible formulate the following optimization problem:

$$\begin{aligned} \min_{\mathcal{X},\mathcal{E}} \quad & \text{rank}(\mathcal{X}) + \lambda \, \text{sparsity}(\mathcal{E}) \\ \text{s.t.} \quad & \mathcal{B} = \mathcal{X} + \mathcal{E}. \end{aligned} \tag{1}$$

The objective function in problem (1) is the weighted cost of the tensor rank of $\mathcal{X}$ denoted as rank($\mathcal{X}$), and the fiber-wise sparsity of $\mathcal{E}$ (denoted as sparsity($\mathcal{E}$)), and λ is a weighting parameter balancing the two costs. A more precise formulation of the problem is provided in Section 4.

In the presence of missing data, we require the decomposition to match the observation data only at the available entries, resulting in the following optimization problem:

$$\begin{aligned} \min_{\mathcal{X},\mathcal{E}} \quad & \text{rank}(\mathcal{X}) + \lambda \, \text{sparsity}(\mathcal{E}) \\ \text{s.t.} \quad & \mathcal{B}_{i_1 i_2 \ldots i_N} = (\mathcal{X} + \mathcal{E})_{i_1 i_2 \ldots i_N}, \\ & \text{where } (i_1, i_2, \ldots, i_N) \text{ is an observed entry.} \end{aligned} \tag{2}$$

In this article, we turn problems (1) and (2) into convex programming problems, and solve them by extending from the matrix case to the tensor case a singular value thresholding algorithm [7, 8] based on *alternating direction method of multipliers* (ADMM) framework. Our algorithm can exactly recover the values of uncorrupted fibers of the low rank tensor, and find the positions of corrupted fibers, based on relatively mild conditions on the observation and corruption ratio.[1]

1.3 Contributions and Outline

To summarize, this work has the following three main contributions:

(1) We propose a new robust tensor recovery with fiber outliers method for traffic extreme event detection under full or partial observations, to take full advantage of spatial-temporal correlations in traffic patterns.
(2) We propose ADMM-based algorithms to solve the robust tensor recovery under fiber-sparse corruption and partial observation. Our algorithm can exactly impute the values of uncorrupted fibers of the low-rank tensor, and find the positions of corrupted fibers under mild conditions, outperforming the existing state-of-the-art methods for tensor recovery and completion under this setting.
(3) We apply our method on a large traffic dataset in downtown Nashville and successfully detect large events.

The rest of the article is organized as follows. In Section 3, we provide a review of tensor basics and related RPCA methods. In Section 4, we formulate the tensor outlier detection problem, and propose efficient algorithms to solve it. In Section 5, we demonstrate the effectiveness of our method by numerical experiments. In Section 6, we apply our method on real world dataset and show its ability to find large scale events.

2 RELATED WORK

In this section, we review the literature on outlier detection. We also compare our methodology with other relevant works.

[1]The resulting source code is available at https://github.com/Lab-Work/Robust_tensor_recovery_for_traffic_events.

2.1 Outlier Detection

The outliers we are interested in this work are due to outliers caused by extreme events. Another related problem considers methods to detect outliers caused by data measurement errors, such as sensor malfunction, malicious tampering, or measurement error [5, 11, 40]. The latter methods can be seen as a part of a standard data cleaning or data pre-processing step. On the other hand, outliers caused by extreme traffic have valuable information for congestion management, and can provide agencies with insights into the performance of urban traffic networks. The works [26, 34, 39] explore the problem of outlier detection caused by events, while the works [28, 33, 44, 57] focus on determining the root causes of the outlier.

2.2 Low Rank Matrix and Tensor Learning

Low rank matrix and tensor learning has been widely used to exploit structure in data. Various applications have benefited from matrix- and tensor-based methods, including data completion [2, 43], link prediction [15], and network structure clustering [50]. There are two threads of work most relevant with ours, namely, robust matrix and tensor principal component analysis (PCA) for outlier detection, and low-rank tensor completion with missing data.

Regarding robust matrix and tensor PCA for outlier detection, l_1 norm regularized robust tensor recovery, as proposed by Goldfarb and Qin [17], is useful when the data are polluted with unstructured random noises. Other distributions, such as the Cauchy distribution and the chi-squared distribution, are also used [54, 59] to deal with gross corruption and small noises simultaneously. Tan et al. [46] use l_1 norm regularized tensor decomposition for traffic data recovery in the presence of random noise corruptions. But if the outliers are structured, for example grouped in columns, l_1 norm regularization does not yield good results. In addition, although traffic is also modeled as a tensor in [46], only a single road segment is considered and thus does not account for network spacial structures.

When the data contains large structured events, the outliers can group in columns or fibers of the dataset, as illustrated in Section 1.2. In this setting, $l_{2,1}$ norm regularized decomposition is suitable for group outlier detection, as shown in [47, 56] for matrices, and [42, 62] for tensors. In addition, Li et al. [30] introduce a multi-view low-rank analysis framework for outlier detection, and Wen et al. [51] use discriminant tensor factorization for event analytics. Our methods differ from the existing tensor outlier pursuit methods [42, 62] which deal with slab outliers, i.e., outliers form an entire slice instead of fibers of the tensor. Moreover, compared with existing works, we also address the setting with partial observations. As stated in Section 1.1, without an overall understanding of the underlying pattern, one can incorrectly impute missing entries that later influence decisions about outliers.

Regarding tensor completion methods, an overview is provided by Song et al. [43] from the perspective of big data analysis. Two tensor factorization methods, namely, CANDECOMP/PARAFAC (CP) decomposition [23] and Tucker decomposition [48], are used for tensor completion. According to the CP decomposition framework, Zhao et al. [61] propose a fully Bayesian probabilistic model for tensor factorization and completion. Wu et al. [55] also use CP decomposition and take numerous priors into account to propose a fused CP method for tensor completion. Tucker decomposition is used in [13], where manifold information is incorporated to propose a *simultaneous tensor decomposition and completion* (STDC) method. Implementations of several methods discussed above can be found at [53].

The major difference of our method with the existing methods is our problem setting. We consider the problem when the data contains simultaneously fiber-wise gross corruption and missing data. This is more difficult than the problem when only missing data exists, or only corrupted

data exists. Moreover, this is an important setting in large-scale traffic analysis where large outlier events occur and missing data is common. In comparison, while Zhou and Feng [62] address large outliers, the situation of missing data is not explicitly considered. Chen et al. [13] address missing data without outliers, and Wu et al. [55] and Zhao et al. [61] assume small Gaussian noise corruption with missing data. We will demonstrate in our simulations in Section 5 that our algorithm can improve the accuracy of the reconstruction by several orders of magnitude when gross corruption and missing data are present, compared to other methods that are designed to handle missing data [13, 59, 61].

3 PRELIMINARIES

In this section, we briefly review the mathematical preliminaries for tensor factorization, adopting the notation of Kolda and Bader [25], and Goldfarb and Qin [17]. We also summarize RPCA [8], since it serves as a foundation for our extension to higher-order tensor decomposition.

3.1 Tensor Basics

In this article, a tensor is denoted by an Euler script letter (e.g., $\mathcal{X}$); a matrix by a boldface capital letter (e.g., $\mathbf{X}$); a vector by a boldface lowercase letter (e.g., $\mathbf{x}$); and a scalar by a lowercase letter (e.g., x). A tensor of order N has N dimensions, and can be equivalently described as an N-way tensor or an N-mode tensor. Thus, a matrix is a second order tensor, and a vector is a first order tensor.

A *fiber* is a column vector formed by fixing all indices of a tensor but one. In a matrix for example, each column can be viewed as a mode-one fiber, and each row a mode-two fiber.

The *unfolding* of a tensor $\mathcal{X}$ in the n^{th} mode results in a matrix $\mathbf{X}_{(n)}$, which is formed by rearranging the mode-n fibers as its columns. This process is also called flattening or matricization. The inverse function of unfolding is denoted as $\text{fold}_n(\cdot)$, i.e.,

$$\text{fold}_n(\mathbf{X}_{(n)}) = \mathcal{X}.$$

The inner product of two tensors $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is the sum of their element-wise product, similar to vector inner products. Let $x_{i_1 i_2 \ldots i_N}$ and $y_{i_1 i_2 \ldots i_N}$ denote the $(i_1, i_2, \ldots, i_N)$ element of $\mathcal{X}$ and $\mathcal{Y}$ respectively. Then tensor inner product can be expressed as

$$\langle \mathcal{X}, \mathcal{Y} \rangle = \sum_{i_1=1}^{I_1} \sum_{i_2=1}^{I_2} \cdots \sum_{i_N=1}^{I_N} x_{i_1 i_2 \ldots i_N} y_{i_1 i_2 \ldots i_N}.$$

The *tensor Frobenius norm* is denoted by $\| \cdot \|_F$, and computed as

$$\|\mathcal{X}\|_F = \sqrt{\langle \mathcal{X}, \mathcal{X} \rangle}.$$

Multiplication of a tensor by a matrix in mode n is performed by multiplying every mode-n fiber of the tensor by the matrix. The mode-n product of a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ and a matrix $\mathbf{A} \in \mathbb{R}^{J \times I_n}$ is denoted by $\mathcal{X} \times_n \mathbf{A} = \mathcal{Y}$, where $\mathcal{Y} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_{n-1} \times J \times I_{n+1} \times \cdots \times I_N}$. It is also equivalently written via its mode-n unfolding as $\mathbf{Y}_{(n)} := \mathbf{A}\mathbf{X}_{(n)}$.

The *Tucker decomposition* [17, 25] is the generalization of matrix PCA in higher dimensions. It approximates a tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ as a core tensor multiplied in each mode n by an appropriately sized matrix $\mathbf{U}^{(n)}$:

$$\mathcal{X} \approx \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times \cdots \times_N \mathbf{U}^{(N)}. \tag{3}$$

$\mathcal{G} \in \mathbb{R}^{c_1 \times c_2 \times \cdots \times c_N}$ in (3) is called the core tensor, where c_1 through c_N are given integers. If $c_n < I_n$ for some n in $(1, 2, \ldots, N)$, the core tensor $\mathcal{G}$ can be viewed as a compressed version of $\mathcal{X}$. The matrices $\mathbf{U}^{(n)} \in \mathbb{R}^{I_n \times c_n}$ are factor matrices, which are usually assumed to be orthogonal.

The *n-rank* of $\mathcal{X}$, denoted by $\text{rank}_n(\mathcal{X})$, is the column rank of $\mathbf{X}_{(n)}$. In other words, it is the dimension of the vector space spanned by the mode-n fibers. If we denote the n-rank of the tensor $\mathcal{X}$ as R_n for $n = 1, 2, \ldots, N$, i.e., $R_n = \text{rank}_n(\mathcal{X})$, then the set of the N n-ranks of $\mathcal{X}$, $(R_1, R_2, \ldots, R_N)$, is called the *Tucker Rank* [25]. In Tucker decomposition (3), if $c_n = \text{rank}_n(\mathcal{X})$ for all n in $(1, 2, \ldots, N)$, then the Tucker decomposition is exact. In this case, we can easily conduct the decomposition by setting $\mathbf{U}^{(n)}$ as the left singular matrix of $\mathbf{X}_{(n)}$. Otherwise, if $c_n < \text{rank}_n(\mathcal{X})$, the decomposition holds only as an approximation.

3.2 RPCA

We briefly summarize robust variants of PCA in the matrix setting, which are extended to higher-order tensor settings in Section 4. RPCA belongs to the family of dimension-reduction methods aiming at combating the so-called *curse of dimensionality* that often appears when dealing with large, high dimensional datasets, by finding the best representing low-dimensional subspace. PCA is a widely used technique in this family, yet it is sensitive to corruptions [8]. For example, consider a large data matrix that comes from a low rank matrix randomly corrupted by large noises, i.e.,

$$\mathbf{B} = \mathbf{X} + \mathbf{E},$$

where $\mathbf{B} \in \mathbb{R}^{I_1 \times I_2}$ is the data matrix, $\mathbf{X} \in \mathbb{R}^{I_1 \times I_2}$ is a low rank matrix, and $\mathbf{E} \in \mathbb{R}^{I_1 \times I_2}$ is a sparse corruption matrix of arbitrary magnitude. In this setting traditional PCA can fail to find the subspace for $\mathbf{X}$ given only $\mathbf{B}$.

To address the problem of gross corruption, Candès et al. [8] proposed an RPCA method known as *Principle Component Pursuit* (PCP):

$$\begin{aligned} \min_{\mathbf{X},\mathbf{E}} \quad & \|\mathbf{X}\|_* + \lambda \|\mathbf{E}\|_1 \\ \text{s.t.} \quad & \mathbf{B} = \mathbf{X} + \mathbf{E}, \end{aligned} \tag{4}$$

with the l_1 matrix norm $\|\cdot\|_1$ of $\mathbf{E}$ given by:

$$\|\mathbf{E}\|_1 := \sum_{i=1}^{I_1} \sum_{j=1}^{I_2} |e_{ij}|,$$

and where $e_{i,j}$ denotes the (i, j)th element of $\mathbf{E}$. The nuclear norm of a matrix $\mathbf{X}$ is denoted as $\|\cdot\|_*$ and is computed as the sum of the singular values of $\mathbf{X}$:

$$\|\mathbf{X}\|_* := \sum_i \sigma_i.$$

where the Singular value decomposition (SVD) of $\mathbf{X}$ is $\mathbf{X} = \sum_i \sigma_i \mathbf{u}_i \mathbf{v}_i^T$.

The nuclear norm in Equation (4) is proposed as the tightest convex relaxation of the matrix rank [7]; and the l_1 norm is the convex approximation for element-wise matrix sparsity. Candès et al. [8] showed that PCP can exactly recover a low rank matrix when it is grossly corrupted at sparse entries. Moreover, by adopting an ADMM algorithm, it is possible to solve Equation (4) in polynomial time. The PCP formulation (4) requires incoherence of the column space of the sparse matrix $\mathbf{E}$ [8, 56], and does not address the setting where entire columns are corrupted.

An alternative problem formulation using an $l_{2,1}$ norm on $\mathbf{E}$ in Equation (4) is introduced for matrix recovery with column-wise corruption [47, 56]. The $l_{2,1}$ norm of a matrix $\mathbf{E} \in \mathbb{R}^{I_1 \times I_2}$ is defined as

$$\|\mathbf{E}\|_{2,1} = \sum_{j=1}^{I_2} \sqrt{\sum_{i=1}^{I_1} (e_{ij})^2}.$$

It is essentially a form of group lasso [22], where each column is treated as a group. Minimizing $\|\mathbf{E}\|_{2,1}$ encourages the entire columns of $\mathbf{E}$ to be zero or non-zero, and leads to fewer non-zero columns.

Note that it is hard to recover an uncorrupted column from a completely corrupted one. Therefore, instead of trying to recover the complete low-rank matrix, Xu et al. [56] seek instead to recover the exact low-dimensional subspace while identifying the location of the corrupted columns. Tang and Nehorai [47] make an assumption that if a column is corrupted (i.e., $\mathbf{E}$ has nonzero entries in this column), then the entries of the corresponding column in the low-rank matrix $\mathbf{X}$ are zero. This choice allows exact recovery of the low-rank matrix in the non-corrupted columns.

Like PCA for a matrix, we note that the Tucker decomposition of a tensor is also sensitive to gross corruption [17]. Motivated by the ideas of Candès et al. [8] and Tang and Nehorai [47] for robust matrix PCA, in the next section we address the problem of robust decomposition of tensors with gross fiber-wise corruption.

4 METHODS

In this section, we define and pose the higher-order tensor decomposition problem in the presence of fiber outliers and its partial-observation variant as convex programs, and provide efficient algorithms to solve them.

4.1 Problem Formulation

The precise setup of the problem is as follows. We are given a high-dimensional data tensor $\mathcal{B} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ which is composed of a low-rank tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ that is corrupted in a few fibers. In other words, we have $\mathcal{B} = \mathcal{X} + \mathcal{E}$, where $\mathcal{E} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is the sparse fiber outlier tensor. We know neither the rank of $\mathcal{X}$, nor the number and position of non-zero entries of $\mathcal{E}$. Given only $\mathcal{B}$, our goal is to reconstruct $\mathcal{X}$ on the non-corrupted fibers, as well as identify the outlier location. Moreover, we might have only partial observations of $\mathcal{B}$, and we seek to complete the decomposition nevertheless.

We do assume knowledge of the mode along which the fiber outliers are distributed; without loss of generality let it be the first dimension. Then it is equivalent to say the mode-1 unfolding of the outlier tensor is column-wise sparse. Thus, we can formulate the problem as:

$$\begin{aligned} \min_{\mathcal{X},\mathcal{E}} \quad & \operatorname{rank}(\mathcal{X}) + \lambda \|\mathbf{E}_{(1)}\|_{2,1} \\ \text{s.t.} \quad & \mathcal{B} = \mathcal{X} + \mathcal{E}. \end{aligned} \tag{5}$$

The mode along which to unfold the outlier tensor $\mathcal{E}$ in Equation (5) will influence what kind of outliers the algorithm will detect. One can extend the algorithm by adding terms of $l_{2,1}$ norm of other unfoldings, corresponding to outliers along other dimensions (e.g., gross corruption in time). In our work, we seek to discover large (in space) events that are acute in time. As a result, we select the $l_{2,1}$ norm along the spatial dimension.

Computing the rank of a tensor $\mathcal{X}$, denoted by $\operatorname{rank}(\mathcal{X})$ is generally an NP-hard problem [17, 21]. One commonly used convex relaxation of the tensor rank is $\sum_i \|\mathbf{X}_{(i)}\|_*$, which sums the nuclear norm of the tensor unfoldings in all modes [17]. In this way, we generalize the matrix nuclear norm to the higher-order case, and explore the potential low rank structure in all dimensions. Problem (5) thus becomes

$$\begin{aligned} \min_{\mathcal{X},\mathcal{E}} \quad & \sum_{i=1}^{N} \|\mathbf{X}_{(i)}\|_* + \lambda \|\mathbf{E}_{(1)}\|_{2,1} \\ \text{s.t.} \quad & \mathcal{B} = \mathcal{X} + \mathcal{E}. \end{aligned} \tag{6}$$

Next, we deal with the case when the data are only partially available, in addition to observation data being grossly corrupted. We only know the entries $(i_1, i_2, \ldots, i_N) \in \Omega$, where $\Omega \subset [I_1] \times [I_2] \times \cdots \times [I_N]$ is an observation index set. Let $\mathcal{X}_\Omega$ denote the projection of $\mathcal{X}$ onto the tensor subspace supported on Ω. Then $\mathcal{X}_\Omega$ can be defined as:

$$\mathcal{X}_\Omega = \begin{cases} \mathcal{X}_{i_1 i_2 \ldots i_N}, & (i_1, i_2, \ldots, i_N) \in \Omega \\ 0, & \text{otherwise.} \end{cases}$$

Then we can force the decomposition to match the observation data only at the available entries, and find the decomposition that minimizes the weighted cost of tensor rank and sparsity, leading to the following model:

$$\begin{aligned} \min_{\mathcal{X}, \mathcal{E}} \quad & \sum_{i=1}^{N} \|\mathbf{X}_{(i)}\|_* + \lambda \|\mathbf{E}_{(1)}\|_{2,1} \\ \text{s.t.} \quad & \mathcal{B}_\Omega = (\mathcal{X} + \mathcal{E})_\Omega. \end{aligned} \tag{7}$$

Note that related problems to Equation (7) for the matrix setting are addressed in [8, 12].

4.2 Algorithm

In this section, we develop algorithm for tensor decomposition with fiber-wise corruption model formulated in Section 4.1. We first solve Equation (6) for the full-observation setting, then for the partial-observation setting Equation (7), adopting an ADMM method [17] for each.

4.2.1 Higher-order RPCA. Problem (6) is difficult to solve because the terms $\|\mathbf{X}_{(i)}\|_*$ in the objective function are interdependent, since each $\mathbf{X}_{(i)}$ is unfolded from the same tensor $\mathcal{X}$. Alternatively, we split $\mathcal{X}$ into N auxiliary variables, $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_N \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, and rewrite (6) as:

$$\begin{aligned} \min_{\mathcal{X}_i, \mathcal{E}} \quad & \sum_{i=1}^{N} \|\mathbf{X}_{i(i)}\|_* + \lambda \|\mathbf{E}_{(1)}\|_{2,1} \\ \text{s.t.} \quad & \mathcal{B} = \mathcal{X}_i + \mathcal{E}, i = 1, 2, \ldots, N, \end{aligned} \tag{8}$$

where $\mathbf{X}_{i(i)}$ are the unfoldings of $\mathcal{X}_i$ in the i^{th} mode. The N constraints $\mathcal{B} = \mathcal{X}_i + \mathcal{E}$ ensure that $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_N$ are all equal to the original $\mathcal{X}$ in problem (6).

Next, we proceed to solve problem (8) via an ADMM algorithm. A full explanation of the general ADMM framework can be found in [6]. The corresponding augmented Lagrangian function for problem (8) is

$$\begin{aligned} \mathcal{L}(\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_N, \mathcal{E}, \mathcal{Y}_1, \mathcal{Y}_2, \ldots, \mathcal{Y}_N; \mu) = & \sum_{i=1}^{N} \|\mathbf{X}_{i(i)}\|_* + \lambda \|\mathbf{E}_{(1)}\|_{2,1} \\ & + \sum_{i=1}^{N} \left(\frac{\mu}{2} \|\mathcal{X}_i + \mathcal{E} - \mathcal{B})\|_F^2 - \langle \mathcal{Y}_i, \mathcal{X}_i + \mathcal{E} - \mathcal{B} \rangle \right). \end{aligned}$$

Here $\mathcal{Y}_i$ are the Lagrange multipliers, and μ is a positive scalar.

Under the ADMM framework, the approach is to iteratively update the three sets of variables $(\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_N), \mathcal{E}, (\mathcal{Y}_1, \mathcal{Y}_2, \ldots, \mathcal{Y}_N)$. To be specific, at the start of the $k + 1^{\text{th}}$ iteration, we fix $\mathcal{E} = \mathcal{E}^k$ and $\mathcal{Y}_i = \mathcal{Y}_i^k$, then for each i solve:

$$\mathcal{X}_i^{k+1} = \underset{\mathcal{X}_i}{\operatorname{argmin}} \, \mathcal{L}(\mathcal{X}_i, \mathcal{E}^k, \mathcal{Y}_i^k; \mu). \tag{9}$$

Then, we fix $\mathcal{X}_i = \mathcal{X}_i^{k+1}$ and $\mathcal{Y}_i = \mathcal{Y}_i^k$ to solve:

$$\mathcal{E}^{k+1} = \underset{\mathcal{E}}{\operatorname{argmin}}\ \mathcal{L}(\mathcal{X}_i^{k+1}, \mathcal{E}, \mathcal{Y}_i^k; \mu). \tag{10}$$

Finally we fix $\mathcal{X}_i = \mathcal{X}_i^{k+1}$ and $\mathcal{E} = \mathcal{E}^{k+1}$, and update $\mathcal{Y}_i^k$:

$$\mathcal{Y}_i^{k+1} = \mathcal{Y}_i^k + \mu\left(\mathcal{B} - \mathcal{X}_i^{k+1} - \mathcal{E}^{k+1}\right). \tag{11}$$

Next we derive closed form solutions for problem (9) and for problem (10). Problem (9), written out, reads:

$$\mathcal{X}_i^{k+1} = \underset{\mathcal{X}_i}{\operatorname{argmin}}\ \|\mathbf{X}_{i(i)}\|_* + \frac{\mu}{2}\|\mathcal{X}_i + \mathcal{E}^k - \mathcal{B})\|_F^2 - \left\langle \mathcal{Y}_i^k, \mathcal{X}_i + \mathcal{E}^k - \mathcal{B}\right\rangle. \tag{12}$$

Using the property of the Frobenius norm, $\|\mathcal{A}_1 + \mathcal{A}_2\|_F^2 = \|\mathcal{A}_1\|_F^2 + \|\mathcal{A}_2\|_F^2 + 2\langle \mathcal{A}_1, \mathcal{A}_2\rangle$, problem (12) can be written as:

$$\begin{aligned} \mathcal{X}_i^{k+1} = &\ \underset{\mathcal{X}_i}{\operatorname{argmin}}\ \|\mathbf{X}_{i(i)}\|_* + \frac{\mu}{2}\left\|\frac{1}{\mu}\mathcal{Y}_i^k + \mathcal{B} - \mathcal{E}^k - \mathcal{X}_i\right\|_F^2 \\ = &\ \underset{\mathcal{X}_i}{\operatorname{argmin}}\ \|\mathbf{X}_{i(i)}\|_* + \frac{\mu}{2}\left\|\frac{1}{\mu}\mathbf{Y}_{i(i)}^k + \mathbf{B}_{(i)} - \mathbf{E}_{(i)}^k - \mathbf{X}_{i(i)}\right\|_F^2. \end{aligned} \tag{13}$$

In the second line of Equation (13), we change the Frobenius norm of a tensor into the Frobenius norm of its i-th unfolding, which does not change the actual value of the norm. As a result, the objective function of problem (13) only involves matrices, so we can solve for $\mathbf{X}_{i(i)}$ using the well-established closed form solution (e.g., see proof in Cai et al. [7]): $\mathbf{X}_{i(i)}^{k+1} = \mathbf{T}_{\frac{1}{\mu}}(\frac{1}{\mu}\mathbf{Y}_{i(i)}^k + \mathbf{B}_{(i)} - \mathbf{E}_{(i)}^k)$. Then we fold the matrix $\mathbf{X}_{i(i)}^{k+1}$ back into a tensor, i.e., $\mathcal{X}_i^{k+1} = \text{fold}_i(\mathbf{X}_{i(i)}^{k+1})$. The truncation operator $\mathbf{T}_\tau(\mathbf{X})$ for a matrix $\mathbf{X} = U\Sigma V^T$ is $\mathbf{T}_\tau(\mathbf{X}) = U\Sigma_{\bar{\tau}}V^T$, where $\Sigma = \text{diag}(\sigma_i)$ is the eigenvalue diagonal matrix for $\mathbf{X}$. The operation $\Sigma_{\bar{\tau}} = \text{diag}(\max(\sigma_i - \tau, 0))$ discards the eigenvalues less than τ, and shrinks the remaining eigenvalues by τ.

We now proceed to derive a closed form solution to update $\mathcal{E}$ in problem (10). Problem (10) is equivalent to solving:

$$\mathcal{E}^{k+1} = \underset{\mathcal{E}}{\operatorname{argmin}}\ \lambda\|\mathbf{E}_{(1)}\|_{2,1} + \sum_{i=1}^{N}(\mu\|\mathcal{X}_i^{k+1} + \mathcal{E} - \mathcal{B})\|_F^2 - \langle \mathcal{Y}_i^k, \mathcal{X}_i^{k+1} + \mathcal{E} - \mathcal{B}\rangle). \tag{14}$$

Following the same technique as earlier, (14) is equivalent to:

$$\mathcal{E}^{k+1} = \underset{\mathcal{E}}{\operatorname{argmin}}\ \lambda\|\mathbf{E}_{(1)}\|_{2,1} + \sum_{i=1}^{N}\left(\frac{\mu}{2}\left\|\frac{1}{\mu}\mathcal{Y}_i^k + \mathcal{B} - \mathcal{X}_i^{k+1} - \mathcal{E}\right\|_F^2\right). \tag{15}$$

By the proof of Goldfarb and Qin [17], problem (15) shares the same solution with:

$$\mathcal{E}^{k+1} = \underset{\mathcal{E}}{\operatorname{argmin}}\tilde{}\ \lambda\left\|\mathbf{E}_{(1)}\right\|_{2,1} + \frac{\mu N}{2}\left\|\mathcal{E} - \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{\mu}\mathcal{Y}_i^k + \mathcal{B} - \mathcal{X}_i^{k+1}\right)\right\|_F^2, \tag{16}$$

since they have the same first-order conditions. In order to simplify expression (16), we denote the term $\frac{1}{N}\sum_{i=1}^{N}(\frac{1}{\mu}\mathcal{Y}_i^k + \mathcal{B} - \mathcal{X}_i^{k+1})$ by a single variable $\mathcal{C} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. Thus,

$$\begin{aligned} \mathcal{E}^{k+1} = &\ \underset{\mathcal{E}}{\operatorname{argmin}}\ \lambda\|\mathbf{E}_{(1)}\|_{2,1} + \frac{\mu N}{2}\|\mathcal{E} - \mathcal{C}\|_F^2 \\ = &\ \underset{\mathcal{E}}{\operatorname{argmin}}\ \lambda\|\mathbf{E}_{(1)}\|_{2,1} + \frac{\mu N}{2}\|\mathbf{E}_{(1)} - \mathbf{C}_{(1)}\|_F^2, \end{aligned} \tag{17}$$

ALGORITHM 1: Tensor RPCA for fiber-wise corruptions

Given $\mathcal{B}, \lambda, \mu$. Initialize $\mathcal{X}_i = \mathcal{E} = \mathcal{Y}_i = \mathbf{0}$.
for $k = 0, 1, \ldots$ **do**
  for $i = 1 : N$ **do** ▷ Update $\mathcal{X}$
    $\mathbf{X}_{i(i)}^{k+1} = \mathbf{T}_{\frac{1}{\mu}}\left(\frac{1}{\mu}\mathbf{Y}_{i(i)}^{k} + \mathbf{B}_{(i)} - \mathbf{E}_{(i)}^{k}\right)$.
    $\mathcal{X}_i^{k+1} = \text{fold}_i\left(\mathbf{X}_{i(i)}^{k+1}\right)$
  end for
  $C = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{\mu}\mathcal{Y}_i^{k+1} + \mathcal{B} - \mathcal{X}_i^{k+1}\right)$ ▷ Update $\mathcal{E}$.
  for $j = 1, 2, \ldots, p$ **do**
    $\mathbf{E}_{(1)j}^{k+1} = \mathbf{C}_{(1)j}\max\left\{0, 1 - \frac{\lambda}{\mu N\|\mathbf{C}_{(1)j}\|_2}\right\}$
  end for
  $\mathcal{E}^{k+1} = \text{fold}_1(\mathbf{E}_{(1)}^{k+1})$
  for $i = 1 : N$ **do** ▷ Update $\mathcal{Y}$.
    $\mathcal{Y}_i^{k+1} = \mathcal{Y}_i^{k} + \mu(\mathcal{B} - \mathcal{X}_i^{k+1} - \mathcal{E}^{k+1})$.
  end for
end for
return $\mathcal{X}^k = \frac{1}{N}\sum_{i=1}^{N}\mathcal{X}_i^k, \mathcal{E}^k$

where in the second line we use the same approach as in (13) in which we replace the tensor Frobenius norm by the equivalent Frobenius norm of the mode one unfolding. The objective function of problem (17) only involves matrices, and the closed form solution is [47]:

$$\mathbf{E}_{(1)j}^{k+1} = \mathbf{C}_{(1)j}\max\left\{0, 1 - \frac{\lambda}{\mu N\|\mathbf{C}_{(1)j}\|_2}\right\}, \text{ for } j = 1, 2, \ldots, p, \tag{18}$$

where $\mathbf{E}_{(1)j}$ is the j^{th} column of $\mathbf{E}_{(1)}$, $\mathbf{C}_{(1)j}$ is the j^{th} column of $\mathbf{C}_{(1)}$, and the integer $p = I_2 \times I_3 \times \cdots \times I_N$ is the total number of columns in $\mathbf{C}_{(1)}$. This operation effectively sets a column of $\mathbf{C}_{(1)}$ to zero if its l_2 norm is less than $\frac{\lambda}{\mu N}$, and scales the elements down by a factor $1 - \frac{\lambda}{\mu N\|\mathbf{C}_{(1)j}\|_2}$ otherwise [47].

Note that compared with the ADMM method where we just update $\mathcal{X}_i$ and $\mathcal{E}$ once, the *augmented Lagrangian multipliers* (ALM) method [31] seeks to find the exact solutions for primal variables $\mathcal{X}_i$ and $\mathcal{E}$ before updating Lagrangian multipliers $\mathcal{Y}_i = \mathcal{Y}_i^k$, yielding the framework as

$$(\mathcal{X}_i^{k+1}, \mathcal{E}^{k+1}) = \underset{\mathcal{X}_i, \mathcal{E}}{\text{argmin}}\ \mathcal{L}(\mathcal{X}_i, \mathcal{E}, \mathcal{Y}_i^k; \mu)$$
$$\mathcal{Y}_i^{k+1} = \mathcal{Y}_i^k + \mu(\mathcal{B} - \mathcal{X}_i^{k+1} - \mathcal{E}^{k+1}).$$

As pointed out by Lin et al. [31], compared with ALM, not only is ADMM still able to converge to the optimal solution for $\mathcal{X}_i$ and $\mathcal{E}$, but also the speed performance is better. It is also noted that while in ALM, the $\mathcal{X}$ and $\mathcal{E}$ are optimized jointly, in the ADMM implementation, they are in fact updated sequentially [6]. It is often observed in the matrix settings (see e.g., Lin et al. [31]) that updating the term containing outliers before the low rank term (i.e., $\mathcal{E}$ before $\mathcal{X}$ in the tensor setting) the low rank term results in faster convergence. As a consequence this is the approach followed in the numerical implementation of Algorithm 1 used later in this work.

In the implementation of Algorithm 1, we set the convergence criterion as:

$$\frac{\|\mathcal{B} - \mathcal{E} - \mathcal{X}\|_F}{\|\mathcal{B}\|_F} \leq \epsilon, \tag{19}$$

Where ϵ is the tolerance.

4.2.2 Partial Observation. Now we provide an algorithm to solve problem (7). Similar to the matrix setting in Tang and Nehorai [47], we set the fibers of the low-rank tensor to be zero in the locations corresponding to outliers. We introduce a compensation tensor $\mathcal{O} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, which is zero for entries in the observation set Ω, and can take any value outside Ω. Thus using the same auxiliary variables technique as in Equation (8), we can reformulate problem (7) as:

$$\begin{aligned} \min_{\mathcal{X}_i, \mathcal{E}} \quad & \sum_{i=1}^{N} \|\mathbf{X}_{i(i)}\|_* + \lambda \|\mathbf{E}_{(1)}\|_{2,1} \\ \text{s.t.} \quad & \mathcal{B} = \mathcal{X}_i + \mathcal{E} + \mathcal{O}, i = 1, 2, \ldots, N, \\ & \mathcal{O}_\Omega = 0. \end{aligned} \tag{20}$$

Since $\mathcal{O}$ compensates for whatever the value is in the unobserved entries of $\mathcal{B}$, we only need to keep track of the indices of the unobserved entries, and can simply set the unobserved entries of $\mathcal{B}$ to zero. The augmented Lagrangian function for problem (20) is:

$$\begin{aligned} \mathcal{L}(\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_N, \mathcal{E}, \mathcal{O}, \mathcal{Y}_1, \mathcal{Y}_2, \ldots, \mathcal{Y}_N; \mu) = & \sum_{i=1}^{N} \|\mathbf{X}_{i(i)}\|_* + \lambda \|\mathbf{E}_{(1)}\|_{2,1} \\ & + \sum_{i=1}^{N} \left(\frac{\mu}{2} \|\mathcal{X}_i + \mathcal{E} + \mathcal{O} - \mathcal{B})\|_F^2 - \langle \mathcal{Y}_i, \mathcal{X}_i + \mathcal{E} + \mathcal{O} - \mathcal{B} \rangle \right). \end{aligned}$$

We again use the ADMM framework now iteratively updating $\mathcal{X}_i$, $\mathcal{E}$, $\mathcal{O}$ and $\mathcal{Y}_i$. The proof of the closed form solution for updating $\mathcal{X}_i$, $\mathcal{E}$ and $\mathcal{Y}_i$ is similar to Algorithm 1. For $\mathcal{O}$, we fix $\mathcal{X}_i = \mathcal{X}_i^{k+1}$, $\mathcal{E} = \mathcal{E}^{k+1}$ and $\mathcal{Y}_i = \mathcal{Y}_i^k$, to solve Equation (21):

$$\begin{aligned} \mathcal{O}^{k+1} = \quad \underset{\mathcal{O}}{\text{argmin}} \quad & \mathcal{L}(\mathcal{X}_i^{k+1}, \mathcal{E}^{k+1}, \mathcal{O}, \mathcal{Y}_i^k; \mu) \\ \text{s.t.} \quad & \mathcal{O}_\Omega = 0. \end{aligned} \tag{21}$$

Following the same procedure as before (see Equations (14)–(16)), we have:

$$\begin{aligned} \mathcal{O}^{k+1} &= \underset{\mathcal{O}}{\text{argmin}} \quad \sum_{i=1}^{N} \left(\frac{\mu}{2} \left\| \mathcal{X}_i^{k+1} + \mathcal{E}^{k+1} + \mathcal{O} - \mathcal{B}) \right\|_F^2 - \langle \mathcal{Y}_i^k, \mathcal{X}_i^{k+1} + \mathcal{E}^{k+1} + \mathcal{O} - \mathcal{B} \rangle \right) \\ &= \underset{\mathcal{O}}{\text{argmin}} \quad \sum_{i=1}^{N} \left(\frac{\mu}{2} \left\| \frac{1}{\mu} \mathcal{Y}_i^k + \mathcal{B} - \mathcal{X}_i^{k+1} - \mathcal{E}^{k+1} - \mathcal{O} \right\|_F^2 \right) \\ &= \underset{\mathcal{O}}{\text{argmin}} \quad \frac{\mu N}{2} \left\| \mathcal{O} - \frac{1}{N} \sum_{i=1}^{N} \left(\frac{1}{\mu} \mathcal{Y}_i^k + \mathcal{B} - \mathcal{X}_i^{k+1} - \mathcal{E}^{k+1} \right) \right\|_F^2, \\ &\quad \text{s.t.} \quad \mathcal{O}_\Omega = 0. \end{aligned} \tag{22}$$

For Equation (22), we simply set $\mathcal{O} = \frac{1}{N} \sum_{i=1}^{N} (\frac{1}{\mu} \mathcal{Y}_i^k + \mathcal{B} - \mathcal{X}_i^{k+1} - \mathcal{E}^{k+1})$ for entries $(I_1, I_2, \ldots, I_N) \in \Omega^C$, and zero otherwise. The procedure is summarized in Algorithm 2.

ALGORITHM 2: ADMM for robust tensor completion

Given $\mathcal{B}, \lambda, \mu$. Initialize $\mathcal{X}_i = \mathcal{E} = \mathcal{Y}_i = \mathcal{O} = \mathbf{0}$.
for $k = 0,1, \cdots$ **do**
  for $i = 1{:}\mathrm{N}$ **do** ▷ Update $\mathcal{X}$
    $\mathbf{X}_{i(i)}^{k+1} = \mathbf{T}_{\frac{1}{\mu}}(\mathbf{B}_{(i)} + \frac{1}{\mu}\mathbf{Y}_{i(i)}^{k} - \mathbf{E}_{(i)}^{k} - \mathbf{O}_{(i)}^{k})$,
    $\mathcal{X}_i^{k+1} = \mathrm{fold}_i(\mathbf{X}_{i(i)}^{k+1})$
  end for
  $\mathcal{C} = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{1}{\mu}\mathcal{Y}_i^{k+1} + \mathcal{B} - \mathcal{X}_i^{k+1} - \mathcal{O}^{k+1}\right)$ ▷ Update $\mathcal{E}$.
  for $j = 1, 2, \ldots, p$ **do**
    $\mathbf{E}_{(1)j}^{k+1} = \mathbf{C}_{(1)j}\max\left\{0, 1 - \frac{\lambda}{\mu N \|\mathbf{C}_{(1)j}\|_2}\right\}$
  end for
  $\mathcal{E}^{k+1} = \mathrm{fold}_1(\mathbf{E}_{(1)}^{k+1})$
  $\mathcal{O}^{k+1} = \left(\sum_{i=i}^{N}\left(\frac{1}{\mu}\mathcal{Y}_i^{k} + \mathcal{B} - \mathcal{X}_i^{k+1} - \mathcal{E}^{k+1}\right)\right)_{\Omega^C}$. ▷ Update $\mathcal{O}$.
  for $i = 1 : N$ **do** ▷ Update $\mathcal{Y}$.
    $\mathcal{Y}_i^{k+1} = \mathcal{Y}_i^{k} + \mu(\mathcal{B} - \mathcal{X}_i^{k+1} - \mathcal{E}^{k+1} - \mathcal{O}^{k+1})$
  end for
end for
return $\mathcal{X}^k = \frac{1}{N}\sum_{i=1}^{N}\mathcal{X}_i^k, \mathcal{E}^k$

We set the convergence criterion of Algorithm 2 as

$$\frac{\|\mathcal{B} - \mathcal{E} - \mathcal{X} - \mathcal{O}\|_F}{\|\mathcal{B}\|_F} \leq \epsilon,$$

which is similar to Equation (19) but accounts for the additional tensor $\mathcal{O}$.

The convergence of Algorithms 1 and 2 is guaranteed. The proposed algorithms are special cases of the ADMM framework. The convergence guarantee thus follows the ADMM algorithm, which is established in [27, 32] for example. We also refer to works [31, 60] for more discussion on convergence of ADMM algorithms for matrix low rank and sparse decompositions.

In terms of computational cost, we note that each iteration is dominated by the singular value decomposition used when updating the low-rank tensor. Because the SVD is the computationally expensive step, in our source code, we use an efficient implementation from [29]. As we show in Section 5, only a few iterations (i.e., 10–40) are sufficient to achieve good precision, consistent with the observation in [8] (for matrices).

5 NUMERICAL EXPERIMENTS

In this section, we apply Algorithms 1 and 2 developed in Section 4.2 on a series of test problems using synthetically generated datasets. We first conduct tensor decomposition on fiber-wise corrupted data, then we examine the case when the data are only partially observed and are also fiber-wise corrupted. For the fully observed case, we compare our approach with l_1 norm constrained decomposition [17, 46]. For the partially observed case, in addition to the l_1 norm constrained decomposition, several other state-of-the-art tensor completion methods are also used for comparison: STDC [13], FBCP [61], W-ST, and C-ST [59]. We demonstrate via the numerical experiments that under fiber-wise gross corruption and partial observation, our method provides an exact recovery and is several orders of magnitude more accurate than the other approaches.

5.1 Performance Measures and Implementation

For each of the numerical experiments, the performance of the algorithms are measured by the relative error (RE) of the low-rank tensor, as well as the precision and recall of the outlier fibers. The RE of low-rank tensor is calculated as:

$$\text{RE} = \frac{\|\mathcal{X}_0 - \hat{\mathcal{X}}\|_F}{\|\mathcal{X}_0\|_F},$$

where $\mathcal{X}_0$ is the true low-rank tensor modified to take the value 0 in the entries corresponding to the corrupted fibers; $\hat{\mathcal{X}}$ is the estimated low-rank tensor resulting from application of Algorithm 1 or 2, which also has the value 0 in the fibers that are estimated to be corrupted.

We compute the *precision* of the algorithm to assess the potential to correctly identify only the outlier fibers. It is computed as:

$$\text{precision} = \frac{\text{tp}}{\text{tp} + \text{fp}},$$

where the *true positives* (tp) corresponds the number of estimated outlier fibers which are true outliers, and the *false positives* (fp) corresponds to the number of estimated outlier fibers which are not true outliers.

The *recall*, which measures the ability to find all outlier fibers, is defined as:

$$\text{recall} = \frac{\text{tp}}{\text{tp} + \text{fn}},$$

where the *false negatives* (fn) correspond to the number of true outlier fibers that were not correctly identified by the estimator.

For the convergence criterion, we set $\epsilon = 10^{-7}$, and we use an empirical value $\lambda = \frac{1}{0.03 I_m}$, where $I_m = \max(I_1, \ldots, I_N)$. The hyperparameter λ in l_1 norm constrained decomposition algorithm is also tuned for its best performance in our settings.

All of the experiments are carried out on a Macbook Pro with quad-core 2.7 GHz Intel i7 Processor and 16 GB RAM, running Matlab R2018a. We modify and extend the code of Lin et al. [31], using PROPACK toolbox [29] to efficiently calculate the SVD. The code is modified to update variables in line with the distinct problem formulation using the $l_{2,1}$ norm and to scale to tensors rather than matrices. The Tensor Toolbox for Matlab [4], [3] is also used for tensor manipulations. The resulting source code is available at https://github.com/Lab-Work/Robust_tensor_recovery_for_traffic_events.

5.2 Tensor RPCA

In this subsection, we apply higher-order RPCA to the problems where we have fully observed data with fiber-wise corrupted entries.

5.2.1 Simulation Conditions. We synthetically generate the observation data as $\mathcal{B} = \mathcal{X}_0 + \mathcal{E}_0 \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, where $\mathcal{X}_0$ and $\mathcal{E}_0$ are the *true* or "ground truth" low-rank tensor and fiber-sparse tensor, respectively. We generate $\mathcal{X}_0 \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ as a core tensor $\mathcal{G} \in \mathbb{R}^{c_1 \times c_2 \times c_3}$ with size $c_1 \times c_2 \times c_3$ and Tucker rank (c_1, c_2, c_3), multiplied in each mode by orthogonal matrices of corresponding dimensions, $\mathbf{U}^{(i)} \in \mathbb{R}^{I_i \times c_i}$:

$$\mathcal{X}_0 = \mathcal{G} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)}.$$

The entries of $\mathcal{G}$ are independently sampled from standard Gaussian distribution. The orthogonal matrices $\mathbf{U}^{(i)}$ are generated via a Gram-Schmidt orthogonalization on c_i vectors of size $\mathbb{R}^{I_i}$ drawn from standard Gaussian distribution. The sparse tensor $\mathcal{E}_0 \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ is formed by first generating a tensor $\mathcal{E}_0' \in \mathbb{R}^{I_1 \times I_2 \times I_3}$, whose entries are i.i.d uniform distribution $\mathcal{U}(0,1)$. Then we randomly keep

Table 1. Application of Algorithm 1 on Fiber-wise Corrupted Tensors with Full Observation, and Comparison with l_1 Norm Constrained Decomposition

(a) Algorithm 1: $l_{2,1}$ norm constrained decomposition

(I_1, I_2, I_3)	(c_1, c_2, c_2)	RE	precision	recall	iter	time(s)
(70,70,70)	(7,7,7)	1.23×10^{-7}	1.0	1.0	29	9.9
(90,90,90)	(9,9,9)	1.24×10^{-7}	1.0	1.0	28	16.2
(150,150,150)	(15,15,15)	6.68×10^{-8}	1.0	1.0	28	50.5
(210,210,210)	(21,21,21)	7.35×10^{-8}	1.0	1.0	28	133.0

(b) Comparison: l_1 norm constrained decomposition

(I_1, I_2, I_3)	(c_1, c_2, c_2)	RE	precision	recall	iter	time(s)
(70,70,70)	(7,7,7)	2.10×10^{-1}	1.0	1.0	28	1.4
(90,90,90)	(9,9,9)	2.28×10^{-1}	1.0	1.0	29	2.6
(150,150,150)	(15,15,15)	2.23×10^{-1}	1.0	1.0	28	14.7
(210,210,210)	(21,21,21)	2.27×10^{-1}	0.99	1.0	35	101.0

For different tensor sizes (I_1, I_2, I_3) and Tucker ranks (c_1, c_2, c_2), where we set $c = 0.1I$, we show the REs of low rank tensors (RE), the precision and recall of outlier fibers identification, as well as the number of iterations (iter) and total time for convergence.

a fraction γ of the fibers of $\mathcal{E}_0'$ to form $\mathcal{E}_0$. Finally, the corresponding fibers of $\mathcal{X}_0$ with respect to non-zero fibers in $\mathcal{E}_0$ are set to zero.

5.2.2 Algorithm Performance for Varying Problem Sizes. We apply Algorithm 1 on $\mathcal{B}$ of varying tensor sizes (I_1, I_2, I_3) and underlying Tucker rank (c_1, c_2, c_3), and predict $\hat{\mathcal{X}}$ and $\hat{\mathcal{E}}$ using Algorithm 1. We also apply l_1 norm constrained decomposition on the same settings. Table 1 compares the result. The corruption rate is set to 5%, i.e., $\gamma = 0.05$. In all cases, for our algorithm the relative residual errors are less than 10^{-6}, which is the same precision that we set for convergence tolerance. That is to say, we can exactly recover the low rank tensors in this setting. The precision and recall are both 1.0, indicating that the outlier detection is also exact. Similar to the observation of Candès' et al. [8], the iteration numbers tend to be constant (between 28 and 29 in this case) regardless of tensor size. This indicates that the number of of SVD computations might be limited and insensitive to the size, which is important since SVD is the computational bottleneck of the algorithm. Furthermore, this property is important to allow the problem to solve quickly even on datasets of moderate sizes, as will be shown in a case study in Section 6.

In comparison, although l_1 norm constrained decomposition can also detect outliers with high precision and recall, the relative residual errors are relatively high, on the order of 10^{-1}. This indicates that l_1 norm constrained decomposition can do an adequate job when the corruption ratio is low ($\gamma = 0.05$), but cannot achieve exact recoveries.

5.2.3 Influence of the Corruption Rate. Next, we investigate the performance of Algorithm 1 as the corruption ratio changes, and compare the result with l_1 norm constrained decomposition. We fix the low-rank tensor $\mathcal{X}_0$ at size $\mathbb{R}^{70\times70\times70}$ with a Tucker rank of $(5, 5, 5)$, then vary the gross corruption ratio γ from 0% to 60%. The results are shown in Figure 3 as an average over 10 trials. In this setting, we see that as long as the corruption ratio is below 0.47, Algorithm 1 can precisely recover the low rank tensor, and correctly identify the outlier fibers. On the other hand, the RE of l_1 norm constrained decomposition is constantly higher. After the corruption ratio exceeds 0.2, the estimation is no longer useful, with the RE exceeding 100%.

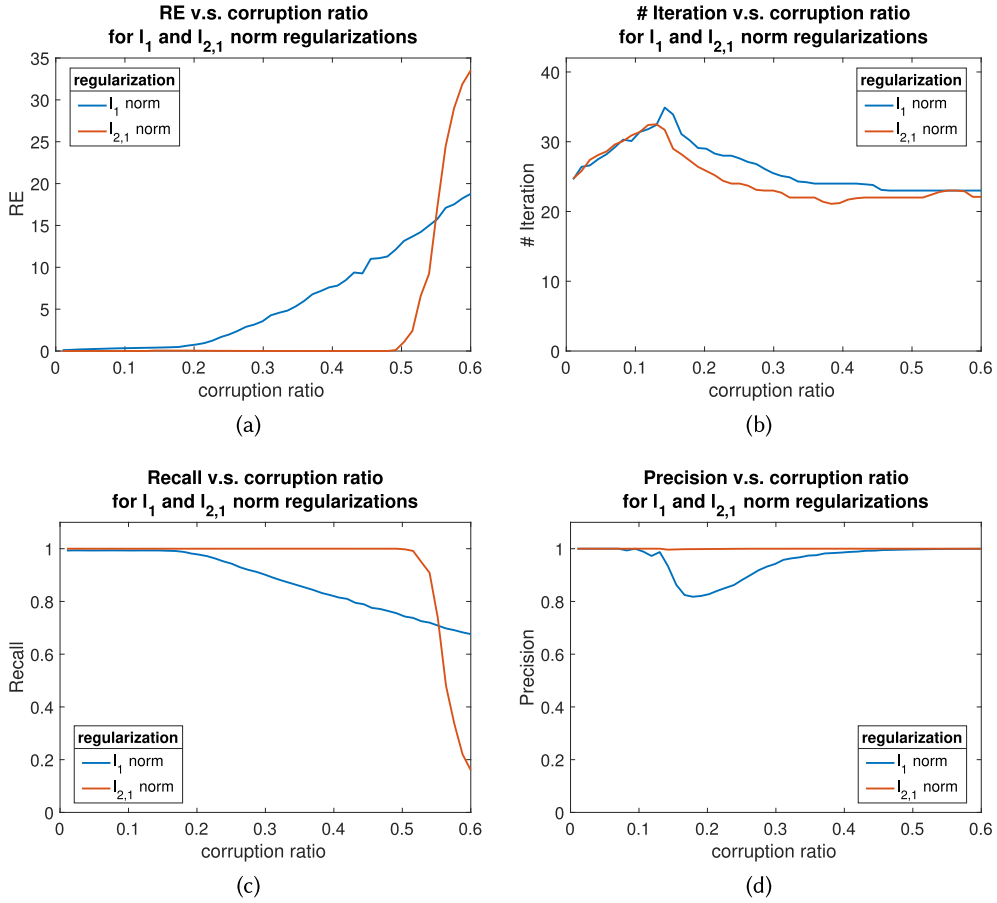

Fig. 3. Comparison of Algorithm 1 ($l_{2,1}$ norm regularized tensor decomposition), with l_1 norm regularized tensor decomposition, as gross corruption rate changes. We fix the tensor size at $\mathbb{R}^{70\times70\times70}$, and fix the Tucker rank of low-rank tensor $\mathcal{X}_0$ at $(5, 5, 5)$, then vary the gross corruption ratio γ from 0% to 100%. The result is an average over 10 trials.

5.3 Robust Tensor Completion

In this subsection, we look at the performance of Algorithm 2 when the data are only partially observed. We first compare it with several baseline tensor completion methods, then demonstrate how the performance of our method changes with varying conditions. We finally investigate sensitivity of our method regarding the algorithm hyper-parameter λ.

5.3.1 Simulation Conditions. We first generate full observation data $\mathcal{B}' = \mathcal{X}_0 + \mathcal{E}_0 \in \mathbb{R}^{I_1\times I_2\times I_3}$ in the same way as Section 5.2. $\mathcal{X}_0 \in \mathbb{R}^{I_1\times I_2\times I_3}$ is the low-rank tensor, and $\mathcal{E}_0 \in \mathbb{R}^{I_1\times I_2\times I_3}$ is the sparse tensor. Then, we form the partial observation data $\mathcal{B} \in \mathbb{R}^{I_1\times I_2\times I_3}$ by randomly keeping a fraction ρ of the entries in $\mathcal{B}'$. We record the indices of the unobserved entries, and set their values in $\mathcal{B}$ as 0.

5.3.2 Method Comparison. We compare the performance of Algorithm 2 ($l_{2,1}$ norm regularized tensor completion), with l_1 norm regularized tensor completion [17, 46], STDC [13], FBCP [61],

Table 2. Performance of Our Algorithm Compared to l_1 Norm Regularization [17, 46] and Tensor Completion Methods [13, 59, 61]

	corruption rate 0%			corruption rate 5%		
observation	100%	60%	20%	100%	60%	20%
our method	-	8.25×10^{-8}	5.38×10^{-1}	$\mathbf{1.30 \times 10^{-7}}$	$\mathbf{2.54 \times 10^{-7}}$	$\mathbf{6.75 \times 10^{-1}}$
l_1 norm [17, 46]	-	9.58×10^{-8}	5.54×10^{-1}	2.19×10^{-1}	4.76×10^{-1}	1.00
W-ST [59]	-	1.55×10^{-1}	6.51×10^{-1}	1.62×10^{-1}	1.84×10^{-1}	6.80×10^{-1}
C-ST [59]	-	2.43×10^{-1}	1.00	1.59×10^{-1}	2.71×10^{-1}	1.00
STDC [13]	-	1.01×10^{-3}	4.17×10^{-1}	5.16	4.33	4.13
FBCP [61]	-	$\mathbf{2.4 \times 10^{-9}}$	$\mathbf{6.33 \times 10^{-10}}$	4.63	4.21	9.12

RE is shown under different gross corruption rate and observation rate. We fix the tensor size at $\mathbb{R}^{50\times50\times50}$, and fix the Tucker rank of low-rank tensor $\mathcal{X}_0$ at (5, 5, 5).

W-ST [59], and C-ST [59], using source code provided in the manuscripts and also at [53]. For each method, algorithm hyperparameters are set according to the source paper, or tuned if improved performance can be obtained. We compare the methods under the setting where no gross corruption is present, and also in the setting in which gross corruption is present (i.e., at 5%). The results are shown in Table 2.

The main findings are as follows. In the setting where gross corruptions are present, our method offers the best performance over a range of missing data rates. Moreover, if at least 60% of the data are observed, and 5% of the entries are corrupted, our method provides an exact recovery and is several orders of magnitude more accurate than the other approaches. When gross corruptions are not present, method [61] outperforms all other approaches, while our method offers similar or better performance to [13] and [59]. These results illustrate the importance of designing algorithms to treat missing and grossly corrupted data, which is the main contribution of our approach.

5.3.3 Influence of the Corruption and Observation Ratios. In the following subsections, we investigate the performance of Algorithm 2 with varying conditions. First, we apply Algorithm 2 on simulated data with a varying corruption ratio and observation ratio. We fix the low-rank tensor $\mathcal{X}_0$ at size $\mathbb{R}^{70\times70\times70}$ with a Tucker rank of (5, 5, 5). For gross corruption ratio γ at 0.05, 0.1, 0.2, we vary the observation ratio ρ from 0.1 to 1 and run Algorithm 2. We run 10 times and average the results.

Figure 4 shows that for the detection of corrupted fibers, the recall stays at 1. The precision stays at 1 when ρ is above 0.6 but drops dramatically for smaller ρ. The RE of low-rank tensor is zero when the observation ratio $\rho > 0.6$ with corruption ratio $\gamma = 0.05$, and when the observation ratio $\rho > 0.8$ with $\gamma = 0.1$. We observe a phase-transition behavior, in that the decomposition is exact when the observation ratio ρ is above a critical threshold, but the performance drops dramatically below the threshold. This critical threshold on the observation ratio ρ varies for each case, namely, the algorithm can handle more missing entries as the number of outliers γ is reduced. But when the corruption ratio is too large, exact recovery is not guaranteed.

Overall, the performance is promising, since we can exactly identify the outlier positions and recover the low-rank tensor at non-corrupted entries, even with relatively a large missing ratio, for example when 5% of the fibers are corrupted and 40% of the data are missing.

5.3.4 Influence of the Observation Ratio and the Tensor Rank. Next we fix the low-rank tensor $\mathcal{X}_0$ at size $\mathbb{R}^{70\times70\times70}$ and gross corruption ratio $\gamma = 0.1$. For $\mathcal{X}_0$ of different Tucker ranks (2,2,2),(5,5,5),

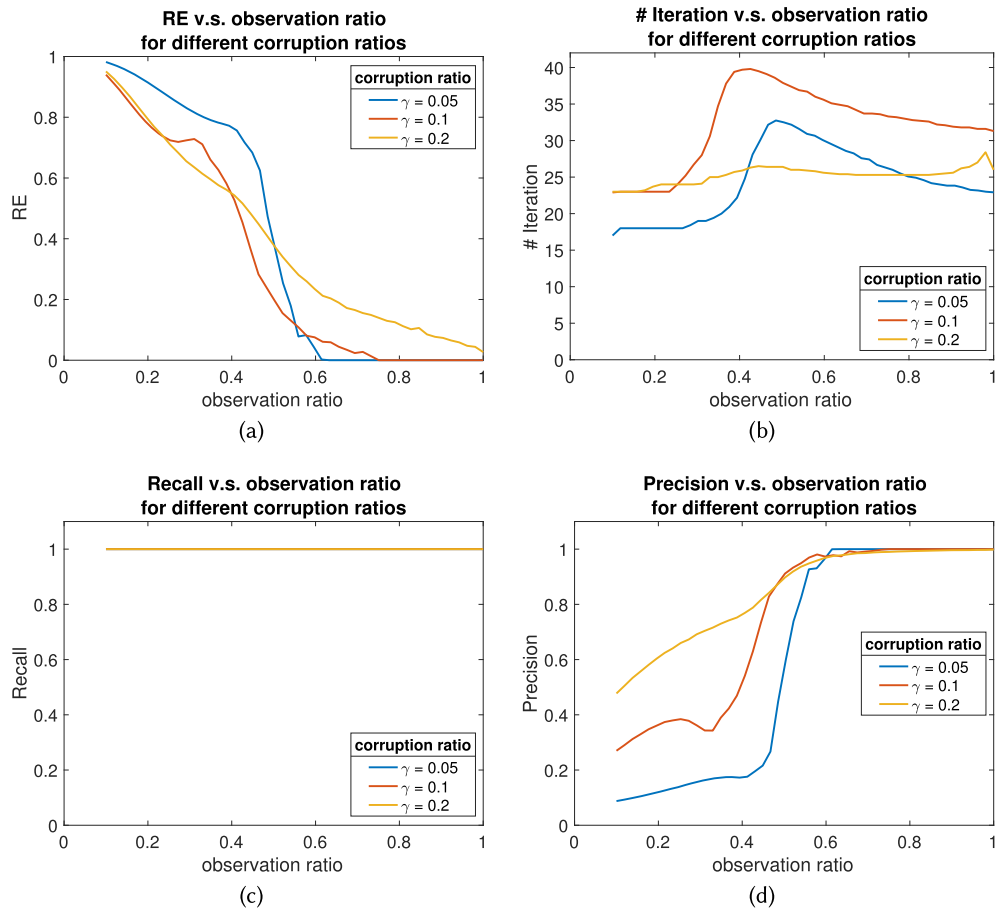

Fig. 4. Results of Algorithm 1 as a function of observation ratio with different corruption rates. The low-rank tensor $\mathcal{X}_0$ size is fixed at $\mathbb{R}^{70\times70\times70}$ with a Tucker rank of (5, 5, 5), and the gross corruption ratio γ is set at 0.05, 0.1, or 0.2. The result is an average over 10 trials.

and (8,8,8), we vary the observation ratio from 0.1 to 1 and run Algorithm 2. The result is shown in Figure 5, which is an average across 10 trials. Again, we observe a phase-transaction behavior, that when observation ratio is above a critical threshold, the decomposition is exact, with precision and recall at one, and RE at zero. For tensor ranks (5,5,5) and (8,8,8), this threshold is about 0.8. For tensor rank (2,2,2), it is lower, about 0.5. This indicates that when the underlying tensor rank is lower, we can exactly conduct the decomposition with even with a lower observation ratio.

5.3.5 Phase Transition Behavior. Now, we further study the phase transition property of Algorithm 2 in terms of the observation ratio and the tensor rank. We fix the gross corruption ratio at $\gamma = 0.1$ and the tensor size at $\mathbb{R}^{70\times70\times70}$. Then we vary the observation ratio from 0.3 to 1, and the Tucker rank of $\mathcal{X}_0$ from (1,1,1) to (20,20,20). For each combination we conduct 10 trials. Figure 6 shows the success rate out of 10 trials for varying tensor rank and observation ratio. We regard a trial *successful* if both the precision and recall of the outlier location identification

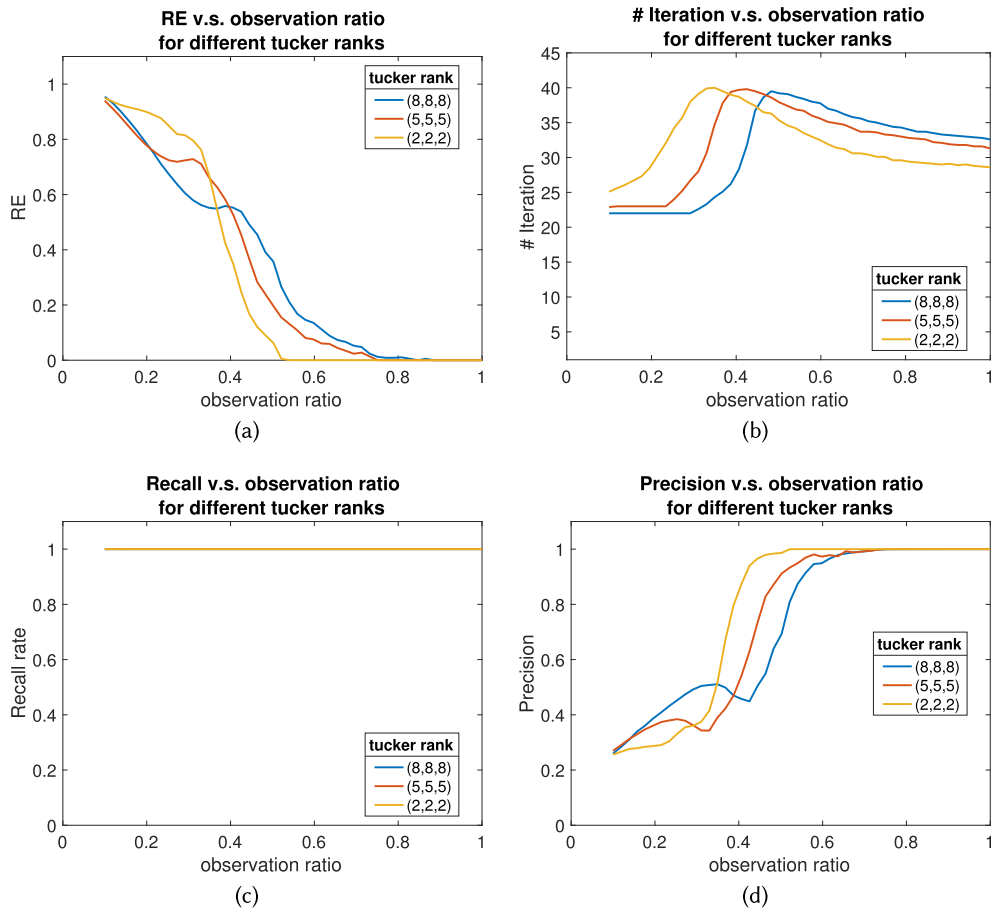

Fig. 5. Results of Algorithm 2 as a function of the observation ratio considering $\mathcal{X}_0$ with varying Tucker rank. The low-rank tensor $\mathcal{X}_0$ is generated with a Tucker rank of (2,2,2), (5,5,5), and (8,8,8) respectively, with a fixed size of $\mathbb{R}^{70\times70\times70}$, and a gross corruption ratio $\gamma = 0.1$. The plotted result is an average over 10 trials.

are greater than 0.99. The result shows that the possibility of success rises as observation ratio increases and Tucker rank of $\mathcal{X}_0$ decreases. For observation ratios greater than 0.7 and Tucker rank smaller than (5,5,5), the outlier identification is always successful.

5.4 Parameter Sensitivity

Since the hyperparameter λ is empirically chosen, we conduct a series of experiments to investigate the sensitivity regarding λ. To show the effective value of λ under different tensor sizes, we vary the tensor size from (50, 50, 50) to (200, 200, 200), and conduct tensor recovery experiments with different λ. The results (Figure 7) show that there is a range of λ that can achieve exact recovery, and that as tensor size increases, the value of λ should decrease. We have thus chosen an empirical value $\lambda = \frac{1}{0.03I_m}$ in our simulation experiments, which is sufficiently simple and falls in the range. We note that one can further adjust/optimize λ to achieve desired results if problem specific information is known.

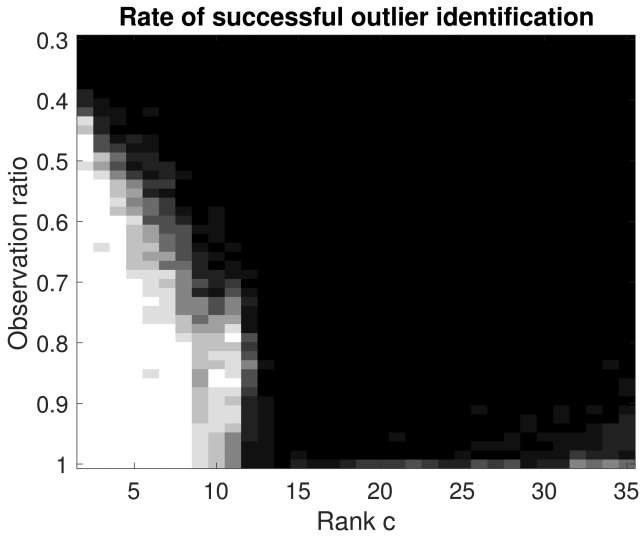

Fig. 6. Rate of successful outlier identification across 10 trials. The color denotes the rate of success. For each trial we create a tensor with size $\mathbb{R}^{70\times70\times70}$ and Tucker rank (c,c,c) (x-axis), fiber-wise corrupt it at ratio $\gamma = 0.1$. We vary the observation ratio *rho* from 0.3 to 1 (y-axis).

6 CASE STUDY: NASHVILLE, TN TRAFFIC DATASET

In this section, we apply our proposed method to a dataset of real-traffic data and use it to detect traffic events. We use the traffic speed data of downtown Nashville from January 1 to April 29, 2018 obtained from a large-scale traffic aggregator. Given that this is a real empirical dataset, we do not have access to the true low-rank traffic conditions and the true outliers. As a consequence, it is not possible to evaluate the precision and recall of the outlier detection algorithm as was done in the numerical examples in the previous section. Nevertheless, our algorithm can mark the events that are confirmed to be severe car crashes, construction lane closures, or large events that caused significant disruption on traffic of downtown Nashville.

We select a subset of road segments within downtown Nashville area that regularly have traffic data available. The base traffic dataset consists of the 1-hour average speed of traffic on each road segment in the network. The dataset has an observation ratio of 0.807, and records 556 road segments for 17 weeks, every week containing $24 \times 7 = 168$ hours, for a total of 2,856 hours. We can thus construct a data tensor of size $556 \times 168 \times 17$. The map of the final-selected road segments are shown in Figure 8, containing major interstate highways I-40, I-24 and I-440, among other major surface streets.

Since the data are not fully observed, we adopt the robust tensor completion algorithm (Algorithm 2). We set $\lambda = 1.47$, leading to a corruption ratio of 1.18%. This means over the 17 weeks (2,856 hours), 36 hours are marked as abnormal. Algorithm 2 takes 19 seconds to run on this dataset. Figure 9 plots the timeline when these outlier events take place.

Next, we investigate the outlier events identified by Algorithm 2. Out of the 36 hours detected as abnormal, 31 can be easily matched to recorded incidents. This includes construction lane closures, car crashes, and large events like the annual St. Jude Rock 'N' Roll Marathon [36]. This process is done by manually comparing the events identified by Algorithm 2 with the

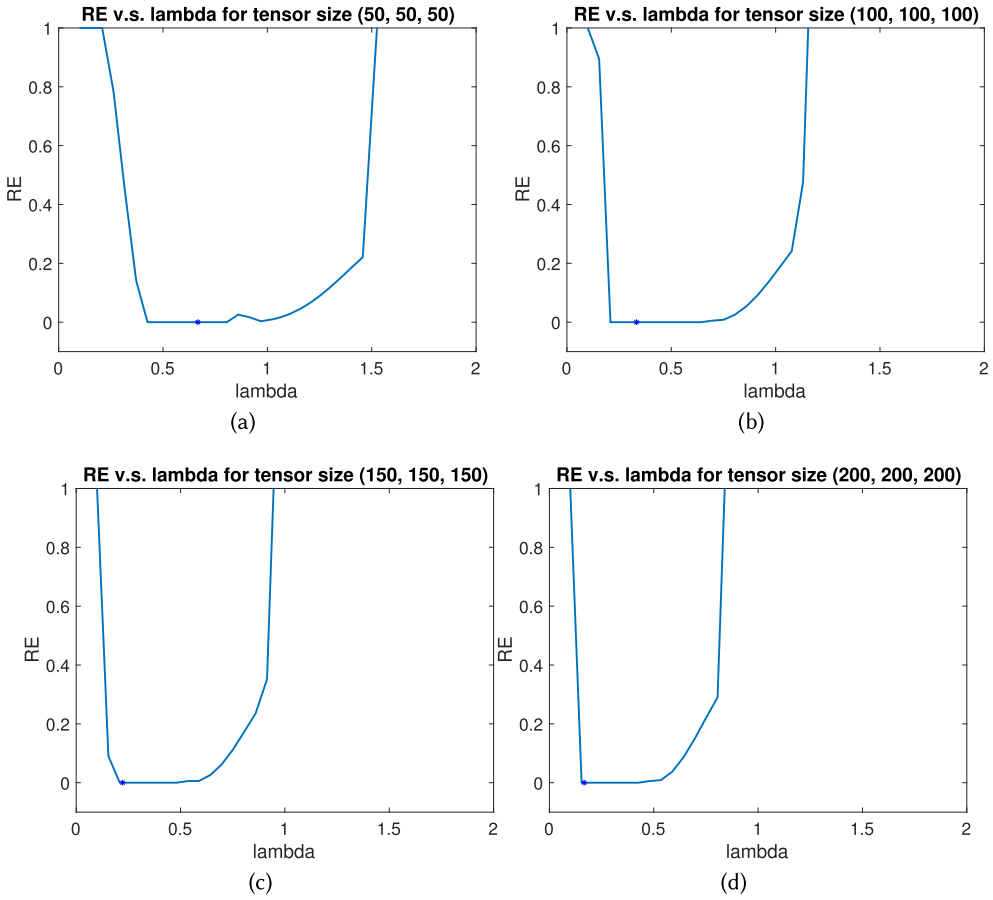

Fig. 7. Sensitivity of parameter λ for different tensor sizes. The gross corruption ratio γ is set at 0.05, and observation ratio is set at 80%. For each tensor size (I, I, I), we set Tucker rank of low-rank tensor at (c,c,c), where c = 0.1 I.

accident records of the Nashville fire department [35], and lane closure records of the *Tennessee Department of Transportation* [38], which is the state transportation authority. Most incidents clear out after 1 hour, while some last for 2 hours or more. For the five outlier hours that do not immediately correspond to events logged by authorities, we observe that the average speed of of many roads appear faster than normal, corresponding to abnormally light traffic conditions.

Figures 10 and 11 visualize some outlier events for illustration. In each figure, the heat maps in the right column show the average speed of the road during each hour, and the left column shows how many standard deviations the road segment is from the average speed of that hour. We calculate the average speed by looking at the low-rank matrix $\hat{\mathcal{X}}$ of Algorithm 2, which is expected to be the normal traffic pattern, and calculate the mean speed of 17 weeks for every hour of the week.

Figure 10 shows an event that corresponds to a series of car crashes. At noon on March 2, 2019, several severe car crashes occurred. The crashes occurred on a major freeway and an important arterial in Nashville, namely, Interstate 40 (labeled 1 in the second row of Figure 10) and Charlotte

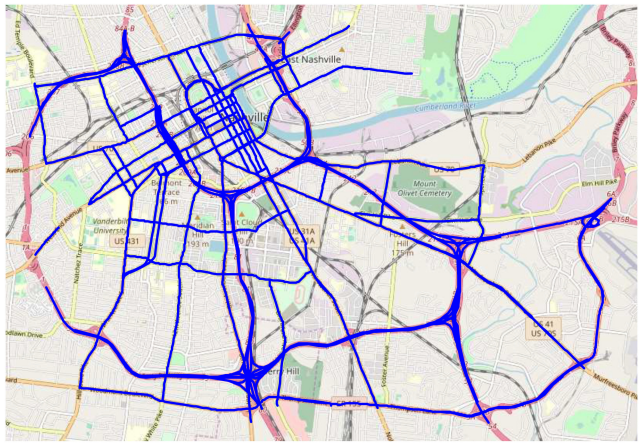

Fig. 8. Map of downtown Nashville. The studied road segments are marked in blue and consist of the major freeways and surface streets.

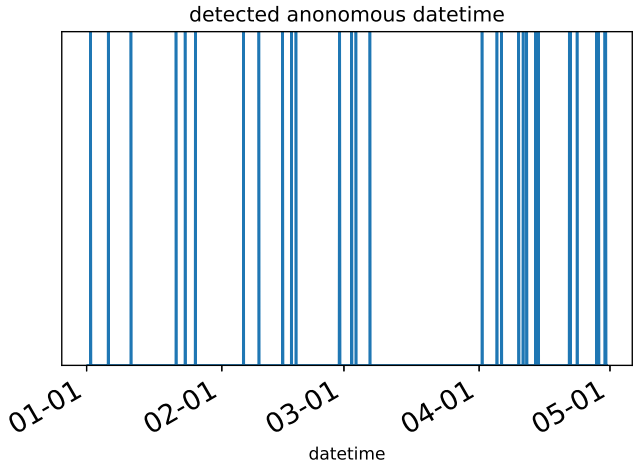

Fig. 9. Stem plot of detected outlier events. Each column indicates the time when an detected outlier event takes place.

Avenue (labeled 2 in the second row), as well as on and Carroll street (labeled 3 in the second row). At about 13:00, two other car crashes occurred on different segments of Interstate 40 (labeled 4 and 5 in the third row of Figure 10) [35]. The sequence of severe crashes created unusual congestion for that time of day, and took 2 hours to clear out. The corresponding hours 12:00 and 13:00 are detected by our method as outliers. We clarify that while we marked the locations identified to be car crashes, we also observe that other road segments are slower than usual, even though we do not know the underlying cause. The fact that we observe outliers on other portions of the network at the same time is in fact consistent with the design of the $l_{2,1}$ norm to find hours in which large portions of the network are anomalous.

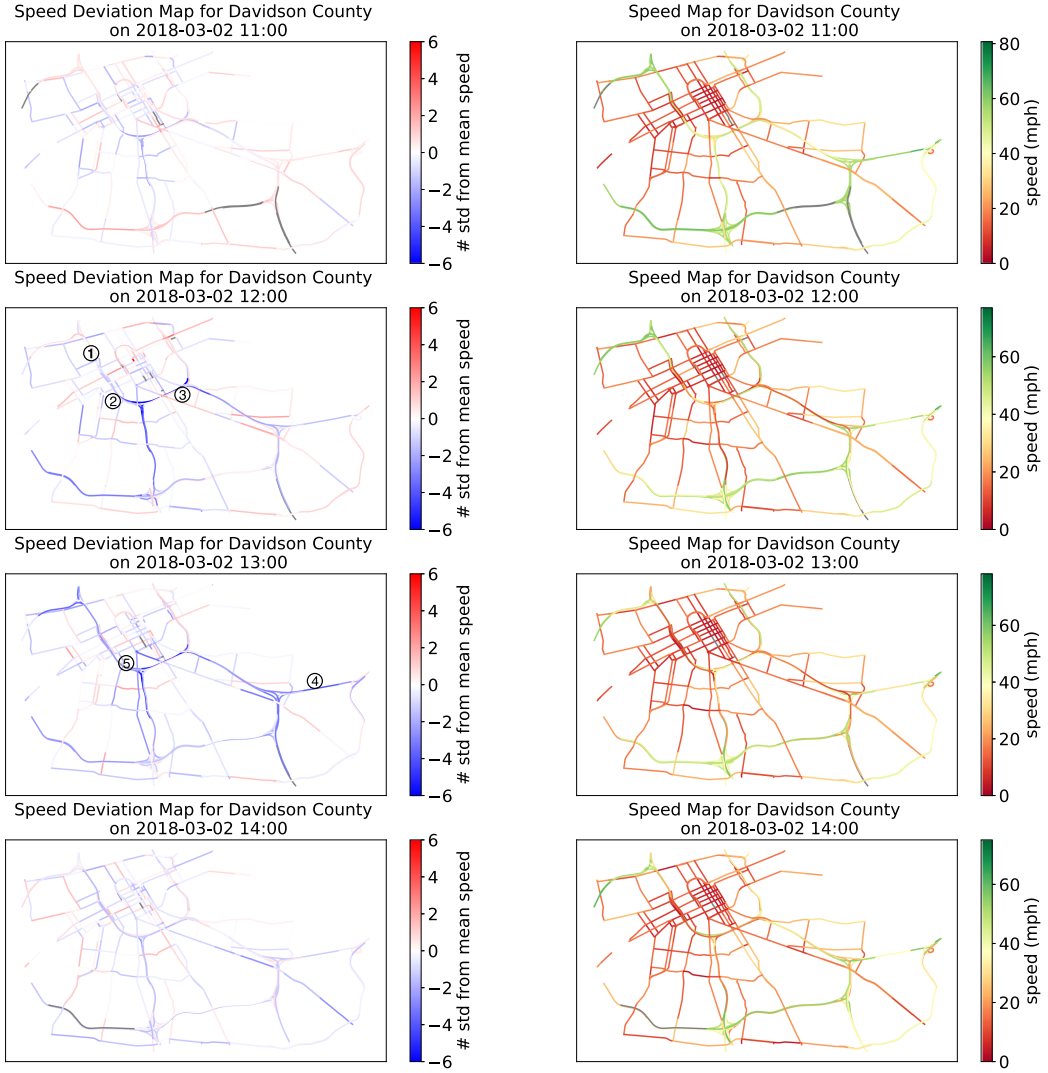

Fig. 10. Detected car crashes. The second and third row, i.e., 12:00 and 13:00 are marked as outliers. At 12:00, several severe car crashes happened on Interstate 40 (area 1 in the second row), Charlotte Avenue (area 2 in the second row), and Carroll street (area 3 in the second row). At about 13:00, two other car crashes occur at different segments of Interstate 40 (areas 4 and 5 in the third row).

Figure 11 shows a detected construction event. From 20:00 Tuesday evening through 5:00 the following day, there were road closures for bridge rehabilitation, resurfacing, and maintenance on the major freeways around downtown Nashville, namely, Interstate 24 (the north-south route marked as 1), Interstate 40 (the east-west route marked as 2), as well as streets connecting to Interstate 24 [37]. The first 2 hours of the lane closure, i.e., 20:00 and 21:00, observed the most congestion and were detected as outliers. The late night hours of Tuesday evening and the early Wednesday morning hours did not experience significant congestion and were consequently not detected as outliers.

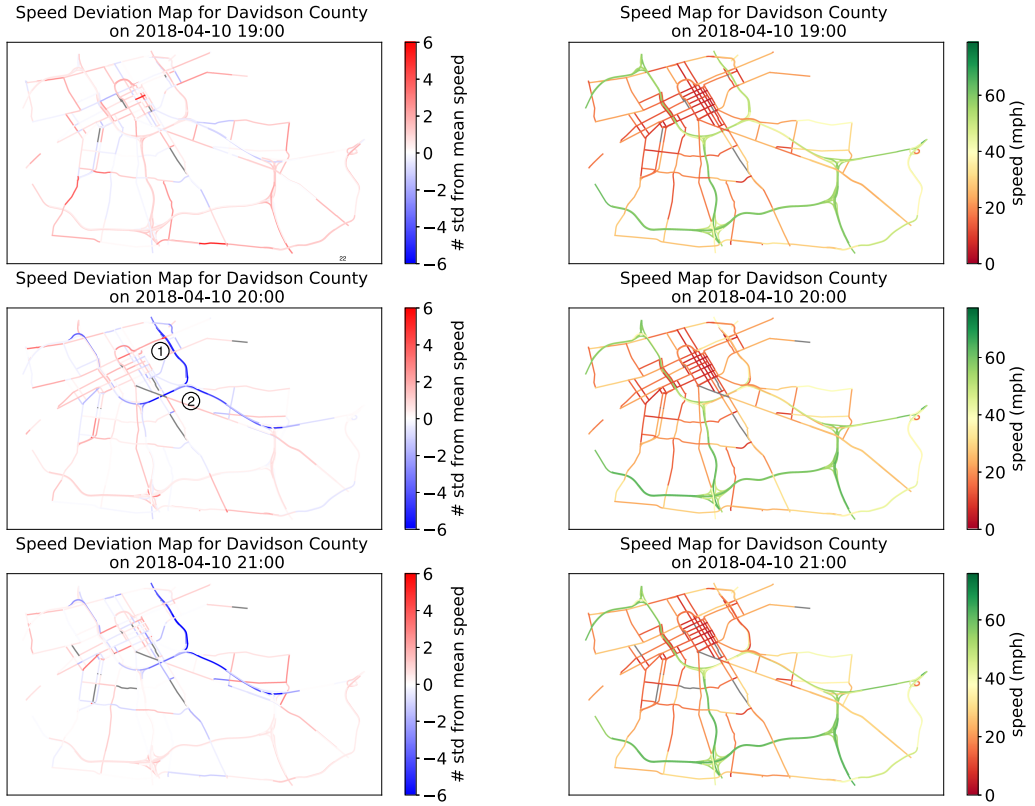

Fig. 11. Detected road closure. The second and third row, i.e., 20:00 and 21:00 are marked as outliers. It is the time when segments of Interstate 24 (the north-south route marked as 1), Interstate 40 (the east-west route marked as 2), as well as streets connecting to Interstate 24, were closed for bridge rehabilitation, resurfacing, and maintenance.

7 CONCLUSIONS

In this work, we introduced a tensor completion problem to detect extreme traffic conditions that exploits the spatial and temporal structure of traffic patterns in cities. An algorithm was proposed to perform the detection even in the presence of missing data. The method was applied to numerical examples that demonstrate exact recovery of the underlying low-rank tensor is possible in a range of settings with corrupted and missing entries, with lower quality results achieved as the fraction of missing entries increases. A case study on traffic conditions in Nashville, TN, demonstrated the practical performance of the method.

One limitation of the proposed approach is that the method exploits linear relationships between the traffic patterns, and is not designed to capture nonlinear spatial and temporal relationships. In our future work, we are interested in exploring possible neural network extensions might generalize the outlier detection tools for more complex relationships.

REFERENCES

[1] Abdelmonem A. Afifi and Robert M. Elashoff. 1966. Missing observations in multivariate statistics I. Review of the literature. *Journal of the American Statistical Association* 61, 315 (1966), 595–604.

[2] Muhammad Tayyab Asif, Nikola Mitrovic, Justin Dauwels, and Patrick Jaillet. 2016. Matrix and tensor based methods for missing data estimation in large traffic networks. *IEEE Transactions on Intelligent Transportation Systems* 17, 7 (2016), 1816–1825.

[3] Brett W. Bader and Tamara G. Kolda. 2006. Algorithm 862: MATLAB tensor classes for fast algorithm prototyping. *ACM Transactions on Mathematical Software* 32, 4 (Dec. 2006), 635–653. DOI: https://doi.org/10.1145/1186785.1186794

[4] Brett W. Bader and Tamara G. Kolda. 2017. MATLAB Tensor Toolbox Version 3.0-dev. Retrieved from https://www.tensortoolbox.org.

[5] Daniel Boto-Giralda, Francisco J. Díaz-Pernas, David González-Ortega, José F. Díez-Higuera, Míriam Antón-Rodríguez, Mario Martínez-Zarzuela, and Isabel Torre-Díez. 2010. Wavelet-based denoising for traffic volume time series forecasting with self-organizing neural networks. *Computer-Aided Civil and Infrastructure Engineering* 25, 7 (2010), 530–545.

[6] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. 2011. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Machine Learning* 3, 1 (2011), 1–122.

[7] Jian-Feng Cai, Emmanuel J. Candès, and Zuowei Shen. 2010. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization* 20, 4 (2010), 1956–1982.

[8] Emmanuel J. Candès, Xiaodong Li, Yi Ma, and John Wright. 2011. Robust principal component analysis? *Journal of the ACM* 58, 3 (2011), 11.

[9] Chenyi Chen, Yin Wang, Li Li, Jianming Hu, and Zuo Zhang. 2012. The retrieval of intra-day trend and its influence on traffic prediction. *Transportation Research Part C: Emerging Technologies* 22 (2012), 103–118. DOI: https://doi.org/10.1016/j.trc.2011.12.006

[10] Haibo Chen, Susan Grant-Muller, Lorenzo Mussone, and Frank Montgomery. 2001. A study of hybrid neural network approaches and the effects of missing data on traffic forecasting. *Neural Computing & Applications* 10, 3 (2001), 277–286.

[11] Shuyan Chen, Wei Wang, and Henk van Zuylen. 2010. A comparison of outlier detection algorithms for ITS data. *Expert Systems with Applications* 37, 2 (2010), 1169–1178.

[12] Yudong Chen, Huan Xu, Constantine Caramanis, and Sujay Sanghavi. 2011. Robust matrix completion and corrupted columns. In *Proceedings of the 28th International Conference on Machine Learning (ICML'11)*. 873–880.

[13] Yi-Lei Chen, Chiou-Ting Hsu, and Hong-Yuan Mark Liao. 2013. Simultaneous tensor decomposition and completion using factor priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36, 3 (2013), 577–591.

[14] J. H. Conklin and B. L. Smith. 2002. The use of local lane distribution patterns for the estimation of missing data in transportation management systems. *Transportation Research Record* 1811, 1 (2002), 50–56.

[15] Daniel M. Dunlavy, Tamara G. Kolda, and Evrim Acar. 2011. Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data* 5, 2 (2011), 10.

[16] Cyril Furtlehner, Yufei Han, Jean-Marc Lasgouttes, Victorin Martin, Fabrice Marchal, and Fabien Moutarde. 2010. Spatial and temporal analysis of traffic states on large scale networks. In *Proceedings of the 13th International IEEE Conference on Intelligent Transportation Systems*. IEEE, 1215–1220.

[17] Donald Goldfarb and Zhiwei Qin. 2014. Robust low-rank tensor recovery: Models and algorithms. *SIAM Journal on Matrix Analysis and Applications* 35, 1 (2014), 225–253.

[18] Jianhua Guo, Wei Huang, and Billy M. Williams. 2015. Real time traffic flow outlier detection using short-term traffic conditional variance prediction. *Transportation Research Part C: Emerging Technologies* 50 (2015), 160–172. DOI: https://doi.org/10.1016/j.trc.2014.07.005

[19] Manish Gupta, Jing Gao, Charu C. Aggarwal, and Jiawei Han. 2013. Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and data Engineering* 26, 9 (2013), 2250–2267.

[20] Yufei Han and Fabien Moutarde. 2011. Analysis of network-level traffic states using locality preservative non-negative matrix factorization. In *Proceedings of the 2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC'11)*. IEEE, 501–506.

[21] Johan Håstad. 1990. Tensor rank is NP-complete. *Journal of Algorithms* 11, 4 (1990), 644–654.

[22] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. 2015. *Statistical Learning With Sparsity: The Lasso and Generalizations*. CRC press.

[23] Frank L. Hitchcock. 1927. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics* 6, 1–4 (1927), 164–189.

[24] Victoria Hodge and Jim Austin. 2004. A survey of outlier detection methodologies. *Artificial Intelligence Review* 22, 2 (2004), 85–126.

[25] Tamara G. Kolda and Brett W. Bader. 2009. Tensor decompositions and applications. *SIAM Review* 51, 3 (2009), 455–500.

[26] Xiangjie Kong, Ximeng Song, Feng Xia, Haochen Guo, Jinzhong Wang, and Amr Tolba. 2018. LoTAD: Long-term traffic anomaly detection based on crowdsourced bus trajectory data. *World Wide Web* 21, 3 (2018), 825–847.

[27] Spyridon Kontogiorgis and Robert R. Meyer. 1998. A variable-penalty alternating directions method for convex optimization. *Mathematical Programming* 83, 1–3 (1998), 29–53.
[28] Simon Kwoczek, Sergio Di Martino, and Wolfgang Nejdl. 2014. Predicting and visualizing traffic congestion in the presence of planned special events. *Journal of Visual Languages & Computing* 25, 6 (2014), 973–980.
[29] Rasmus Munk Larse. [2012]. PROPACK - Software for large and sparse SVD calculations. Retrieved from http://sun.stanford.edu/~rmunk/PROPACK/.
[30] Sheng Li, Ming Shao, and Yun Fu. 2018. Multi-view low-rank analysis with applications to outlier detection. *ACM Transactions on Knowledge Discovery from Data* 12, 3 (2018), 32.
[31] Zhouchen Lin, Risheng Liu, and Zhixun Su. 2011. Linearized alternating direction method with adaptive penalty for low-rank representation. In *Proceedings of the 24th International Conference on Neural Information Processing Systems*. 612–620. Retrieved from http://papers.nips.cc/paper/4434-linearized-alternating-direction-method-with-adaptive-penalty-for-low-rank-representation.pdf.
[32] Pierre-Louis Lions and Bertrand Mercier. 1979. Splitting algorithms for the sum of two nonlinear operators. *SIAM Journal on Numerical Analysis* 16, 6 (1979), 964–979.
[33] Siyuan Liu, Lei Chen, and Lionel M. Ni. 2014. Anomaly detection from incomplete data. *ACM Transactions on Knowledge Discovery from Data* 9, 2 (2014), 11.
[34] Wei Liu, Yu Zheng, Sanjay Chawla, Jing Yuan, and Xie Xing. 2011. Discovering spatio-temporal causal interactions in traffic data streams. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1010–1018.
[35] Metro Government of Nashville and Davidson County. 2019. Nashville Fire Department. Retrieved January 10, 2019 from https://www.nashville.gov/Fire-Department.aspx.
[36] Metro Government of Nashville and Davidson County. 2019. Road closures will begin early Saturday for Annual Marathon. Retrieved January 10, 2019 from https://www.nashville.gov/News-Media/News-Article/ID/7468/Road-Closures-will-Begin-Early-Saturday-for-Annual-Marathon.aspx.
[37] Teneessee Department of Transportation. 2019. Middle Tennessee Construction Lane Closures, April 5–11, 2018. Retrieved February 2, 2019 from https://www.tn.gov/tdot/news/2018/4/4/middle-tennessee-construction-lane-closures--april-5-11--2018.html.
[38] Teneessee Department of Transportation. 2019. Weekly construction reports. Retrieved January 10, 2019 from https://www.tn.gov/tdot/news.
[39] Linsey Xiaolin Pang, Sanjay Chawla, Wei Liu, and Yu Zheng. 2013. On detection of emerging anomalous traffic patterns using GPS data. *Data & Knowledge Engineering* 87 (2013), 357–373. DOI: https://doi.org/10.1016/j.datak.2013.05.002
[40] Eun Park, Shawn Turner, and Clifford Spiegelman. 2003. Empirical approaches to outlier detection in intelligent transportation systems data. *Transportation Research Record: Journal of the Transportation Research Board* 1840, 1 (2003), 21–30.
[41] Li Qu, Yi Zhang, Jianming Hu, Liyan Jia, and Li Li. 2008. A BPCA based missing value imputing method for traffic flow volume data. In *Proceedings of the 2008 IEEE Intelligent Vehicles Symposium*. IEEE, 985–990.
[42] Jineng Ren, Xingguo Li, and Jarvis Haupt. 2016. Robust PCA via tensor outlier pursuit. In *Proceedings of the 50th Asilomar Conference on Signals, Systems and Computers*. IEEE, 1744–1749.
[43] Qingquan Song, Hancheng Ge, James Caverlee, and Xia Hu. 2019. Tensor completion algorithms in big data analytics. *ACM Transactions on Knowledge Discovery from Data* 13, 1 (2019), 6.
[44] Fangzhou Sun, Abhishek Dubey, and Jules White. 2017. DxNAT—Deep neural networks for explaining non-recurring traffic congestion. In *Proceedings of the 2017 IEEE International Conference on Big Data (Big Data'17)*. IEEE, 2141–2150.
[45] Huachun Tan, Guangdong Feng, Jianshuai Feng, Wuhong Wang, Yu-Jin Zhang, and Feng Li. 2013. A tensor-based method for missing traffic data completion. *Transportation Research Part C: Emerging Technologies* 28 (2013), 15–27. DOI: https://doi.org/10.1016/j.trc.2012.12.007
[46] Huachun Tan, Jianshuai Feng, Guangdong Feng, Wuhong Wang, and Yu-Jin Zhang. 2013. Traffic volume data outlier recovery via tensor model. *Mathematical Problems in Engineering* 2013 (2013), 1–8. DOI: https://doi.org/10.1155/2013/164810
[47] Gongguo Tang and Arye Nehorai. 2011. Robust principal component analysis based on low-rank and block-sparse matrix decomposition. In *Proceedings of the 2011 45th Annual Conference on Information Sciences and Systems (CISS'11)*. IEEE, 1–5.
[48] Ledyard R. Tucker. 1966. Some mathematical notes on three-mode factor analysis. *Psychometrika* 31, 3 (1966), 279–311.
[49] Rod E. Turochy and Brian Lee Smith. 2000. Applying quality control to traffic condition monitoring. In *Proceedings of 2000 IEEE Intelligent Transportation Systems*. IEEE, 15–20.
[50] Jingyuan Wang, Fei Gao, Peng Cui, Chao Li, and Zhang Xiong. 2014. Discovering urban spatio-temporal structure from time-evolving traffic networks. In *Proceedings of the Asia-Pacific Web Conference*. Springer, 93–104.

[51] Xidao Wen, Yu-Ru Lin, and Konstantinos Pelechrinis. 2018. Event analytics via discriminant tensor factorization. *ACM Transactions on Knowledge Discovery from Data* 12, 6 (2018), 72.

[52] Billy M. Williams and Lester A. Hoel. 2003. Modeling and forecasting vehicular traffic flow as a seasonal ARIMA process: Theoretical basis and empirical results. *Journal of Transportation Engineering* 129, 6 (2003), 664–672.

[53] Yuankai Wu. 2019. Tensor Decomposition Completion and Recovery Papers and Codes. Retrieved from https://github.com/Kaimaoge/Tensor-decomposition-completion-and-recovery-papers-and-codes

[54] Yuankai Wu, Huachun Tan, Yong Li, Feng Li, and Hongwen He. 2017. Robust tensor decomposition based on Cauchy distribution and its applications. *Neurocomputing* 223 (2017), 107–117. DOI: https://doi.org/10.1016/j.neucom.2016.10.030

[55] Yuankai Wu, Huachun Tan, Yong Li, Jian Zhang, and Xiaoxuan Chen. 2018. A fused CP factorization method for incomplete tensors. *IEEE Transactions on Neural Networks and Learning Systems* 30, 3 (2018), 751–764.

[56] Huan Xu, Constantine Caramanis, and Sujay Sanghavi. 2010. Robust PCA via outlier pursuit. In *Proceedings of the Advances in Neural Information Processing Systems*. 2496–2504.

[57] Lin Xu, Yang Yue, and Qingquan Li. 2013. Identifying urban traffic congestion pattern from historical floating car data. *Procedia-Social and Behavioral Sciences* 96 (2013), 2084–2095. DOI: https://doi.org/10.1016/j.sbspro.2013.08.235

[58] Shiming Yang, Konstantinos Kalpakis, and Alain Biem. 2014. Detecting road traffic events by coupling multiple time-series with a nonparametric Bayesian method. *IEEE Transactions on Intelligent Transportation Systems* 15, 5 (2014), 1936–1946.

[59] Yuning Yang, Yunlong Feng, and Johan A.K. Suykens. 2015. Robust low-rank tensor recovery with regularized redescending M-estimator. *IEEE Transactions on Neural Networks and Learning Systems* 27, 9 (2015), 1933–1946.

[60] Xiaoming Yuan and Junfeng Yang. 2009. Sparse and low-rank matrix decomposition via alternating direction methods. *Pacific Journal of Optimization* 12, 1 (2009), 2.

[61] Qibin Zhao, Liqing Zhang, and Andrzej Cichocki. 2015. Bayesian CP factorization of incomplete tensors with automatic rank determination. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37, 9 (2015), 1751–1763.

[62] Pan Zhou and Jiashi Feng. 2017. Outlier-robust tensor PCA. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2263–2271.

Received July 2019; revised July 2020; accepted August 2020