
ROBUST TENSOR RECOVERY WITH FIBER OUTLIERS FOR TRAFFIC EVENTS

A PREPRINT

Yue Hu

Vanderbilt University
Nashville, TN37212

yue.hu@vanderbilt.edu

Daniel Work

Vanderbilt University
Nashville, TN37212

dan.work@vanderbilt.edu

August 28, 2019

ABSTRACT

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 exactly detect outliers even with missing data rates as high as 40%, conditioned on the outlier corruption rate and the Tucker rank of the low rank tensor. Finally, we apply our method on a real traffic dataset corresponding to downtown Nashville, TN, USA and successfully detect the events like severe car crashes, construction lane closures, and other large events that cause significant traffic disruptions.

Keywords robust tensor recovery, tensor factorization, multilinear analysis, outlier detection, traffic events, urban computing

1 Introduction

1.1 Motivation

Event detection is an increasing interest in urban studies [1, 2]. 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 data is accumulating via loop detectors, GPS, high-resolution cameras, etc. The large amount of data provides us insight into the dynamics of urban environment in face 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 event detecting outliers in traffic datasets. Firstly, most large traffic datasets are incomplete [3, 4, 5], 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 sensor failure (e.g., a traffic sensor which loses communication, power, is physically damaged). Missing data can heavily influence the performance of traffic estimation [4, 5, 6, 7], especially as the missing data rate increases. Naive imputation of the missing entries to create a complete dataset is problematic, because without a clear

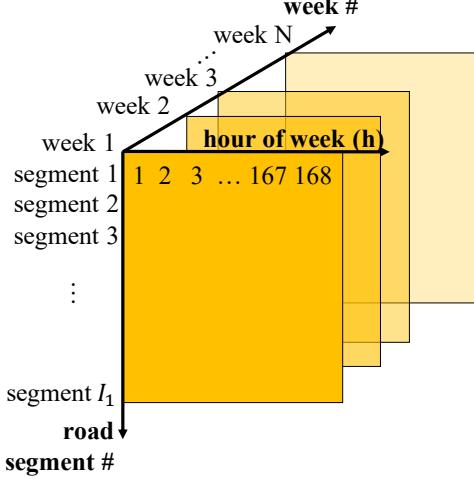


Figure 1: Traffic data is arranged in three-way tensor, with the dimensions corresponding to *i*) the hour of the week, *ii*) the road segment number, and *iii*) the week number.

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 [8, 9, 10, 11]. 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 down stream and upper stream traffic volume.

Most existing researches have 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 [12, 13], or it is assumed that the dataset is clean, for the purpose of missing data imputation [14, 15]. Yet in reality missing data and outliers often exist at the same time. On the other hand, currently most studies on traffic outliers consider only a single monitor spot or a small region [16, 17, 18, 19], not fully exploiting the spatio-temporal correlations. Only a few studies scale up to large regions to capture the urban-wise correlation, including the work of Yang et al. [8] proposing a coupled Bayesian *robust principal component analysis* (robust PCA or RPCA) approach to detect road traffic events, and Liu et al. [20] constructing 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 correlation. Specifically, we model a robust tensor decomposition problem, as illustrated in subsection 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 the dataset, and in either full or partial observations.

1.2 Solution approach

In this subsection, we develop the robust tensor decomposition model for traffic extreme event detection problem, and develop a robust tensor completion problem to take partial observation into account.

To exploit these temporal and spatial structures, a tensor [21, 22] 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-1am all the way to Sunday 11pm-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 [21, 22, 23], which is the generalization of matrix rank to higher dimensions.

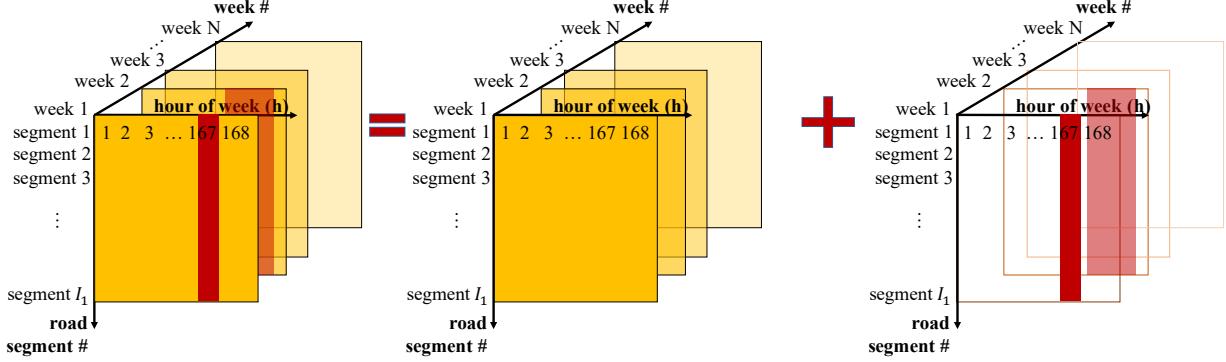


Figure 2: Observation data decomposed into low rank tensor for regular traffic and fiber-sparse tensor for outlier events.

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 unstructured. This fiber-wise sparsity problem is studied in 2D matrix cases [24], 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 patterns 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:

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

The objective function in problem (1) is the weighted cost of tensor rank of \mathcal{X} denoted as $\text{rank}(\mathcal{X})$, the fiber-wise sparsity of \mathcal{E} is denoted as $\text{sparsity}(\mathcal{E})$, 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, and come to the optimization problem:

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

In this paper, we turn problem (1) and (2) into convex programming problems, and solve them by extending from matrix case to tensor case a singular value thresholding algorithm [25, 26] 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 condition of observation and corruption ratio.¹

1.3 Contributions and outline

To summarize, this work has three main contributions:

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

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

The rest of the paper is organized as follows. In Section 3, we provide a review of tensor basics and related robust PCA 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 data set and show its ability to find large scale events.

2 Related work

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

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 [17, 18, 27]. 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 network. The works [20, 28, 29] explore the problem of outlier detection caused by events, while the works [1, 30, 31, 32] 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 utilize the inner structure of the data. Various application have benefited from matrix and tensor based methods, including data completion [9, 33], link prediction [34], network structure clustering [35], etc.

The most relevant works with ours are robust matrix and tensor PCA for outlier detection. l_1 norm regularized robust tensor recovery, as proposed by Goldfarb and Qin [22], is useful when data is polluted with unstructured random noises. Tan et al. [36] also used l_1 norm regularized tensor decomposition for traffic data recovery, in face of random noise corruption. But if outliers are structured, for example grouped in columns, l_1 norm regularization does not yield good results. In addition, although traffic is also modeled in tensor format in [36], only a single road segment is considered, not taking into account network spacial structures.

In face of large events, outliers tend to group in columns or fibers in the dataset, as illustrated in section 1.2. $l_{2,1}$ norm regularized decomposition is suitable for group outlier detection, as shown in [24, 37] for matrices, and [38, 39] for tensors. In addition, Li et al. [40] introduced a multi-view low-rank analysis framework for outlier detection, and Wen et al. [2] used discriminant tensor factorization for event analytics. Our methods differ from the existing tensor outliers pursuit [38, 39] in that they are dealing with slab outliers, i.e., outliers form an entire slice instead of fibers of the tensor. Moreover, compared with existing works, we take one step further and deal with partial observations. As stated in Section 1.1, without an overall understanding of the underlying pattern, we can easily impute the missing entries incorrectly and influence our decision about outliers. We will show in Section 5.3 simulation that our new algorithm can exactly detect the outliers even with 40% missing rate, conditioned on the outlier corruption rate and the rank of the low rank tensor.

3 Preliminaries

In this section, we briefly review the mathematical preliminaries for tensor factorization, adopting the notation of Kolda and Bader [21], and Goldfarb and Qin [22]. We also summarize robust PCA [26], 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, matrix is a second order tensor, and 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 $x_{i_1 i_2 \dots i_N}$ and $y_{i_1 i_2 \dots i_N}$ denote the (i_1, i_2, \dots, i_N) element of \mathcal{X} and \mathcal{Y} respectively. Then tensor inner product can be expressed as

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

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 $\mathbf{Y}_{(n)} := \mathbf{AX}_{(n)}$.

The *Tucker decomposition* [21, 22] 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 $\mathbf{U}^{(n)}$:

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

$\mathcal{G} \in \mathbb{R}^{c_1 \times c_2 \times \cdots \times c_N}$ 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 $(1, 2, \dots, N)$, 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 $n = 1, 2, \dots, N$, i.e., $R_n = \text{rank}_n(\mathcal{X})$, then the set of the N *n-ranks* of \mathcal{X} , (R_1, R_2, \dots, R_N) , is called the *Tucker Rank* [21]. In Tucker decomposition (3), if $c_n = \text{rank}_n(\mathcal{X})$ for all n in $(1, 2, \dots, N)$, 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, and will be harder to be solved.

3.2 Robust PCA

We briefly summarize robust variants of PCA in the matrix setting, which are extended to higher-order tensor settings in Section 4. Robust PCA 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 [26]. For example, if we have 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 fails at finding the subspace for \mathbf{X} given only \mathbf{B} .

To address the problem of gross corruption, Candès et al. [26] proposed a 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 SVD of \mathbf{X} is $\mathbf{X} = \sum_i \sigma_i \mathbf{u}_i \mathbf{v}_i^T$.

The nuclear norm in (4) is proposed as the tightest convex relaxation of matrix rank [25]; and the l_1 norm is the convex approximation for element-wise matrix sparsity. Candès et al. [26] showed that PCP can exactly recover a low rank matrix when it is grossly corrupted at sparse entries. Moreover, by adopting ADMM algorithm, it is possible to solve (4) in polynomial time. The PCP formulation (4) requires incoherence of the column space of the sparse matrix \mathbf{E} [24, 26], 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 (4) is introduced for matrix recovery with column-wise corruption [24, 37]. 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 [41], 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. [24] seeks instead to recover the exact low-dimensional subspace while identifying the location of the corrupted columns. Tang et al. [37] makes 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 [22]. Motivated by the ideas of Candès et al. [26] and Tang [37] 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 $\mathcal{B} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ which is composed of a low-rank tensor $\mathcal{X} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ that is corrupted in a few fibers. In other words, we have $\mathcal{B} = \mathcal{X} + \mathcal{E}$, where $\mathcal{E} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$ 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:

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

Computing the rank of a tensor \mathcal{X} , denoted by $\text{rank}(\mathcal{X})$ is generally an NP-hard problem [22, 42]. 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 [22]. 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

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

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

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

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:

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

Note that related problems to (7) for the matrix setting are addressed in [26, 43]).

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 (6) for the full-observation setting, then for the partial-observation setting (7), adopting an ADMM method [22] 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, $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_N \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$, and rewrite (6) as:

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

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 $\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_N$ 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 [44]. The corresponding augmented Lagrangian function for problem (8) is

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

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 $(\mathcal{X}_1, \mathcal{X}_2, \dots, \mathcal{X}_N), \mathcal{E}, (\mathcal{Y}_1, \mathcal{Y}_2, \dots, \mathcal{Y}_N)$. 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:

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

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

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

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

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

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

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

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:

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

In the second line of (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 [25]): $\mathbf{X}_{i(i)}^{k+1} = \mathbf{T}_{\frac{1}{\mu}}(\frac{1}{\mu} \mathbf{Y}_{i(i)}^k + \mathbf{B}_{(i)} - \mathbf{E}_{(i)}^k)$. 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 $\mathbf{T}_\tau(\mathbf{X})$ in 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:

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

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

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

By the proof of Goldfarb and Qin [22], problem (15) shares the same solution with:

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

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

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

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 [37]:

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

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 $p = I_2 \times I_3 \times \dots \times I_N$ 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 [37].

Algorithm 1 Tensor robust PCA for fiber-wise corruptions

```

1: Given  $\mathcal{B}, \lambda, \mu$ . Initialize  $\mathcal{X}_i = \mathcal{E} = \mathcal{Y}_i = \mathbf{0}$ .
2: for  $k = 0, 1, \dots$  do
3:   for  $i = 1 : N$  do ▷ Update  $\mathcal{X}$ 
4:      $\mathbf{X}_{i(i)}^{k+1} = \mathbf{T}_{\frac{1}{\mu}} \left( \frac{1}{\mu} \mathbf{Y}_{i(i)}^k + \mathbf{B}_{(i)} - \mathbf{E}_{(i)}^k \right)$ .
5:      $\mathcal{X}_i^{k+1} = \text{fold}_i \left( \mathbf{X}_{i(i)}^{k+1} \right)$ 
6:   end for
7:    $\mathcal{C} = \frac{1}{N} \sum_{i=1}^N \left( \frac{1}{\mu} \mathcal{Y}_i^{k+1} + \mathcal{B} - \mathcal{X}_i^{k+1} \right)$  ▷ Update  $\mathcal{E}$ .
8:   for  $j = 1, 2, \dots, p$  do
9:      $\mathbf{E}_{(1)j}^{k+1} = \mathbf{C}_{(1)j} \max \left\{ 0, 1 - \frac{\lambda}{\mu N \|\mathbf{C}_{(1)j}\|_2} \right\}$ 
10:    end for
11:     $\mathcal{E}^{k+1} = \text{fold}_1(\mathbf{E}_{(1)}^{k+1})$ 
12:    for  $i = 1 : N$  do ▷ Update  $\mathcal{Y}$ .
13:       $\mathcal{Y}_i^{k+1} = \mathcal{Y}_i^k + \mu(\mathcal{B} - \mathcal{X}_i^{k+1} - \mathcal{E}^{k+1})$ .
14:    end for
15:  end for
16: return  $\mathcal{X}^k = \frac{1}{N} \sum_{i=1}^N \mathcal{X}_i^k, \mathcal{E}^k$ 

```

Note that compared with the ADMM method where we just update \mathcal{X}_i and \mathcal{E} once, the *augmented Lagrangian multipliers* (ALM) method [45] 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

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

As pointed out by Lin et al. [45], 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 [44]. It is often observed in the matrix settings (see e.g., Lin et al. [45]) 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

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

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 et al. [37], we set the fibers of the low rank tensor to be zero in the locations corresponding to outliers. We introduce a compensation tensor $\mathcal{O} \in \mathbb{R}^{I_1 \times I_2 \times \dots \times I_N}$, 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 (8), we can reformulate problem (7) as:

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

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

Algorithm 2 ADMM for robust tensor completion

```

1: Given  $\mathcal{B}, \lambda, \mu$ . Initialize  $\mathcal{X}_i = \mathcal{E} = \mathcal{Y}_i = \mathcal{O} = \mathbf{0}$ .
2: for  $k = 0, 1, \dots$  do
3:   for  $i = 1:N$  do ▷ Update  $\mathcal{X}$ 
4:      $\mathbf{X}_{i(i)}^{k+1} = \mathbf{T}_{\frac{1}{\mu}}(\mathbf{B}_{(i)} + \frac{1}{\mu}\mathbf{Y}_{i(i)}^k - \mathbf{E}_{(i)}^k - \mathbf{O}_{(i)}^k)$ ,
5:      $\mathcal{X}_i^{k+1} = \text{fold}_i(\mathbf{X}_{i(i)}^{k+1})$ 
6:   end for
7:    $\mathcal{C} = \frac{1}{N} \sum_{i=1}^N \left( \frac{1}{\mu} \mathcal{Y}_i^{k+1} + \mathcal{B} - \mathcal{X}_i^{k+1} - \mathcal{O}^{k+1} \right)$  ▷ Update  $\mathcal{E}$ .
8:   for  $j = 1, 2, \dots, p$  do
9:      $\mathbf{E}_{(1)j}^{k+1} = \mathbf{C}_{(1)j} \max \left\{ 0, 1 - \frac{\lambda}{\mu N \|\mathbf{C}_{(1)j}\|_2} \right\}$ 
10:    end for
11:     $\mathcal{E}^{k+1} = \text{fold}_1(\mathbf{E}_{(1)}^{k+1})$ 
12:     $\mathcal{O}^{k+1} = \left( \sum_{i=1}^N \left( \frac{1}{\mu} \mathcal{Y}_i^k + \mathcal{B} - \mathcal{X}_i^{k+1} - \mathcal{E}^{k+1} \right) \right)_{\Omega^C}$ . ▷ Update  $\mathcal{O}$ .
13:    for  $i = 1 : N$  do ▷ Update  $\mathcal{Y}$ .
14:       $\mathcal{Y}_i^{k+1} = \mathcal{Y}_i^k + \mu(\mathcal{B} - \mathcal{X}_i^{k+1} - \mathcal{E}^{k+1} - \mathcal{O}^{k+1})$ 
15:    end for
16:  end for
17: return  $\mathcal{X}^k = \frac{1}{N} \sum_{i=1}^N \mathcal{X}_i^k, \mathcal{E}^k$ 

```

function for problem (20) is

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

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 (21):

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

Following the same procedure as before (see (14), (15) and (16)), we have:

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

For (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 $(I_1, I_2, \dots, I_N) \in \Omega^C$, and zero otherwise. The procedure is summarized in Algorithm 2.

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 (19) but accounts for the additional tensor \mathcal{O} .

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 is only partially observed and is also fiber-wise corrupted. For both cases, we compare the performance of our algorithms, which are $l_{2,1}$ norm constrained decomposition, with l_1 norm constrained decomposition [22, 36]. We demonstrate via the numerical experiments that our algorithms are able to exactly recover the original low-rank tensor, and identify the position of corrupted fibers. In comparison, l_1 norm constrained algorithms performs poorly in the fiber-wise corrupted settings, unable to achieve exact recoveries.

5.1 Performance measures and implementation

For each of the numerical experiments, the performance of the algorithms are measured by the relative error of the low rank tensor, as well as the precision and recall of the outlier fibers. The *relative error* (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 $\lambda = \frac{1}{0.03I_m}$, where $I_m = \max(I_1, \dots, I_N)$. 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 2.7GHz Intel i7 Processor and 16GB RAM, running Matlab R2018a. We modify and extend the code of Lin et. al [45], using PROPACK toolbox 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 [46] [47] 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 robust PCA

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, $\mathbf{U}^{(i)} \in \mathbb{R}^{I_i \times c_i}$:

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

Table 1: Application of Algorithm 1 on fiber-wise corrupted tensors with full observation, and comparison with l_1 norm constrained decomposition. For different tensor sizes (I_1, I_2, I_3) and Tucker ranks (c_1, c_2, c_3) , where we set $c = 0.1I$, we show the relative errors 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) Algorithm 1: $l_{2,1}$ norm constrained decomposition

(I_1, I_2, I_3)	(c_1, c_2, c_3)	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_3)	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

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 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. [26], the iteration numbers tend to be constant (between 28 and 29 in this case) regardless of tensor size. This indicates that the number 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 in high precision and recall, the relative residual errors are relatively high, in the order of 10^{-1} . This indicates that l_1 norm constrained decomposition can do an adequate job when 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 $\mathbb{R}^{70 \times 70 \times 70}$ 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 relative error of l_1 norm constrained decomposition is constantly higher. After the corruption ratio exceeds 0.2, the estimation is no longer useful, with the relative error exceeding 100%.

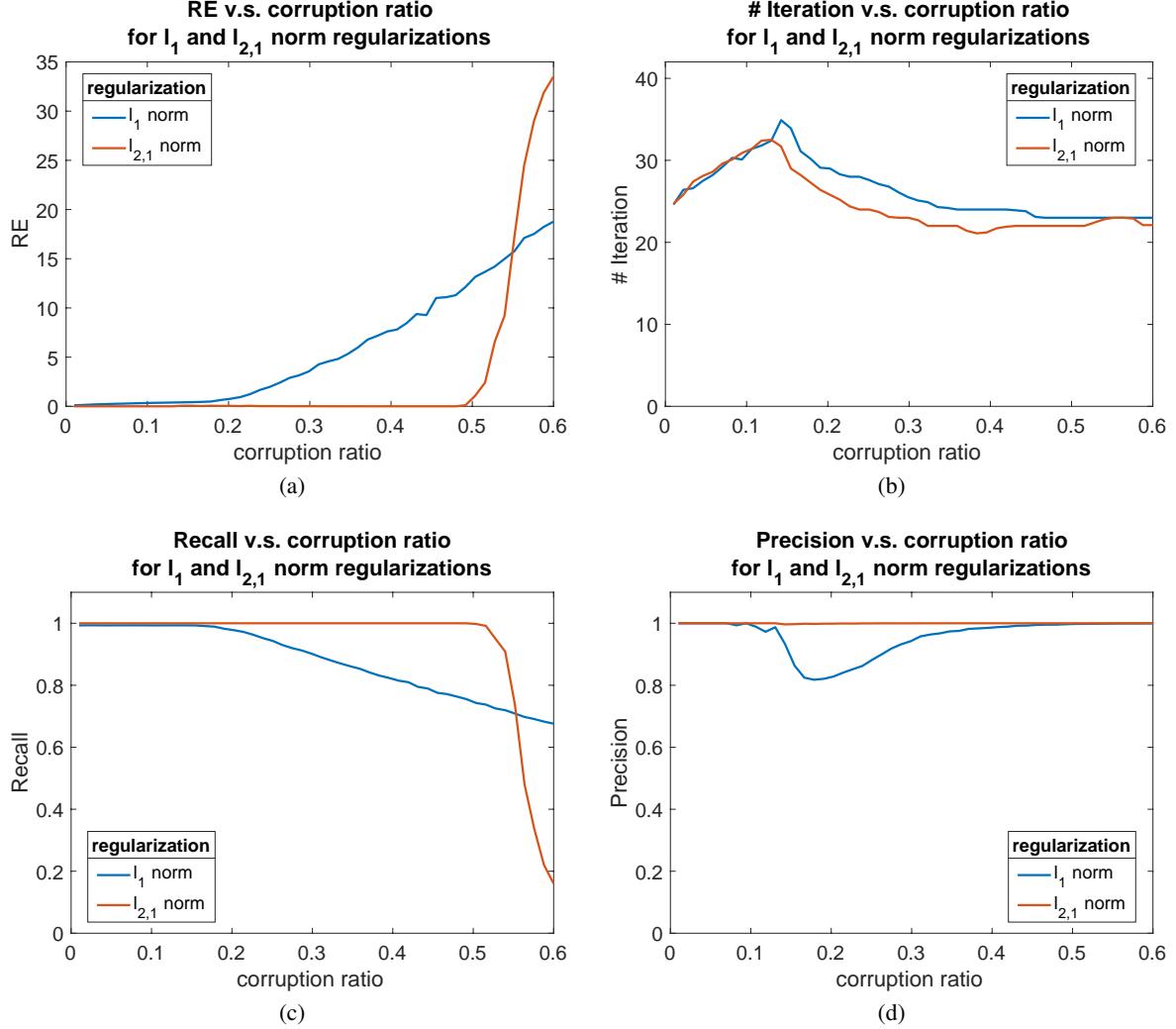


Figure 3: Comparison of Algorithm 1 ($l_{2,1}$ norm regularized tensor decomposition), with l_1 norm regularized tensor decomposition, as gross corruption rate changes. We fix the tensor size at $\mathbb{R}^{70 \times 70 \times 70}$, and fix the tucker rank of low rank tensor \mathcal{X}_0 at $(5, 5, 5)$, 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 is only partially observed, and compare it with l_1 norm constrained decomposition.

5.3.1 Simulation conditions

We first generate a 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.

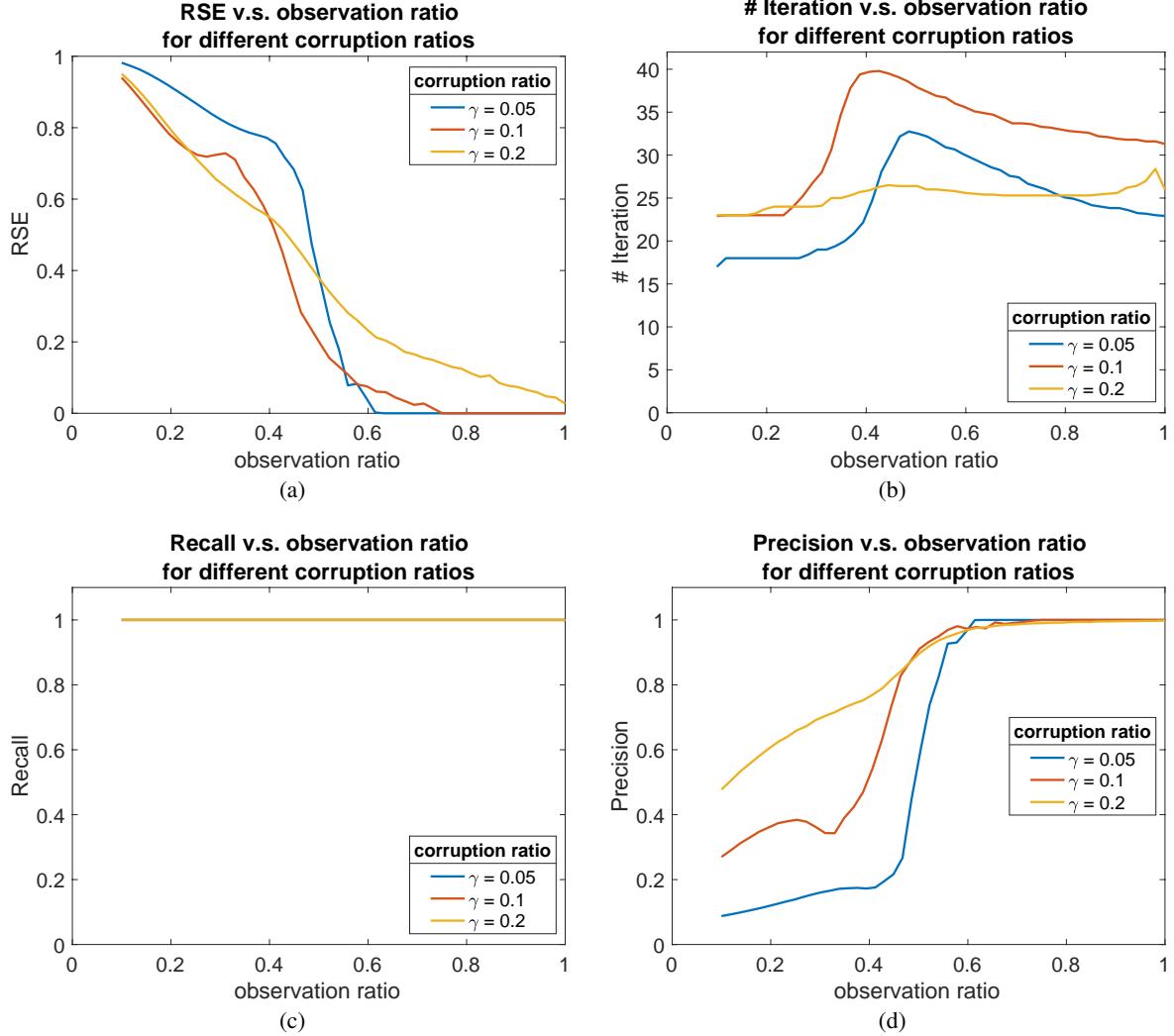


Figure 4: Results of Algorithm 1 as a function of observation ratio with different corruption rates. The low-rank tensor \mathcal{X}_0 size is fixed at $\mathbb{R}^{70 \times 70 \times 70}$ with a tucker rank of $(5, 5, 5)$, and the gross corruption ratio γ is set at 0.05, 0.1, or 0.2. The result is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

5.3.2 Influence of the corruption and observation ratios

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 $\mathbb{R}^{70 \times 70 \times 70}$ 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 relative error 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 is missing.

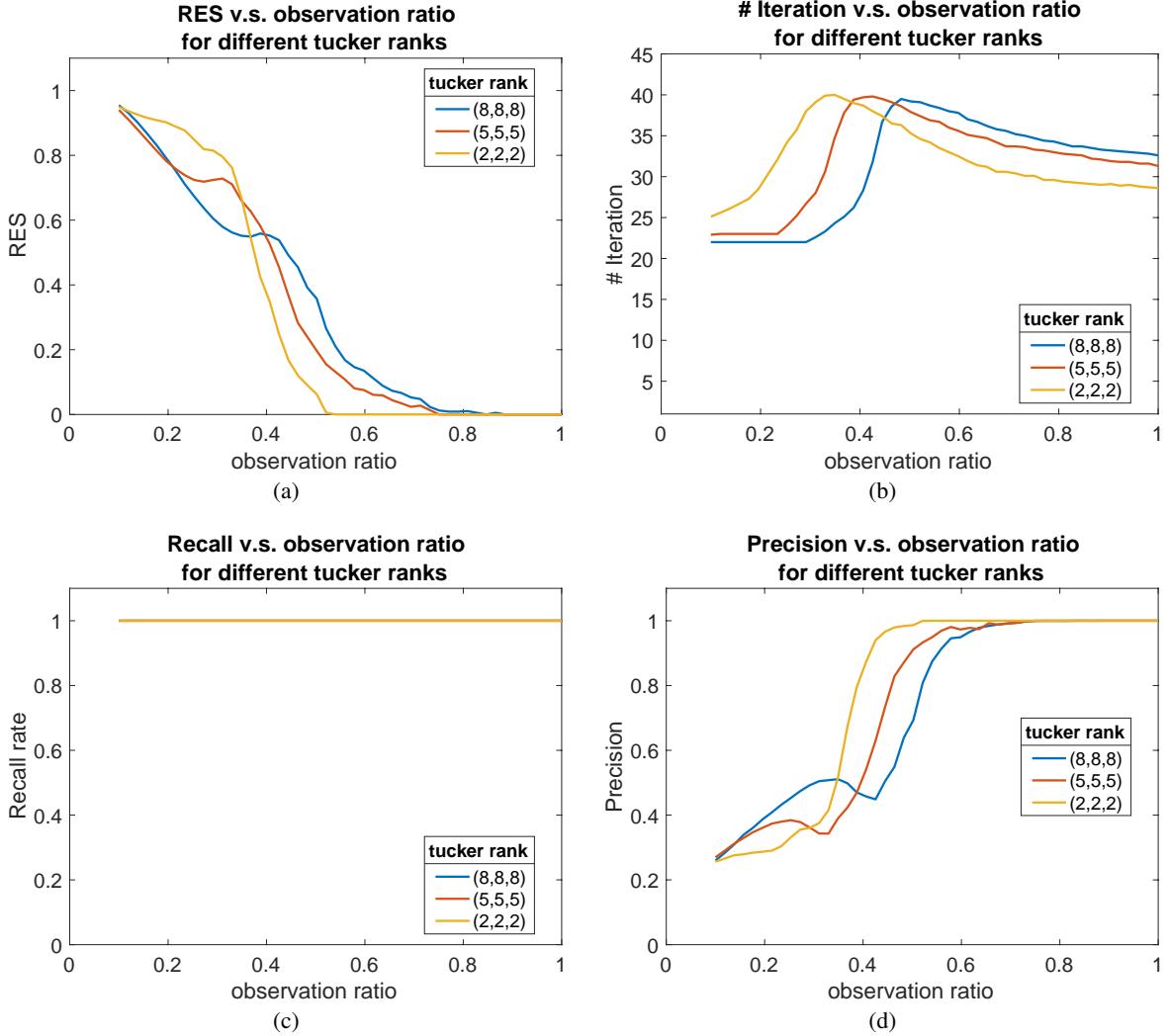


Figure 5: Results of Algorithm 2 as a function of the observation ratio considering \mathcal{X}_0 with varying tucker rank. The low-rank tensor \mathcal{X}_0 is generated with a tucker rank of (2,2,2), (5,5,5), and (8,8,8) respectively, with a fixed size of $\mathbb{R}^{70 \times 70 \times 70}$, and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

5.3.3 Influence of the observation ratio and the tensor rank

Next we fix the low-rank tensor \mathcal{X}_0 at size $\mathbb{R}^{70 \times 70 \times 70}$ and gross corruption ratio $\gamma = 0.1$. For \mathcal{X}_0 of different tucker ranks (2,2,2), (5,5,5), 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-transition behavior, that when observation ratio is above a critical threshold, the decomposition is exact, with precision and recall at one, and relative error 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.

5.3.4 Comparison with $l_{2,1}$ regularized tensor decomposition

Next, we compare the performance of Algorithm 2 ($l_{2,1}$ norm regularized tensor completion), with l_1 norm regularized tensor completion. We fix the low-rank tensor \mathcal{X}_0 at size $\mathbb{R}^{70 \times 70 \times 70}$ with a tucker rank of (5,5,5), and fix gross corruption ratio $\gamma = 0.1$. We vary the observation ratio ρ from 0.1 to 1 and run Algorithm 2 and l_1 norm regularized tensor completion. We run 10 times and average the results.

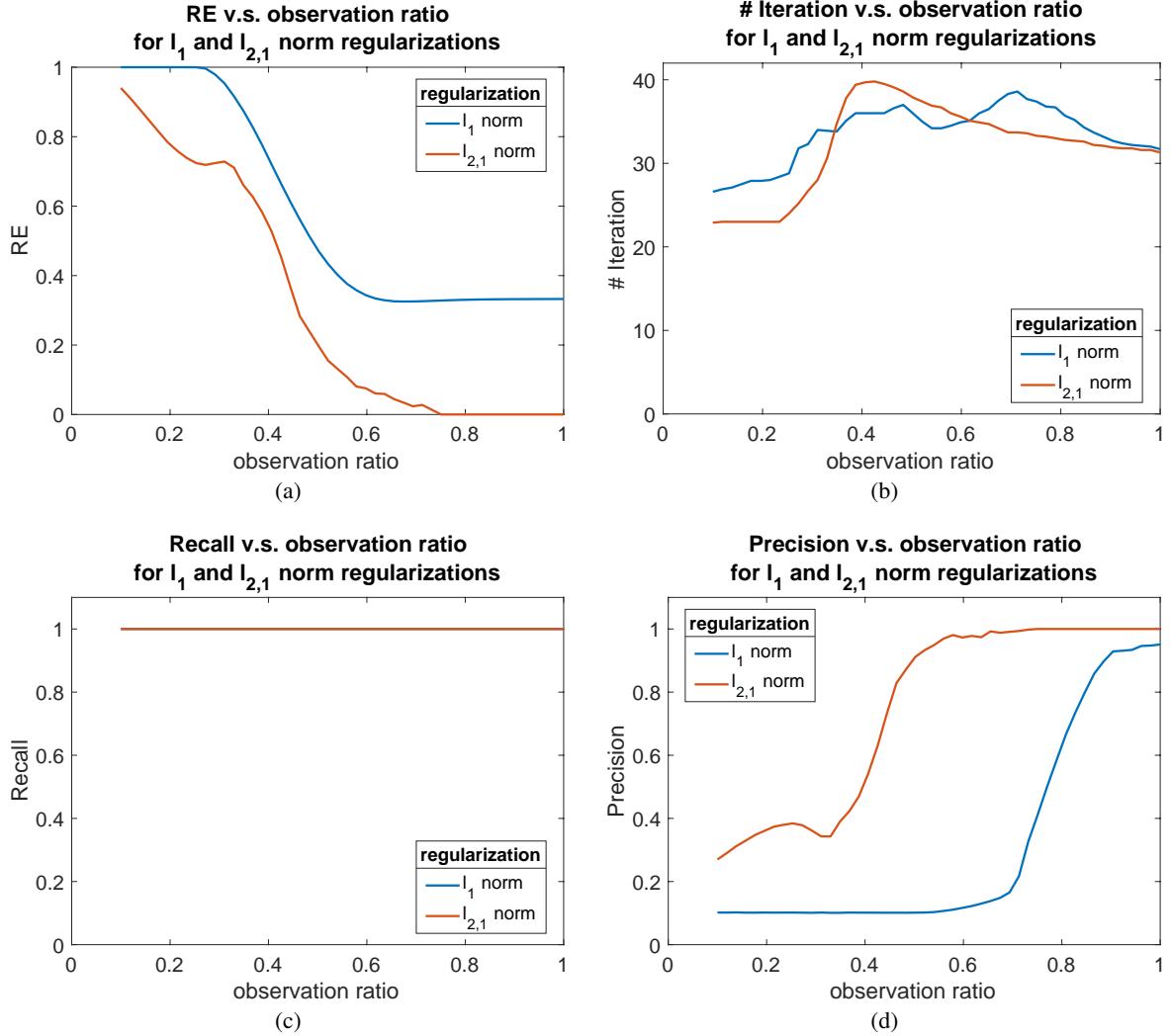


Figure 6: Comparison of Algorithm 2 ($l_{2,1}$ norm regularized tensor completion), with l_1 norm regularized tensor completion, as observation ratio varies. The low-rank tensor \mathcal{X}_0 size is fixed at $\mathbb{R}^{70 \times 70 \times 70}$ with a tucker rank of $(5, 5, 5)$, and the gross corruption ratio γ is set at 0.1. The result is an average over 10 trials.

Figure 6 shows the result. We can see that the performance under l_1 norm regularization is constantly worse, with a higher relative residual error and a lower precision. With a corruption ratio of 0.1, even when data is fully observed, l_1 norm regularized tensor completion can only achieve a relative error of around 0.3, and a precision of around 0.9. Moreover, l_1 norm regularization has a smaller range of observation ratio, outside of which the performance drops sharply, namely $\rho > 0.9$, compared with approximately $\rho > 0.7$ for $l_{2,1}$ norm regularization.

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 $\mathbb{R}^{70 \times 70 \times 70}$. 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 7 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 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.

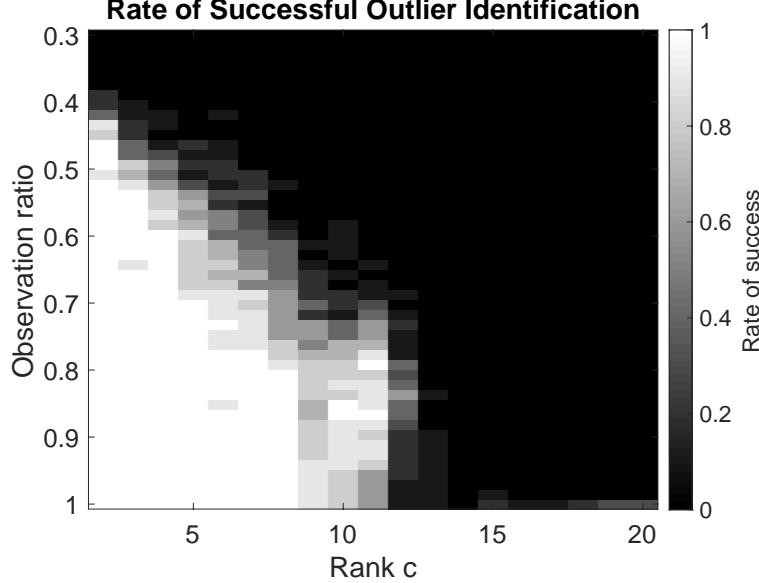


Figure 7: Rate of successful outlier identification across 10 trials. The color denotes the rate of success. For each trial we create a tensor with size $\mathbb{R}^{70 \times 70 \times 70}$ and tucker rank (c,c,c) (x-axis), fiber-wise corrupt it at ratio $\gamma = 0.1$. We vary the observation ratio ρ 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 Jan 1 to Apr 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 one-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 2856 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 is 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 (2856 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 [48]. This process is done by manually comparing the events identified by Algorithm 2 with the accident records of the Nashville fire department [49], and lane closure records of the Tennessee Department of Transportation (TDOT) [50], which is the state transportation authority. Most incidents clear out after one hour, while some last for two hours or more. For the rest 5 hours detected as outliers, the average speed of the roads appear faster than normal, and we are not able to identify obvious causes for the abnormally light traffic conditions.

Figures 10 and 11 visualize some outlier events as examples. The right columns of the heat maps show the average speed of the road in one hour, and the left columns show 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.

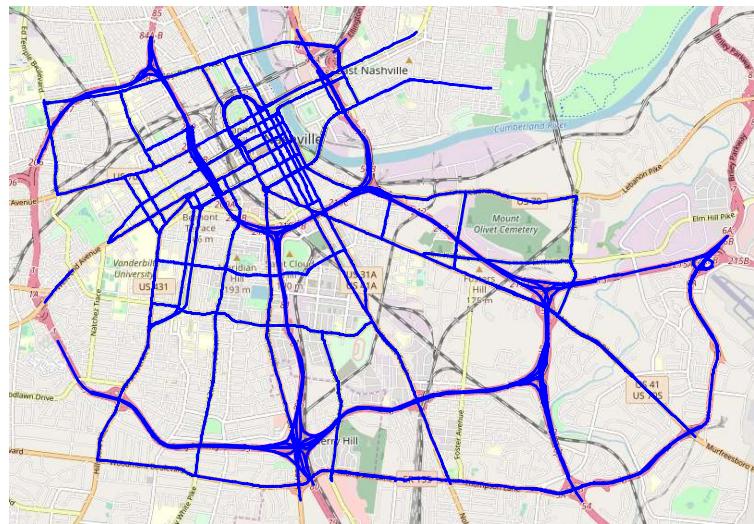


Figure 8: Map of downtown Nashville. The studied road segments are marked in blue and consist of the major freeways and surface streets.

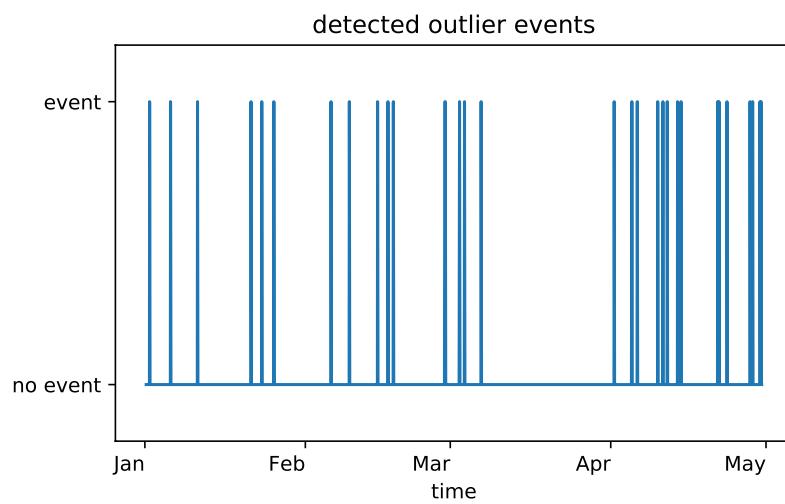


Figure 9: Stem plot of detected outlier events. Each column indicates the time when an detected outlier event takes place.

Figure 10 shows an event that corresponds to a series of car crashes. On 12:00 noon March 2, 2019 several severe car crashes happened on the major freeway and arterial going around downtown Nashville, namely Interstate 40 (labeled 1 in the second row of Figure 10), Charlotte Avenue (labeled 2 in the second row), and Carroll street (labeled 3 in the second row). At about 13:00, two other car crashes happened in different segments of Interstate 40 (labeled 4 and 5 in the third row of Figure 10) [49]. The sequence of severe crashes created unusual congestion for that time of day and took two hours to clear out. The Algorithm marks the the hours 12:00 and 13:00 as outliers.

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 [51]. The first two 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.

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 is 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, demonstrates 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.

8 Acknowledgement

This material is based upon work supported by the National Science Foundation under Grant No. CMMI-1727785.

References

- [1] Siyuan Liu, Lei Chen, and Lionel M Ni. Anomaly detection from incomplete data. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 9(2):11, 2014.
- [2] Xidao Wen, Yu-Ru Lin, and Konstantinos Pelechrinis. Event analytics via discriminant tensor factorization. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 12(6):72, 2018.
- [3] JH Conklin and BL Smith. The use of local lane distribution patterns for the estimation of missing data in transportation management systems. *Transportation Research Record*, 1811:50–56, 2002.
- [4] Chenyi Chen, Yin Wang, Li Li, Jianming Hu, and Zuo Zhang. The retrieval of intra-day trend and its influence on traffic prediction. *Transportation Research Part C: Emerging Technologies*, 22:103–118, 2012.
- [5] Huachun Tan, Guangdong Feng, Jianshuai Feng, Wuhong Wang, Yu-Jin Zhang, and Feng Li. A tensor-based method for missing traffic data completion. *Transportation Research Part C: Emerging Technologies*, 28:15–27, 2013.
- [6] Haibo Chen, Susan Grant-Muller, Lorenzo Mussone, and Frank Montgomery. A study of hybrid neural network approaches and the effects of missing data on traffic forecasting. *Neural Computing & Applications*, 10(3):277–286, 2001.
- [7] Billy M Williams and Lester A Hoel. Modeling and forecasting vehicular traffic flow as a seasonal arima process: Theoretical basis and empirical results. *Journal of Transportation Engineering*, 129(6):664–672, 2003.
- [8] Shiming Yang, Konstantinos Kalpakis, and Alain Biem. Detecting road traffic events by coupling multiple timeseries with a nonparametric Bayesian method. *IEEE Transactions on Intelligent Transportation Systems*, 15(5):1936–1946, 2014.
- [9] Muhammad Tayyab Asif, Nikola Mitrovic, Justin Dauwels, and Patrick Jaillet. Matrix and tensor based methods for missing data estimation in large traffic networks. *IEEE Transactions on intelligent transportation systems*, 17(7):1816–1825, 2016.

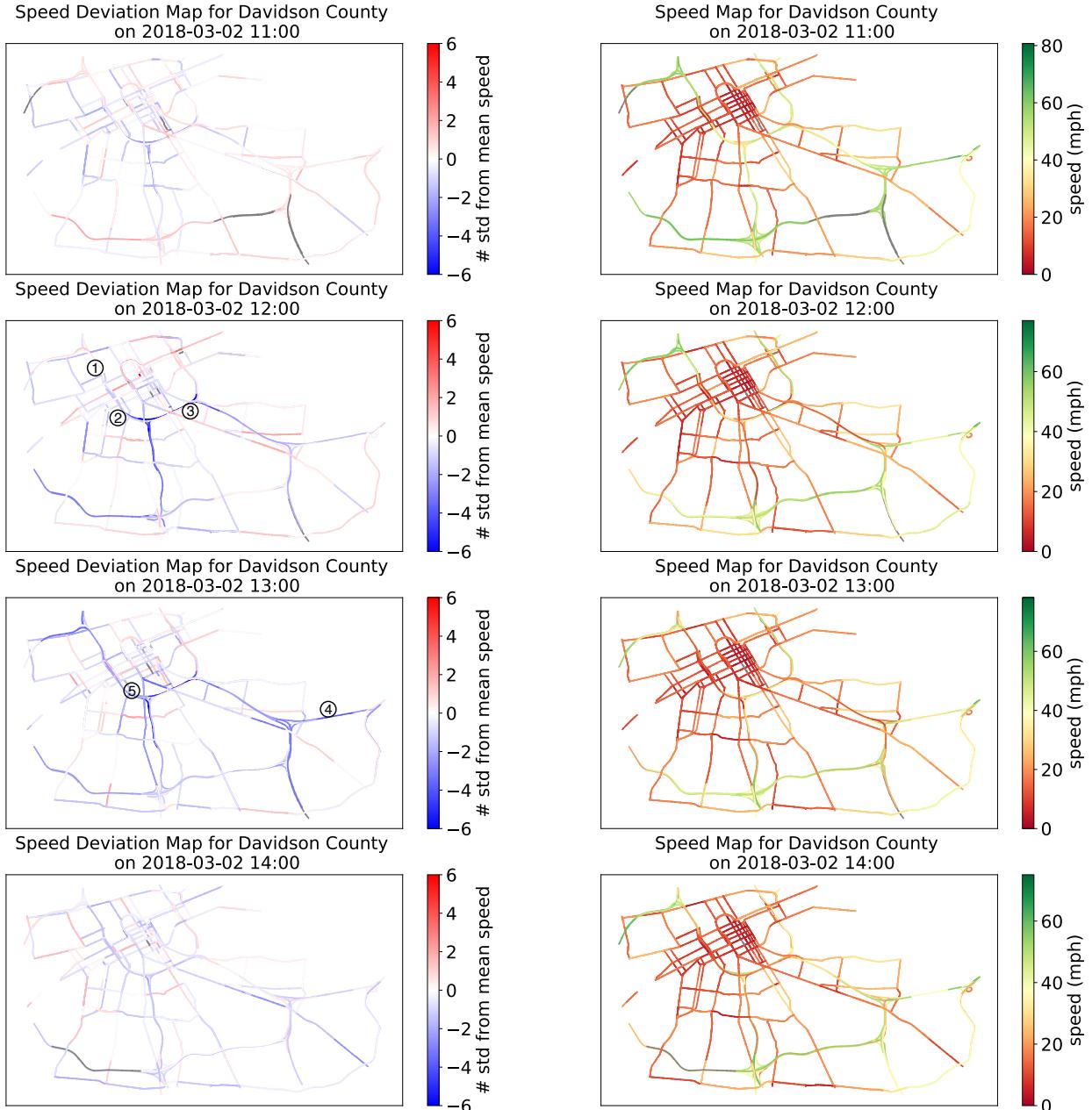


Figure 10: Detected car crashes. The second and third row, i.e., 12:00 and 13:00 are marked as outliers. At 12:00, several severe car crashes happened on Interstate 40 (area 1 in the second row), Charlotte Avenue (area 2 in the second row), and Carroll street (area 3 in the second row). At about 13:00,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

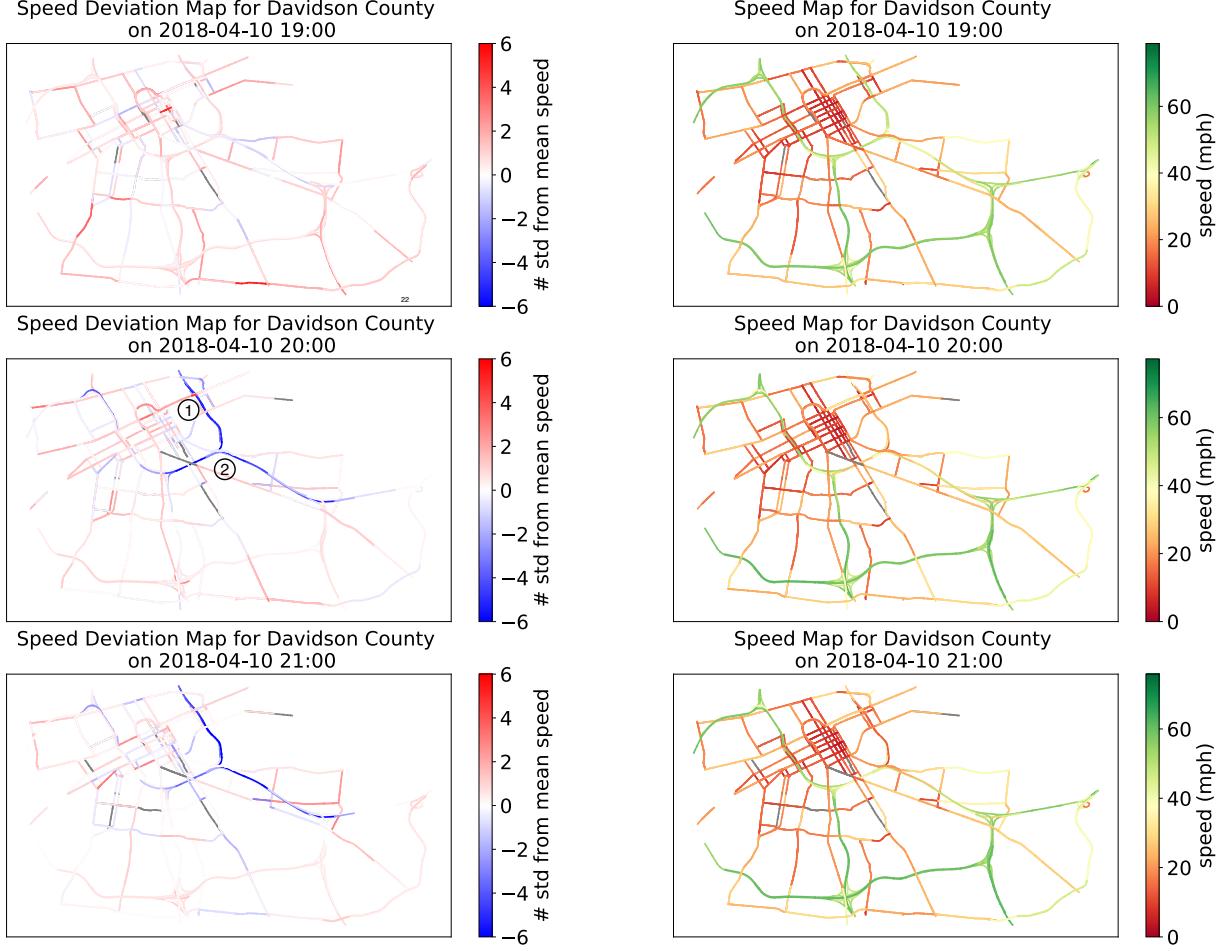


Figure 11: Detected road closure. The second and third row, i.e. 20:00 and 21:00 are marked as outliers. It is the time when segments of Interstate 24 (the north-south route marked as 1), Interstate 40 (the east-west route marked as 2), as well as streets connecting to Interstate 24, were closed for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

- [10] Cyril Furtlechner, Yufei Han, Jean-Marc Lasgouttes, Victorin Martin, Fabrice Marchal, and Fabien Moutarde. Spatial and temporal analysis of traffic states on large scale networks. In *13th International IEEE Conference on Intelligent Transportation Systems*, pages 1215–1220. IEEE, 2010.
- [11] Yufei Han and Fabien Moutarde. Analysis of network-level traffic states using locality preservative non-negative matrix factorization. In *2011 14th international IEEE conference on Intelligent Transportation Systems (ITSC)*, pages 501–506. IEEE, 2011.
- [12] Victoria Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial intelligence review*, 22(2):85–126, 2004.
- [13] Manish Gupta, Jing Gao, Charu C Aggarwal, and Jiawei Han. Outlier detection for temporal data: A survey. *IEEE Transactions on Knowledge and data Engineering*, 26(9):2250–2267, 2013.
- [14] Li Qu, Yi Zhang, Jianming Hu, Liyan Jia, and Li Li. A bPCA based missing value imputing method for traffic flow volume data. In *2008 IEEE Intelligent Vehicles Symposium*, pages 985–990. IEEE, 2008.
- [15] Abdelmonem A. Afifi and Robert M. Elashoff. Missing observations in multivariate statistics i. review of the literature. *Journal of the American Statistical Association*, 61(315):595–604, 1966.
- [16] Jianhua Guo, Wei Huang, and Billy M Williams. Real time traffic flow outlier detection using short-term traffic conditional variance prediction. *Transportation Research Part C: Emerging Technologies*, 50:160–172, 2015.
- [17] Shuyan Chen, Wei Wang, and Henk van Zuylen. A comparison of outlier detection algorithms for its data. *Expert Systems with Applications*, 37(2):1169–1178, 2010.

- [18] Eun Park, Shawn Turner, and Clifford Spiegelman. Empirical approaches to outlier detection in intelligent transportation systems data. *Transportation Research Record: Journal of the Transportation Research Board*, (1840):21–30, 2003.
- [19] Rod E Turochy and Brian Lee Smith. Applying quality control to traffic condition monitoring. In *Proceedings of 2000 IEEE Intelligent Transportation Systems. (Cat. No. 00TH8493)*, pages 15–20. IEEE, 2000.
- [20] Wei Liu, Yu Zheng, Sanjay Chawla, Jing Yuan, and Xie Xing. Discovering spatio-temporal causal interactions in traffic data streams. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1010–1018. ACM, 2011.
- [21] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM Review*, 51(3):455–500, 2009.
- [22] Donald Goldfarb and Zhiwei Qin. Robust low-rank tensor recovery: Models and algorithms. *SIAM Journal on Matrix Analysis and Applications*, 35(1):225–253, 2014.
- [23] Ledyard R Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1966.
- [24] Huan Xu, Constantine Caramanis, and Sujay Sanghavi. Robust pca via outlier pursuit. In *Advances in Neural Information Processing Systems*, pages 2496–2504, 2010.
- [25] Jian-Feng Cai, Emmanuel J Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20(4):1956–1982, 2010.
- [26] Emmanuel J Candès, Xiaodong Li, Yi Ma, and John Wright. Robust principal component analysis? *Journal of the ACM (JACM)*, 58(3):11, 2011.
- [27] Daniel Boto-Giralda, Francisco J Díaz-Pernas, David González-Ortega, José F Díez-Higuera, Míriam Antón-Rodríguez, Mario Martínez-Zarzuela, and Isabel Torre-Díez. Wavelet-based denoising for traffic volume time series forecasting with self-organizing neural networks. *Computer-Aided Civil and Infrastructure Engineering*, 25(7):530–545, 2010.
- [28] Xiangjie Kong, Ximeng Song, Feng Xia, Haochen Guo, Jinzhong Wang, and Amr Tolba. Lotad: long-term traffic anomaly detection based on crowdsourced bus trajectory data. *World Wide Web*, 21(3):825–847, 2018.
- [29] Linsey Xiaolin Pang, Sanjay Chawla, Wei Liu, and Yu Zheng. On detection of emerging anomalous traffic patterns using GPS data. *Data & Knowledge Engineering*, 87:357–373, 2013.
- [30] Fangzhou Sun, Abhishek Dubey, and Jules White. Dxnat—deep neural networks for explaining non-recurring traffic congestion. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 2141–2150. IEEE, 2017.
- [31] Lin Xu, Yang Yue, and Qingquan Li. Identifying urban traffic congestion pattern from historical floating car data. *Procedia-Social and Behavioral Sciences*, 96:2084–2095, 2013.
- [32] Simon Kwoczek, Sergio Di Martino, and Wolfgang Nejdl. Predicting and visualizing traffic congestion in the presence of planned special events. *Journal of Visual Languages & Computing*, 25(6):973–980, 2014.
- [33] Qingquan Song, Hancheng Ge, James Caverlee, and Xia Hu. Tensor completion algorithms in big data analytics. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 13(1):6, 2019.
- [34] Daniel M Dunlavy, Tamara G Kolda, and Evrim Acar. Temporal link prediction using matrix and tensor factorizations. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(2):10, 2011.
- [35] Jingyuan Wang, Fei Gao, Peng Cui, Chao Li, and Zhang Xiong. Discovering urban spatio-temporal structure from time-evolving traffic networks. In *Asia-Pacific Web Conference*, pages 93–104. Springer, 2014.
- [36] Huachun Tan, Jianshuai Feng, Guangdong Feng, Wuhong Wang, and Yu-Jin Zhang. Traffic volume data outlier recovery via tensor model. *Mathematical Problems in Engineering*, 2013, 2013.
- [37] Gongguo Tang and Arye Nehorai. 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)*, pages 1–5. IEEE, 2011.
- [38] Pan Zhou and Jiashi Feng. Outlier-robust tensor pca. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2263–2271, 2017.
- [39] Jineng Ren, Xingguo Li, and Jarvis Haupt. Robust pca via tensor outlier pursuit. In *Proceedings of the 50th Asilomar Conference on Signals, Systems and Computers*, pages 1744–1749. IEEE, 2016.
- [40] Sheng Li, Ming Shao, and Yun Fu. Multi-view low-rank analysis with applications to outlier detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 12(3):32, 2018.

- [41] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical Learning With Sparsity: the Lasso and Generalizations*. CRC press, 2015.
- [42] Johan Håstad. Tensor rank is NP-complete. *Journal of Algorithms*, 11(4):644–654, 1990.
- [43] Yudong Chen, Huan Xu, Constantine Caramanis, and Sujay Sanghavi. Robust matrix completion and corrupted columns. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 873–880, 2011.
- [44] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, Jonathan Eckstein, et al. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine learning*, 3(1):1–122, 2011.
- [45] Zhouchen Lin, Risheng Liu, and Zhixun Su. Linearized alternating direction method with adaptive penalty for low-rank representation. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 612–620. Curran Associates, Inc., 2011.
- [46] Brett W. Bader, Tamara G. Kolda, et al. Matlab tensor toolbox version 3.0-dev. Available online, October 2017.
- [47] Brett W. Bader and Tamara G. Kolda. Algorithm 862: MATLAB tensor classes for fast algorithm prototyping. *ACM Transactions on Mathematical Software*, 32(4):635–653, December 2006.
- [48] Metro Government of Nashville and Davidson County. Road closures will begin early saturday for annual marathon. <https://www.nashville.gov/News-Media/News-Article/ID/7468/Road-Closures-will-Begin-Early-Saturday-for-Annual-Marathon.aspx>, 2019. Accessed: 2019-01-10.
- [49] Metro Government of Nashville and Davidson County. Nashville fire department. <https://www.nashville.gov/Fire-Department.aspx>, 2019. Accessed: 2019-01-10.
- [50] Tennessee Department of Transportation. Weekly construction reports. <https://www.tn.gov/tdot/news>, 2019. Accessed: 2019-01-10.
- [51] Tennessee Department of Transportation. Middle tennessee construction lane closures, april 5-11, 2018. <https://www.tn.gov/tdot/news/2018/4/4/middle-tennessee-construction-lane-closures--april-5-11--2018.html>, 2019. Accessed: 2019-02-10.