CHAPTER

14

Outlier Discrimination and Correction in Intelligent Transportation Systems

Junqin Huang, Linghe Kong, and Guihai Chen
Shanghai Jiao Tong University, Shanghai, China

1 INTRODUCTION

The rapid growth of vehicles brings the problem of traffic congestion. Shanghai, New York, and Paris, for example, which are the largest cities in the world, suffer from constant traffic congestion. It is essential for drivers to check the traffic conditions before going out via navigation applications such as AMAP, Google Maps, Baidu Map, and so forth.

Traditionally, these navigation applications have provided traffic conditions by collecting location data from users or in-vehicle sensing devices in real time. Due to its openness, malicious users could upload fake data to the data center intentionally, which would make navigation applications provide misleading traffic information. As a result, the transportation system falls into chaos. On the other hand, hardware heterogeneity and failure [1] may cause bias or missing information in collected location data.

In Fig. 1, a snapshot of the road network in downtown Chengdu, which we research in this chapter, is shown. We collect traffic condition data of these roads from AMAP [2] every 5 min for 24 h. Fig. 2 shows the missing ratio of traffic condition data. By statistics, 42% of the total data are missing. The large missing ratio and potential faulty values increase the misleading traffic information. It is significant to discriminate the outliers to guarantee an efficient transportation system, which motivates us to propose a novel method to address it.

Lots of work has been contributed to this field, such as Yoon et al. [3] utilize traffic patterns on a specific road segment to estimate traffic conditions, which needs large datasets in identifying traffic conditions, and each road, which is analyzed independently, is not suitable for the real-time environment (e.g., navigation applications). Li et al. [4] propose a compressive sensing-based method to estimate missing values in urban traffic sensing; however, this method does not take faulty data into consideration, which may lead to reconstruction deviation. Cheng et al. [5] propose a DECO framework to improve data quality for crowd-sensing applications. They believe users who have a good reputation will not upload faulty data, which is impractical, too. In summary, although some algorithms and methods are proposed to address such problems, few of them are practical in real life.

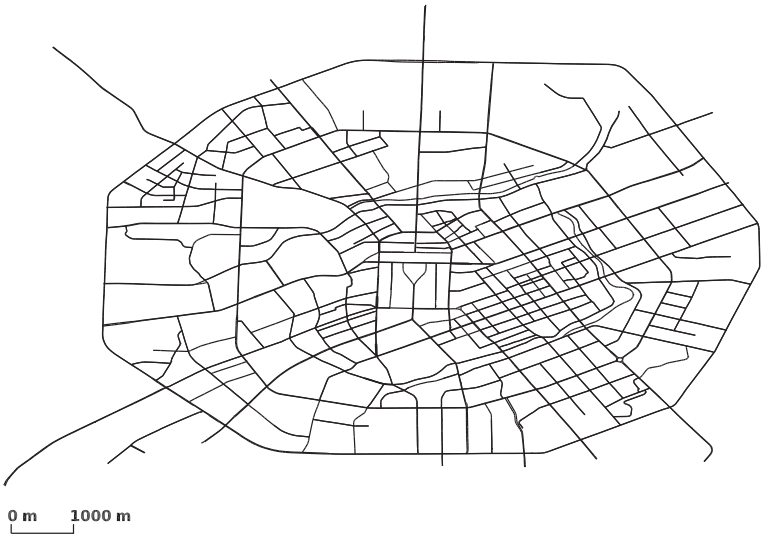
In this chapter, we propose an outlier discrimination and correction (ODC) method to discriminate faulty data and reconstruct missing data in traffic condition datasets. Through simulations conducted on traffic condition datasets from AMAP, we observe that ODC can produce only about a 6 km/h reconstruction error, even with a missing data ratio $\beta = 50\%$ and faulty data ratio $\alpha = 30\%$.

Our contributions can be summarized as follows:

- We propose the ODC method, which is applied to outlier detection and missing data reconstruction in the field of traffic information. It only uses single dimension information, that is, the average speed for analysis, so it takes very little time and is suitable for real time applications.
- We mine temporal stability and spatial correlation features from traffic condition datasets, and utilize these hidden structures to improve performance of compressive sensing.
- We evaluate ODC based on real urban traffic condition datasets, demonstrating that it produces low reconstruction errors, even when the missing ratio is 50% and the faulty ratio is 30% in traffic condition datasets.

https://doi.org/10.1016/B978-0-12-815032-0.00014-7

FIG. 1 Road network of Chengdu downtown region.



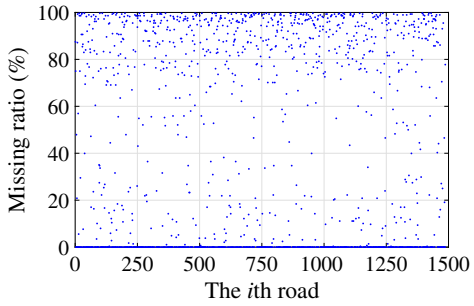


FIG. 2 Missing ratio of traffic condition data from AMAP.

The rest of this chapter is organized as follows. In Section 2, we formulate the problem. In Section 3, we analyze the hidden structures in traffic condition datasets. The compressive sensing-based method, ODC, is proposed in Section 4. In Section 5, we evaluate the ODC method through real urban traffic traces. We introduce related work in Section 6 and conclude this chapter in Section 7.

2 PROBLEM FORMULATION

The better traffic conditions are, the higher the driving speed allowed, so we adopt the mean velocity of roads as the metric for quantifying traffic conditions in this chapter. As shown in Fig. 1, the road network consists of n roads (different directions of the same road are considered different roads). The sensing period includes t time slots. Each road updates its mean velocity once per time slot by calculating real-time changes of location data from users. $x(i,j)$ denotes the mean velocity of road i at time slot j, where $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, t$.

Definition 1. Traffic Condition Matrix (TCM) describes the mean velocity of road i in time slot j. TCM is defined by $X = (x(i,j))_{n\times t}$.

Each line of X represents a time series of mean velocity of one road. And every data point is valid, that is, no missing or faulty data points.

Definition 2. Direct Sensory Matrix (DSM) is an $n \times t$ matrix, which contains mean velocity calculated through raw data, where we may find missing or faulty data points. DSM is denoted by S, defined as follows:

$$S(i,j) = (s(i,j))_{n\times t} = \begin{cases} 0 & \text{if } x(i,j) \text{ is missing} \\ x(i,j) + \xi & \text{otherwise} \end{cases} \tag{1}$$

Data error is denoted by ξ, $s(i,j)$ is considered a faulty data point if $|\xi| > \eta$, where η is a predefined threshold, otherwise as a normal one.

Definition 3. Faulty Matrix (FM) is an $n \times t$ matrix, denoted by F, set $F(i,j)$ to 1 if $s(i,j)$ is faulty, defined as:

$$F(i,j) = \begin{cases} 1 & \text{if } s(i,j) \text{ is faulty} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

Definition 4. Existence Matrix (EM) is denoted by E, represents if $x(i,j)$ is collected in S, defined as:

$$E(i,j) = (\varepsilon(i,j))_{n\times t} = \begin{cases} 0 & \text{if } x(i,j) \text{ is not in } S \\ 1 & \text{otherwise} \end{cases} \tag{3}$$

Definition 5. Faulty Data Detection Matrix (FDM) is an $n \times t$ matrix, denoted by D, marks out the faulty data detected through the ODC method, defined similar to F.

Definition 6. Reconstruct Matrix (RM) is generated by interpolating the missing or faulty values in DSM to approximate to TCM. RM is defined by $\hat{X} = (\hat{x}(i,j))_{n\times t}$.

Definition 7. Binary Index Matrix (BIM) combines E and D, marks out trusted data, that is, not missing or faulty data. BIM is denoted by B, defined as:

$$B = (b(i,j))_{n\times t} = E \cap \overline{D} \tag{4}$$

The first problem is to detect faulty data in S, formulated as:

Problem 1 (Faulty Data Detection in Direct Sensory Matrix (FDSM)).

Given S and E, the FDSM problem is to find D, the expectation of D is as close to F as possible. That is,

$$\begin{aligned} &\text{Objective}: \min \|D - F\|_F \\ &\text{Subject to}: S, E \end{aligned} \tag{5}$$

where $\|\cdot\|_F$ is the Frobenius norm used to measure the difference between D and F. That is, $\|D-F\|_F = \sqrt{\sum_{i,j}(D(i,j)-F(i,j))^2}$.

The second problem is to reconstruct the real traffic condition (TCM) based on the sensory data (DSM).

Problem 2 (Traffic Condition Reconstruction in Road Networks (TCRN)).

Given S and B, the TCRN problem is to find the optimal $\hat{X}$ that approximates X as closely as possible, that is,

$$\begin{aligned} &\text{Objective}: \min \left\|X - \hat{X}\right\|_F \\ &\text{Subject to}: S, B \end{aligned} \tag{6}$$

3 TRAFFIC CONDITION DATA MINING

We gather the real-time traffic condition data from AMAP [2], which is available on the Internet. The dataset collects the mean velocity of roads in the Chengdu downtown region, covering a period of 24 h on November 25, 2017. It contains 1487 roads and 288 time slots, with slot intervals of 5 min. However, we cannot utilize the raw data from AMAP because of the existence of missing and faulty data, which will lead to the lack of ground truth. To generate TCM, we perform preprocessing and select complete subsets from raw data. The selected subset contains 617 roads $\times$ 288 slots.

3.1 Discovery Over Low Rank Structure

The mean velocity of different roads over different times are not independent, there are structures. We reveal the inherent structure with the singular value decomposition (SVD), which is a kind of factorization of a matrix. SVD is usually used for creating a low rank matrix approximation. The matrix $(x(i,j))_{n\times t}$ can be decomposed as:

$$X = U\Sigma V^T = \sum_{i=1}^{\min(n,t)} \sigma_i u_i v_i^T \tag{7}$$

where U is an $n \times n$ unitary matrix (i.e., $UU^T = U^TU = I$), V^T is the transpose of an $t \times t$ unitary matrix, Σ is an $n \times t$ diagonal matrix with the singular value σ_i of X on the main diagonal, where $\sigma_i \geq \sigma_{i+1}$, $i = 1,2,\ldots,\min(n,t)$.

The matrix is low rank if $r \ll \min(n,t)$, r is equal to nonzero singular values. σ_i represents the ith largest singular value of X, which means X can be approximately denoted by top r singular values.

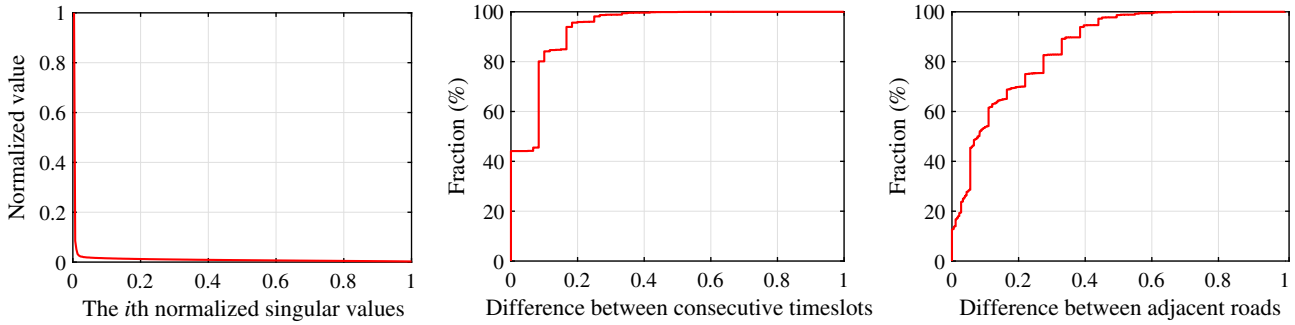


FIG. 3 Traffic condition hidden structures mining from the complete subset. (A) Low rank feature; (B) temporal stability feature; (C) spatial correlation feature.

$$\sum_{i=1}^{r}\sigma_i \approx \sum_{i=1}^{\min(n,t)}\sigma_i \tag{8}$$

In Fig. 3A, we illustrate the distribution of singular values in TCM. The X-axis denotes the ith largest singular value, and the Y-axis presents the values of ith singular value, and both of them are normalized. This figure shows that the top 50% singular values contribute the most energy in traffic condition data. This means that TCM has low rank structures, which is the prerequisite for using compressive sensing.

3.2 Temporal Stability Feature

The mean velocity of a road changes stably between adjacent timeslots in general when the interval is not too large. On the basis of this observation, we analyze the traffic condition data in the time dimension to reveal temporal features. We measure the temporal stability of road i at timeslot j by calculating the difference between adjacent timeslots as

$$\triangle x(i,j) = |x(i,j) - x(i,j-1)| \tag{9}$$

The CDF of $\triangle x(i,j)$ is plotted in Fig. 3B. The X-axis denotes difference between adjacent timeslots (normalized), and the Y-axis presents the cumulative probability of $\triangle x(i,j)$. We can observe that above 40% $\triangle x(i,j)$ in TCM are 0, and above 95% $\triangle x(i,j)$ are less than 0.2. This indicates that temporal stability exists in TCM. On the basis of this observation, we can improve the compressive sensing by adding a temporal feature dimension.

3.3 Spatial Correlation Feature

From the road network in Fig. 1, we can observe that there are many adjacent roads, and the mean velocity between adjacent roads is usually similar. Hence, we can consider the difference from the space dimension.

In traffic condition datasets from AMAP, roads are characterized through polylines, which consist of several latitude and longitude coordinates. Thus, we can utilize these latitude and longitude coordinates to find out if two roads are adjacent. To illustrate it in mathematical form, the adjacent roads matrix (ARM) is defined as follows:

$$H(a,b) = \begin{cases} 1 & \text{if } a \text{ and } b \text{ are adjacent} \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

where $a = 1, 2, \ldots, n$ and $b = 1, 2, \ldots, n$. ARM is an $n \times n$ matrix, denoted by H, which presents the spatial correlation of any two roads.

We measure the spatial correlation of road i at timeslot j by calculating the difference in mean velocity between road i and the average of its all adjacent roads. $\theta x(i,j)$ is calculated as:

$$\theta x(i,j) = \left| x(i,j) - \frac{H(i,:)X(:,j)}{\sum H(i,:)} \right| \tag{11}$$

where $H(i,:)$ is the ith row of H, $X(:,j)$ is the jth column of X. $H(i,:)X(:,j)$ presents the sum values of all adjacent roads of road i at timeslot j. $\sum H(i,:)$ presents the number of adjacent roads of road i.

The CDF of $\theta x(i,j)$ is plotted in Fig. 3C. The X-axis depicts the normalized difference in mean velocity between road i and the average of its all adjacent roads. The Y-axis presents the cumulative possibility. This figure shows that above 80% $\theta x(i,j)$ are smaller than 0.3 and almost all of $\theta x(i,j)$ are smaller than 0.5. This observation indicates that TCM also has a spatial correlation feature, which means we can improve compressive sensing from a spatial dimension.

4 ODC METHOD BASED ON COMPRESSIVE SENSING

Based on the preceding observations, we propose and detail the ODC method in this section.

4.1 Overview

The ODC method is proposed to discriminate faulty data and correct them in traffic condition datasets. Thus the ODC method consists of two main procedures, which are discrimination and correction. The ODC method is based on compressive sensing technology, which is effective for reconstruct missing data in structured or redundancy datasets. However, the result of reconstruction is not so effective when there is massive faulty data in dataset, that is, faulty data can cause a large deviation during the reconstruction procedure. Therefore, we need to discover faulty data before correcting to mitigate reconstruction deviation.

Fig. 4 shows the program flow chart of the ODC method. We take Direct Sensory Matrix S, Existence Matrix E, ARM H, λ_1, λ_2, λ_3, rank bound r, and iteration times *maxIter* as inputs and Reconstruct Matrix $\hat{X}$, and Faulty Data Detection Matrix D as outputs. D is set to all ones at the beginning of the procedure because the ODC method aims to find more faulty data. There are three main functions in the ODC method, which are *FaultyDataDetection*(), *ASD*(), and *updateFDM*(). The task of *FaultyDataDetection*() is to discriminate faulty data by utilizing its temporal-spatial correlation, as we discuss in Sections 3.2 and 3.3, and mark faulty data points in D. Then ODC combines E and D, and marks out trusted data points in B. B is taken as an input of *ASD*(), which is to reconstruct missing data in the dataset (faulty or missing data are both taken as missing data in *ASD*()). The output of *ASD*() is Reconstruct Matrix $\hat{X}$, *updateFDM*() compares $\hat{X}$ and S to update the Faulty Data Detection Matrix D. The ODC method repeats this process until D is not

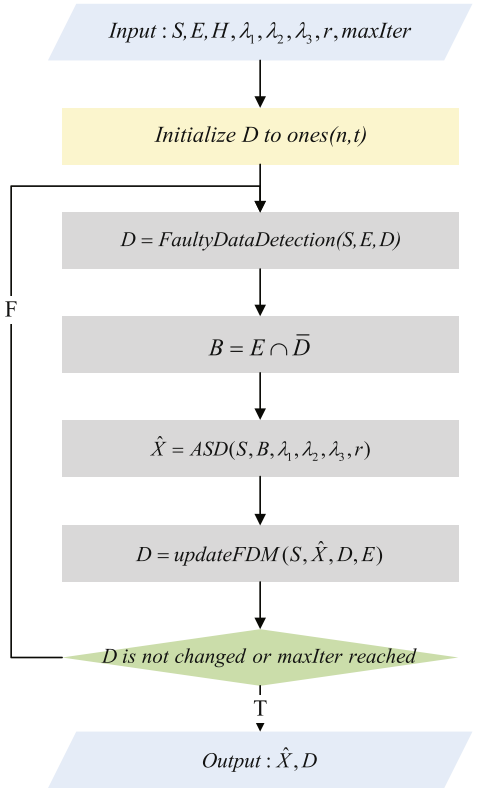


FIG. 4 ODC method flow chart.

changed after updating or iteration times have reached *maxIter*. We will detail these three functions, *FaultyDataDetection*(), *ASD*(), and *updateFDM*(), in the rest of this section, that is, the Optimization of Compressive Sensing and Faulty Data Detection Approach Based on the Temporal Stability Feature.

4.2 Optimization of Compressive Sensing

Compressive sensing is known as an effective technique to reconstruct missing data for sparse matrices. The main idea of compressive sensing is that datasets in the real world often contain structures or redundancy, which we have revealed in Section 3.1 for traffic condition datasets. Thus, we utilize compressive sensing to compute an estimate of traffic condition matrix $\hat{X}$ that approximates the real traffic condition matrix X.

Because of the low rank feature of X, we can approximately represent X by its r largest singular values. According to Eqs. (7), (8), we can approximately represent X as

$$\hat{X} = \sum_{i=1}^{r} \sigma_i u_i v_i^T \tag{12}$$

Actually, $\hat{X}$ is the best r-rank approximation that minimizes the Frobenius norm between X and $\hat{X}$.

However, it is impossible to directly find the $\hat{X}$, because we do not know X and the proper rank. Thus, we can alternatively solve the following rank minimization problem:

$$\begin{aligned} &\text{Objective}: \min\left(rank\left(\hat{X}\right)\right) \\ &\text{Subject to}: \hat{X} \circ B = S \end{aligned} \tag{13}$$

where $\circ$ refers to the Hadamard product ($\hat{X} \circ B = S$ means $\hat{X}(i,j)B(i,j) = S(i,j)$).

However, it is difficult to solve this minimization problem because it is nonconvex. To address this difficulty, we make use of the SVD-like factorization of $\hat{X}$:

$$\hat{X} = U\Sigma V = LR^T \tag{14}$$

where $L = U\Sigma^{1/2}$ and $R = V\,\Sigma^{1/2}$. According to the compressive sensing theory, if the restricted isometry property [6] holds, minimizing the nuclear form [7–9] can perform rank minimization exactly for a low rank matrix, that is, we just find matrix L and R that minimize the summation of their Frobenius norms as:

$$\begin{aligned} &\text{Objective}: \min\left(\|L\|_F^2 + \|R\|_F^2\right) \\ &\text{Subject to}: \left(LR^T\right) \circ B = S \end{aligned} \tag{15}$$

In practice, L and R that strictly satisfy the constraint are likely to fail for two reasons. First, there are noises in the sensory data that may lead to the over-fit problem. Second, TCM can be a low rank matrix, although it may not exactly be low ranking. Thus, we use the Lagrange multiplier to relax the constraint:

$$\min\left(\left\|\left(LR^T\right) \circ B - S\right\|_F^2 + \lambda_1\left(\|L\|_F^2 + \|R\|_F^2\right)\right) \tag{16}$$

where λ_1 controls the trade-off between rank minimization and accuracy fitness.

Although the performance of compressive sensing relies on low rank structure, it can also be improved by taking temporal stability and spatial correlation into consideration.

4.2.1 *Temporal Stability Improvement*

The temporal constraint matrix $\mathbb{T}$ is defined as:

$$\mathbb{T} = \begin{bmatrix} 1 & -1 & 0 & \cdots & 0 \\ 0 & 1 & -1 & \ddots & \vdots \\ 0 & 0 & 1 & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & -1 \\ -1 & 0 & 0 & \cdots & 1 \end{bmatrix}_{t \times t} \tag{17}$$

which describes the difference between two consecutive timeslots.

$LR^T\mathbb{T}$ captures temporal stability of the traffic condition dataset, which we have revealed in Section 3.2. Hence, we can introduce the temporal constraint $\left\|LR^T\mathbb{T}\right\|_F^2$ into Eq. (16) to filter more noises and errors.

4.2.2 Spatial Correlation Improvement

The spatial constraint matrix is denoted by $\mathbb{H}$, which outlines the difference between adjacent roads at the same timeslot; its definition is similar to $\mathbb{H}$ in [10].

$\mathbb{H}$ indeed is a transformation of the ARM H: (1) The central diagonal of $\mathbb{H}$ is given by $\mathbb{H}(i,i) = (-\sum H(i,:))$, where $H(i,:)$ means the ith row in H. Other elements in $\mathbb{H}$ are the same as H. (2) Normalize rows of $\mathbb{H}$ and we get the final $\mathbb{H}$ transformed from H. For example, if there is an ARM H:

$$H = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \tag{18}$$

after the preceding steps of transformation, the spatial constraint matrix is:

$$\mathbb{H} = \begin{bmatrix} 1 & 0 & -1 \\ \frac{1}{2} & 1 & \frac{1}{2} \\ -1 & 0 & 1 \end{bmatrix} \tag{19}$$

It is obvious that $\mathbb{H}LR^T$ represents the difference between road i and the average value of its adjacent roads. Similar to $LR^T\mathbb{T}$, $\mathbb{H}LR^T$ captures spatial correlation of the traffic condition dataset, which has been mined in Section 3.3. We can also introduce the spatial constraint $\left\|\mathbb{H}LR^T\right\|_F^2$ into Eq. (16) to help make a more accurate estimation of LR^T.

Hence, after exploiting the temporal-spatial stability features in the traffic condition dataset, we optimize the compressive sensing approach by developing Eq. (16) as:

$$\min\left(\left\|\left(LR^T\right)\circ B - S\right\|_F^2 + \lambda_1\left(\|L\|_F^2 + \|R\|_F^2\right) + \lambda_2\left\|\mathbb{H}LR^T\right\|_F^2 + \lambda_3\left\|LR^T\mathbb{T}\right\|_F^2\right) \tag{20}$$

where λ_2 and λ_3 are used as the scaling of $\left\|\mathbb{H}LR^T\right\|_F^2$ and $\left\|LR^T\mathbb{T}\right\|_F^2$.

4.2.3 Optimized Compressive Sensing Algorithm

The target of the optimized compressive sensing algorithm is to find L and R that minimize Eq. (20). We define the objective function as

$$f(L,R) = \left\|\left(LR^T\right)\circ B - S\right\|_F^2 + \lambda_1\left(\|L\|_F^2 + \|R\|_F^2\right) + \lambda_2\left\|\mathbb{H}LR^T\right\|_F^2 + \lambda_3\left\|LR^T\mathbb{T}\right\|_F^2 \tag{21}$$

It is obvious that $f(L,R)$ is nonconvex. However, if we fix L or R, the function on the other variables is convex. Thus, we utilize the alternating steepest descent (ASD) algorithm [11, 12] to do the minimization, which is commonly used in the low rank matrix completion. The pseudo code of ASD is shown in Algorithm 1, Σ_r means the first r columns of Σ.

The main idea of ASD is to apply the steepest gradient descent to $f(L,R)$ with respect to L and R. First, L and R are randomly initialized. Then we fix L and update R by using a single step of simple line search along the gradient descent direction. And next, we fix R and update L using a similar approach. This process is repeated until convergence (we consider it is convergent when the change of the function value is less than a threshold).

To detail the line search along the gradient descent directions, we denote

$$f(L,R) = f_1(L,R) + f_2(L,R) + f_3(L,R) + f_4(L,R) \tag{22}$$

where

$$\begin{aligned} f_1(L,R) &= \left\|\left(LR^T\right)\circ B - S\right\|_F^2 \\ f_2(L,R) &= \lambda_1\left(\|L\|_F^2 + \|R\|_F^2\right) \\ f_3(L,R) &= \lambda_2\left\|\mathbb{H}LR^T\right\|_F^2 \\ f_4(L,R) &= \lambda_3\left\|LR^T\mathbb{T}\right\|_F^2 \end{aligned}$$

The gradient descent directions are

$$\begin{aligned} \nabla_l &= \nabla_l^1 + \nabla_l^2 + \nabla_l^3 + \nabla_l^4 \\ \nabla_r &= \nabla_r^1 + \nabla_r^2 + \nabla_r^3 + \nabla_r^4 \end{aligned} \tag{23}$$

ALGORITHM 1

ALTERNATING STEEPEST DESCENT: *ASD*()

Require: Direct Sensory Matrix S, Binary Index Matrix B, $\mathbb{H}$, $\mathbb{T}$, λ_1, λ_2, λ_3, rank bound r
Ensure: Reconstruct Matrix $\hat{X}$
$[n,t] \leftarrow size(S)$;
$S' \leftarrow S$;
for $S'(i,j)$ in S' **do**
if $B(i,j) = 0$ **then**
$S'(i,j) \leftarrow$ average of two nearest values;
end if
end for
$[U,\Sigma,V] \leftarrow SVD(S')$;
$L \leftarrow U \cdot \Sigma_r^{1/2}$;
$R \leftarrow V \cdot \Sigma_r^{1/2}$;
repeat
$\beta_1 \leftarrow f(L,R)$;
Calculate ∇_l and ∇_r according to Eq. (25);
$\alpha_l \leftarrow \arg\ \min_t f(L - \alpha\nabla_l, R)$;
$\alpha_r \leftarrow \arg\ \min_t f(L, R - \alpha\nabla_r)$;
$L \leftarrow L - \alpha_l \nabla_l$;
$R \leftarrow R - \alpha_r \nabla_r$;
until $\frac{\beta_2 - \beta_1}{\beta_1} < threshold$;
$\hat{X} \leftarrow L \cdot R^T$;
return $\hat{X}$;

where

$$\begin{aligned}
\nabla_l^1 &= 2\left(\left(LR^T\right) \circ B - S\right)R \\
\nabla_l^2 &= 2\lambda_1 L \\
\nabla_l^3 &= 2\lambda_2 \mathbb{H}^T \mathbb{H} L R^T R \\
\nabla_l^4 &= 2\lambda_3 L R^T \mathbb{T}\mathbb{T}^T R \\
\nabla_r^1 &= 2\left(\left(RL^T\right) \circ B^T - S^T\right)L \\
\nabla_r^2 &= 2\lambda_1 R \\
\nabla_r^3 &= 2\lambda_2 R L^T \mathbb{H}^T \mathbb{H} L \\
\nabla_r^4 &= 2\lambda_3 \mathbb{T}\mathbb{T}^T R L^T L
\end{aligned}$$

The steepest descent along L and R are selected to minimize the updated value of $f(L,R)$ along the direction ∇_l and ∇_r, respectively. We denote α_l and α_r as the steepest descent stepsize along the gradient descent directions. Thus, the values of L and R are updated as:

$$\begin{aligned}
L &= L - \alpha_l \nabla_l \\
R &= R - \alpha_r \nabla_r
\end{aligned} \tag{24}$$

α_l and α_r are selected to minimize the value of $f(L,R)$ along the direction ∇_l and ∇_r, that is,

$$\begin{aligned}
\alpha_l &= \arg\ \min_t\ f(L - \alpha\nabla_l, R) \\
\alpha_r &= \arg\ \min_t\ f(L, R - \alpha\nabla_r)
\end{aligned} \tag{25}$$

We differentiate the f and set it to zero, then we can solve α_l and α_r.

However, considering $f(L,R)$ is nonconvex, ASD may converge to a local optimal point. To mitigate this, we initialize the value of L and R by: (1) Let $S' = S$ and the missing data points in S' are replaced with the average value of two nearest existing values. (2) Use SVD to decompose the S' and compute the L and R. In this way, we can get the optimized L and R with closer starting points to the optimal one, which mitigates the local optimal problem.

4.3 Faulty Data Detection Approach Based on the Temporal Stability Feature

The Faulty Data Detection Approach is based on the temporal stability of the traffic condition dataset. In Fig. 3B, we can observe that the traffic condition dataset has good temporal stability features (difference between consecutive timeslots of >95% data are <20%). Thus, we can utilize this feature to discriminate faulty data before reconstruction and make D closer to F at the start point.

ALGORITHM 2

FAULTYDATADETECTION()

Require: Direct Sensory Matrix S, Existence Matrix E, Faulty Data Detection Matrix D
Ensure: Faulty Data Detection Matrix D
$[n,t] \leftarrow size(S)$;
$T = S\mathbb{T}$;
for $i \leftarrow 1$ **to** n **do**
for $j \leftarrow 1$ **to** t **do**
if $E(i,j) = 0$ **or** $D(i,j) = 0$ **then**
continue ;
end if
$el \leftarrow new\ Array()$;
for $k \leftarrow j - \lfloor window/2 \rfloor$ **to** $j + \lfloor window/2 \rfloor$ **do**
if $k \neq j$ **and** $E(i,j) \neq 0$ **and** $D(i,j) \neq 0$ **then**
$el \leftarrow [el, T(i,k)]$;
end if
end for
$ratio = \max(el) - \min(el)$;
$m = mean(el)$;
if $|T(i,k) - m| < ratio/2$ **then**
$D(i,j) \leftarrow 0$;
end if
end for
end for
return D;

The difference between consecutive timeslots is formulated in Eq. (9). Thus we can introduce matrix T to store the difference between the consecutive timeslots in each road, which is calculated as $T = X\mathbb{T}$, where $\mathbb{T}$ is defined in Eq. (17), $T(i, j)$ represents the difference between $X(i, j-1)$ and $X(i, j)$.

The pseudo code of the Faulty Data Detection Approach is shown in Algorithm 2. The main idea of the faulty data detection method is to calculate the average value of the difference between consecutive timeslots in ith road, and compare it with $T(i, j)$, if the difference between them is less than the dynamic threshold, $X(i, j)$ is considered normal data. The threshold is determined by the largest variation range in each road. But considering that change gradients differ at different times of a day, we further refine the 288 timeslots into several segments with *window* width so that we can make the threshold more precisely. So we calculate the *ratio* as Algorithm 2 in each loop, and compare $T(i, j)$ with the average value of points in the range of the window to judge whether $X(i, j)$ is normal data or not. D is initialized as all ones for the first time its executed, this may cause some normal data to be misjudged. Nevertheless, the aim of D is to mark out faulty data as much as possible, and this also ensures the convergence of the ODC method.

Other than detecting faulty data before reconstruction, we need to update D according to estimated $\hat{X}$ after that. We need a $n \times t$ matrix to record the difference between X and $\hat{X}$, which is defined as:

$$\delta_{n \times t} = \left| X - \hat{X} \right| \tag{26}$$

If $\delta(i, j)$ is less than lower threshold $thres_l$ and $D(i, j)$ is 1, we set it to 0. If $\delta(i, j)$ is larger than the upper threshold $thres_u$ and $D(i, j)$ is 0, we set it to 1. $thres_l$ and $thres_u$ depend on the tolerance of data deviation. No operation will be done if $X(i, j)$ is a missing data point (i.e., $E(i, j)$ is 0). The pseudo code of *updateFDM*() is shown in Algorithm 3.

ALGORITHM 3

UPDATEFDM()

Require: Direct Sensory Matrix S, Existence Matrix E, Faulty Data Detection Matrix D, Reconstructed Matrix $\hat{X}$
Ensure: Faulty Data Detection Matrix D
for $i \leftarrow 1$ **to** n **do**
for $i \leftarrow 1$ **to** t **do**
if $E(i,j) = 0$ **then**
continue ;
end if
if $\delta < thres_l$ **and** $D(i,j) = 1$ **then**
$D(i,j) = 0$;
end if
if $\delta > thres_u$ **and** $D(i,j) = 0$ **then**
$D(i,j) = 1$;
end if
end for
end for
return D;

5 PERFORMANCE EVALUATION

5.1 Evaluation Settings

Dataset for evaluation is the same as Section 3. We select a complete subset that contains 617 roads × 288 slots, with slot intervals of 5 min.

Generation of missing and faulty data: For the generation of missing data, we rely on the generation of Existence Matrix E. We set E to all ones first, then select a fraction of elements randomly in E and set them to zeros with the control of missing ratio (β) predefined. The generation of faulty data is done in a similar way. We set Faulty Data Matrix F to all zeros at first, and then select a fraction of elements to ones with the control of faulty ratio (α). Besides that we add a random bias ξ to corresponding elements in X, which are marked as faulty data. Thus, we can get the Direct Sensory Matrix S as:

$$S(i,j) = X(i,j)E(i,j) + \xi_{i,j} \cdot F(i,j) \tag{27}$$

where $\xi_{i,j}$ means the random value of bias here.

Evaluation criteria: The performance of ODC will be analyzed in faulty data discrimination and missing data reconstruction. For faulty data discrimination, we evaluate the performance through *Precision*, *Recall*, and *Accuracy*, which are defined as:

$$Precision = \frac{TP}{TP+FP}, \quad Recall = \frac{TP}{TP+FN}, \quad Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \tag{28}$$

where TP, FP, TN, and FN are:

- True Positive: considered faulty data, and indeed, is faulty data
- False Positive: considered faulty data, but indeed, is normal data
- True Negative: considered normal data, and indeed, is normal data
- False Negative: considered normal data, but indeed, is faulty data

For missing data reconstruction, we evaluate the performance through ER, which represents the average values of reconstruction errors in missing and faulty data, which is defined as:

$$ER = \frac{\sum_{where\ B(i,j)=0} \left| X(i,j) - \hat{X}(i,j) \right|}{\sum_{where\ B(i,j)=0} 1} \tag{29}$$

To verify the effectiveness of the ODC method, we select another two methods when evaluating the performance in faulty data discrimination for comparison:

- ODC-without-HT: ODC method without temporal-spatial improvement.
- FDD: The faulty data detection method that is proposed in Section 4.3.

And another three methods when evaluating missing data reconstruction:

- ODC-without-HT: ODC method without temporal-spatial improvement.
- ASD: Compressive sensing algorithm applied in ODC, which is used for reconstructing missing data.
- ASD-without-HT: This is similar to ASD, the difference is that they are not improved by temporal-spatial features.

5.2 Performance Analysis: Faulty Data Discrimination

In this section, we evaluate the performance in faulty data discrimination. We conduct three experiments in this evaluation, where missing data ratio β = 10%, 30%, and 50%, respectively, and faulty data ratio α varies from 10% to 50% in each experiment. The result of evaluation is shown in Fig. 5.

We observe that the performance of ODC-like methods in precision is better than FDD, and the gap between ODC and ODC-without-HT is not large. Comparing with Fig. 5A–C, we can observe that precision reaches the maximum value when the faulty data ratio $\alpha = 30\%$, and precision decreases with the growth of missing data ratio β on the whole. Because of the intervals of traffic information from the AMAP update is a bit long (i.e., 5 min), the variation range of

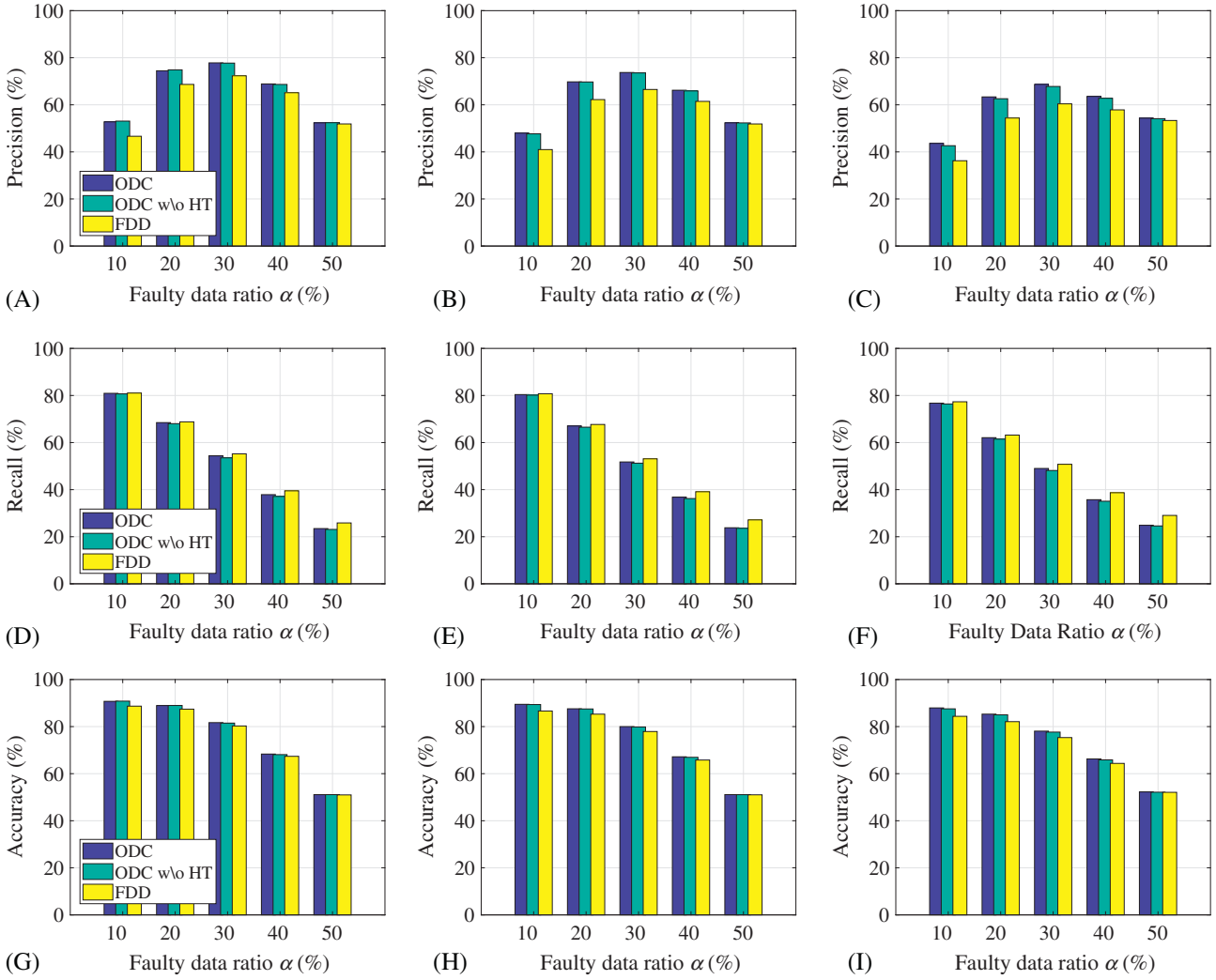


FIG. 5 Performance analysis in faulty data discrimination. (A) Missing data ratio $\beta = 10\%$; (B) missing data ratio $\beta = 30\%$; (C) missing data ratio $\beta = 50\%$; (D) missing data ratio $\beta = 10\%$; (E) missing data ratio $\beta = 30\%$; (F) missing data ratio $\beta = 50\%$; (G) missing data ratio $\beta = 10\%$; (H) missing data ratio $\beta = 30\%$; (I) missing data ratio $\beta = 50\%$.

mean velocity cannot be well defined, which leads to discriminating faulty data with difficulty. But it is clear that ODC-like methods do help to improve faulty data discrimination.

The recall of three methods decrease with the growth of the faulty data ratio, and the gap among them is not large. In terms of accuracy, we can observe that ODC-like methods outperform the FDD method. The accuracy of ODC-like methods is above 80% when faulty ratio $\alpha \leq 30\%$ and missing ratio $\beta \leq 30\%$, even when missing ratio is 50%, there remains nearly 80% accuracy in ODC-like methods.

In summary, the differences of reconstruction in ODC-like methods do not influence the performance of faulty data discrimination much, and ODC-like methods outperform FDD method in terms of precision and accuracy on the whole.

5.3 Performance Analysis: Missing Data Reconstruction

In this section, we evaluate the performance in missing data reconstruction. The experiments are conducted when missing data ratio $\beta = 10\%$, 30%, and 50%, and faulty data ratio α varies from 10% to 50% in each experiment. The result is shown in Fig. 6. It is obvious that the ODC method outperforms another three methods.

In Fig. 6, we can observe that when the faulty ratio is small (i.e., $\alpha = 10\%$), the performance of ODC and ODC-without-HT are almost the same. And the gap of the reconstruction error between ODC and ODC-without-HT increase with the growth of the faulty data ratio. We notice that when the faulty ratio $\leq$20%, the performance of

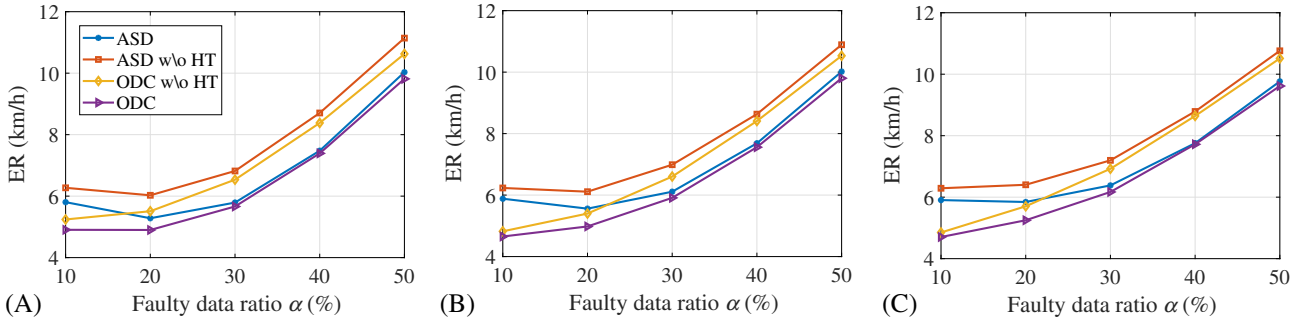


FIG. 6 Performance analysis in missing data reconstruction. (A) Missing data ratio $\beta = 10\%$; (B) missing data ratio $\beta = 30\%$; (C) missing data ratio $\beta = 50\%$.

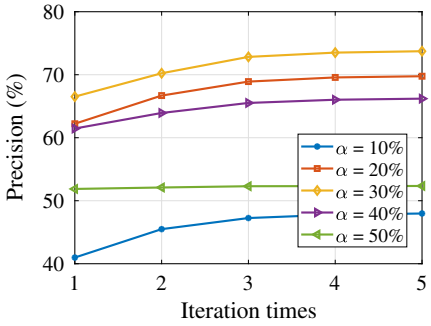


FIG. 7 Convergence rate of ODC when $\beta = 30\%$.

ODC-without-HT is better than ASD. However, when the faulty ratio is larger than 20%, ASD outperforms ODC-without-HT. We can conclude that temporal-spatial features play an important role in improving performance of missing data reconstruction.

The reconstruction error of the ODC method is only about 6 km/h, even when faulty ratio $\alpha = 30\%$ and missing ratio $\beta = 50\%$, which can be tolerated. The performance of ODC is improved by about 15% compared with ASD-without-HT on average.

5.4 Convergence of ODC

We analyze the convergence rate of ODC at last. Fig. 7 shows that when missing ratio $\beta = 30\%$, faulty ratio varies from 10% to 50%, the precision will be convergent within five times of iterations. We can notice that the first three iterations contribute to nearly all improvement, and later iterations contribute to little improvement. Thus we can conclude that the convergence of ODC is good.

6 RELATED WORK

The rapid growth in the number of motor vehicles causes heavy traffic jams, drivers are now mostly dependent on navigation applications before going out. But missing data and faulty data are common in data from sensors [10, 13, 14]. There is a great deal of research in this area.

SEER [15] figure out the redundancy of traffic datasets and utilize multichannel singular spectrum analysis to recover missing data. However, SEER does not take faulty data into consideration. The traditional outlier detection technique [16, 17] can be an alternative method for faulty data detection, but this technique cannot tolerate a high rate of missing data. Yoon et al. [3] estimate traffic conditions through traces of vehicles in each road segment, which needs large amounts of data to build a road segment model. And each road segment can only be analyzed independently, which is not suitable for real-time applications. Cheng et al. [5] propose the DECO model, and utilize compressive sensing techniques to reconstruct missing data, and detect faulty data based on users' reputations. They believe that users who have a good reputation will not upload faulty data, which may not necessarily be true.

Although some methods and algorithms have been proposed, few of them can effectively exploit the hidden structures in traffic condition datasets. The compressive sensing [18, 19] technique can tolerate a high rate of missing data, and is good for recovering missing data in a low rank matrix. However, a compressive sensing technique cannot be directly applied to traffic condition datasets because of the existence of faulty data, which may lead to reconstruction deviation [20]. Thus, we need to discriminate faulty data before recovering missing data.

7 CONCLUSION

In this chapter, we proposed an ODC method to discriminate misleading information and correct missing data in intelligent transportation systems. First, we uncovered the common problem that misleading and missing information exists in traffic condition datasets. Then we mined the hidden structures in traffic condition datasets, such as the low rank feature, and temporal-spatial correlation. On the basis of these observations, we have designed an outlier detection method to discriminate faulty data. And then we utilized temporal-spatial features to improve performance in missing data reconstruction based on a compressive sensing technique. Extensive evaluations are conducted using real urban traffic datasets, which demonstrates that ODC can produce only about a 6 km/h reconstruction error, even when the missing ratio is 50% and the faulty ratio is 30%.

In this work, we only utilize a single dimension of traffic conditions (i.e., the mean velocity of roads), which brings limited improvement. In the future, we can mine more dimensions of traffic conditions by using raw GPS data collected from vehicles, such as the number of vehicles on each road, and the headway directions. Also, we can utilize some information from map data such as lanes of roads, road speed limitations, locations of traffic lights, and so forth, which can help us improve performance in discrimination and correction.

References

[1] S. Nikolic, V. Penca, M. Segedinac, Semantic web based architecture for managing hardware heterogeneity in wireless sensor network, in: International Conference on Web Intelligence, Mining and Semantics, WIMS 2011, Sogndal, Norway, May, 2011, pp. 1–9.

[2] AMAP, Traffic condition data collected from AMAP. Available from: http://lbs.amap.com/api/webservice/guide/api/trafficstatus/ (Accessed 15 January 2018).

[3] J. Yoon, B. Noble, M. Liu, Surface street traffic estimation, in: International Conference on Mobile Systems, Applications and Services, 2007, pp. 220–232.

[4] Z. Li, Y. Zhu, H. Zhu, M. Li, Compressive sensing approach to urban traffic sensing, in: International Conference on Distributed Computing Systems, 2011, pp. 889–898.

[5] L. Cheng, J. Niu, L. Kong, C. Luo, Y. Gu, W. He, S.K. Das, Compressive sensing based data quality improvement for crowd-sensing applications, J. Netw. Comput. Appl. 77 (2016) 123–134.

[6] E.J. Candes, The restricted isometry property and its implications for compressed sensing, C. R. Math. 346 (9–10) (2008) 589–592.

[7] E.J. Candes, B. Recht, Exact matrix completion via convex optimization, Found. Comput. Math. 9 (6) (2008) 717.

[8] B. Recht, W. Xu, B. Hassibi, Necessary and sufficient conditions for success of the nuclear norm heuristic for rank minimization, IEEE Conf. Decis. Control 16 (5) (2008) 3065–3070.

[9] B. Recht, M. Fazel, P.A. Parrilo, Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization, SIAM Rev. 52 (3) (2007) 471–501.

[10] L. Kong, M. Xia, X.Y. Liu, M.Y. Wu, X. Liu, Data loss and reconstruction in sensor networks, in: INFOCOM, 2013 Proceedings IEEE, 2013, pp. 1654–1662.

[11] J. Tanner, K. Wei, Low rank matrix completion by alternating steepest descent methods, Appl. Comput. Harmon. Anal. 40 (2) (2016) 417–429.

[12] F. Wu, D. Liu, Z. Wu, Y. Zhang, G. Chen, Cost-efficient indoor white space exploration through compressive sensing, IEEE/ACM Trans. Netw. (99) (2017) 1–17.

[13] M. Balazinska, A. Deshpande, M.J. Franklin, P.B. Gibbons, J. Gray, M. Hansen, M. Liebhold, S. Nath, A. Szalay, V. Tao, Data management in the worldwide sensor web, IEEE Pervasive Comput. 6 (2) (2007) 30–40.

[14] H. Kurasawa, H. Sato, A. Yamamoto, H. Kawasaki, M. Nakamura, Y. Fujii, H. Matsumura, Missing sensor value estimation method for participatory sensing environment, in: IEEE International Conference on Pervasive Computing and Communications, 2014, pp. 103–111.

[15] H. Zhu, Y. Zhu, M. Li, L.M. Ni, SEER: metropolitan-scale traffic perception based on lossy sensory data, in: INFOCOM, 2009, pp. 217–225.

[16] A.J. Fox, Outliers in time series, J. R. Stat. Soc. 34 (3) (1972) 350–363.

[17] V. Hodge, J. Austin, A survey of outlier detection methodologies, Artif. Intell. Rev. 22 (2) (2004) 85–126.

[18] D.L. Donoho, Compressed sensing, IEEE Trans. Inf. Theory 52 (4) (2006) 1289–1306.

[19] E.J. Candes, T. Tao, Near-optimal signal recovery from random projections: universal encoding strategies? IEEE Trans. Inf. Theory 52 (12) (2006) 5406–5425.

[20] E.J. Candes, Y. Plan, Matrix completion with noise, Proc. IEEE 98 (6) (2009) 925–936.