GPSClean: A Framework for Cleaning and Repairing GPS Data

CHENGLONG FANG and FENG WANG, Nanjing University of Aeronautics and Astronautics, China
BIN YAO, Shanghai Jiao Tong University, China
JIANQIU XU, Nanjing University of Aeronautics and Astronautics, China

The rise of GPS-equipped mobile devices has led to the emergence of big trajectory data. The collected raw data usually contain errors and anomalies information caused by device failure, sensor error, and environment influence. Low-quality data fails to support application requirements and therefore raw data will be comprehensively cleaned before usage. Existing methods are suboptimal to detect GPS data errors and do the repairing. To solve the problem, we propose a framework called *GPSClean* to analyze the anomalies data and develop effective methods to repair the data. There are primarily four modules in *GPSClean*: (i) data preprocessing, (ii) data filling, (iii) data repairing, and (iv) data conversion. For (i), we propose an approach named **MDSort (Maximum Disorder Sorting)** to efficiently solve the issue of data disorder. For (ii), we propose a method named **NNF (Nearest Neighbor Filling)** to fill missing data. For (iii), we design an approach named **RCSWS (Range Constraints and Sliding Window Statistics)** to repair anomalies and also improve the accuracy of data repairing by mak7ing use of driving direction. We use 45 million real trajectory data to evaluate our proposal in a prototype database system SECONDO. Experimental results show that the accuracy of RCSWS is three times higher than an alternative method SCREEN and nearly an order of magnitude higher than an alternative method EWMA.

CCS Concepts: • **Information systems** → **Data cleaning**; *Spatial-temporal systems*; *Mobile information processing systems;*

Additional Key Words and Phrases: Trajectory data, data detection, data repairing, data cleaning

ACM Reference format:
Chenglong Fang, Feng Wang, Bin Yao, and Jianqiu Xu. 2022. GPSClean: A Framework for Cleaning and Repairing GPS Data. *ACM Trans. Intell. Syst. Technol.* 13, 3, Article 40 (April 2022), 24 pages.
https://doi.org/10.1145/3469088

1 INTRODUCTION

The widespread use of GPS-enabled devices such as mobile phones has made the recording of position data very easy and consequently a large amount of such data is collected. Due to a number of factors such as device failure, sensor error, and environment influence, raw data usually contain

This work is supported by National Natural Science Foundation of China (61972198), Natural Science Foundation of Jiangsu Province of China (BK20191273).
Authors' addresses: C. Fang, F. Wang, and J. Xu (corresponding author), Nanjing University of Aeronautics and Astronautics, 29 Jiangjun Road, Nanjing, Jiangsu Province, China, 211106; emails: {fangcl, marvel_agent, jianqiu}@nuaa.edu.cn; B. Yao, Shanghai Jiao Tong University, Shanghai, China; email: yaobin@cs.sjtu.edu.cn.

2157-6904/2022/04-ART40 $15.00
https://doi.org/10.1145/3469088

dirty, abnormal, and incorrect information [29, 33], which will seriously affect data usage such as doing analytics and querying processing. High-quality GPS data can be utilized in a wide range of applications such as intelligent transportation planning and optimization [9–12, 23, 39, 55, 56], trip planning [18], and urban management [42, 46, 57].

In this article, we study effectively detecting and efficiently repairing GPS data anomalies. We propose a framework named *GPSClean*, as illustrated in Figure 1. The framework is mainly composed of four components: (i) data preprocessing; (ii) data filling; (iii) data repairing; and (iv) data conversion. *Data preprocessing* performs data sorting and removes redundancy data. *Data filling* is to fill missing GPS data, and *data repairing* deals with repairing drifting data. Drifting data refers to data records that do not conform to the driving rules and clearly deviate from the driving trajectory. The last component, *data conversion,* transforms cleaned GPS records into data forms managed by the database system.

There are two major challenges when performing GPS data cleaning: (i) data detection and (ii) data repairing. In particular, data detection is a fundamental step for data repairing. In the literature, there are a number of proposals dealing with data detection and repairing. The method [25] treats abnormal data points as useless information and discards them without doing repairing. If the number of such data points is large, the produced data will be small and may be not useful for doing comprehensive analysis. In contrast, we aim at not only comprehensively detecting abnormal data points but also repairing them to preserve data integrity. Simultaneously clustering and cleaning dirty data is done in [43]. The approach repairs noisy data and makes the boundary point of the nearest cluster to improve the degree of data aggregation. The algorithm has a significant effect on repairing time series data, but the accuracy of repairing GPS anomalies is not high. This is because the algorithm modifies the abnormal data to the nearest boundary point on the nearest cluster, which is the same as deleting the original noise data and adding the duplicate data to the original data. An algorithm called *SCREEN* is proposed by Song et al. [45] to detect and repair abnormal data through speed-based constraints. *SCREEN* can well detect and repair the data with large abnormal values in time series data that violate the speed constraint, but the method is difficult in detecting and repairing the data fulfilling the constraint condition with small deviation trajectories. For data points that deviate far from the correct movement, the constraint-based method performs anomaly detection well, but the repairing accuracy is low. *SCREEN* does not detect drifting points fulfilling the constraints and thus will not do the repairing. The paper [16] proposes a method named **Sorted-Neighborhood Method (SNM)** to detect duplicate data in a window through an adaptive window method.

Regarding the issue of data repairing, most current algorithms only use constraint-based or statistical-based methods to repair abnormal data. The smoothing of **Moving Average (MA)** model [7] is likely to modify correct data and thus generates noise data. The approach making use of speed constraints only repairs the data exceeding the maximum threshold. The data with a small deviation is not repaired. The paper [54] uses statistical-based maximum likelihood estimation, while the efficiency is low as they count the changes in the acceleration of the entire dataset. The method has a good effect on repairing data points with small deviations, but using the maximum probability of data variation makes the repairing procedure smooth. The paper [45] shows that the smoothing method will make the data over-repaired, i.e., almost all data points are modified, most of which are originally correct and do not require repair. Similarly, relying only on the possibility of data changes also causes over-repairing.

In this article, we detect five types of abnormal data including duplicate data, out-of-order data, invalid data, missing data, and drifting data. Different types of abnormal data cause challenges in detecting and distinguishing abnormal types. Duplicate data is caused by repeated data storage. Out-of-order data is incorrect data due to network transmission delay. Other abnormal data are

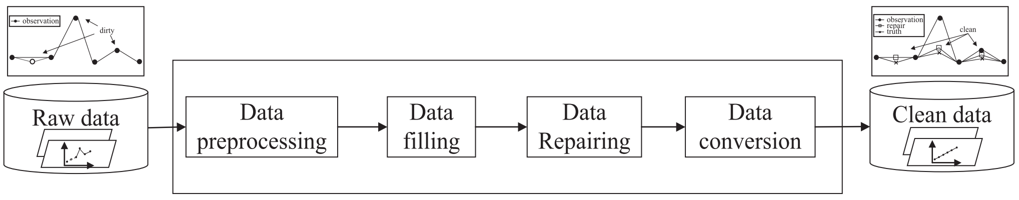


Fig. 1. The framework of GPSClean.

mostly caused by device positioning problems, and the noise data caused by the equipment positioning problem is the most important part. Detecting invalid data and missing data is straightforward. Invalid data is determined by whether the latitude or longitude of the data is out of the range. Missing data is determined by comparing the time difference between the two data and the frequency of data collection. To detect drifting data, we propose range constraints and sliding window statistics. Based on the original data, we propose an algorithm utilizing the driving direction and further improve the accuracy by analyzing the changes in the data. Our method does not only detect large abnormal data points, but also implements the detection of small abnormal data by statistical methods. The efficiency of duplicate data detection depends on the data order. We propose an efficient method to perform the sorting with a constraint condition to avoid sorting the overall data set, which is a time-consuming task.

The quality of anomaly repairing depends on (i) whether the detection algorithm can find out all possible anomalies, and (ii) whether the approach fully makes use of the original data and performs a thorough analysis. Repairing duplicate and disordered data are processed at the abnormal detection stage. In the case of restricting offsets in disordered data, we develop a sorting algorithm based on heap sorting and achieve the time complexity of $O(N \cdot \lg c)$ in which N is the number of sorted objects and c is the disorder offset constant of the original data. When data is sorted already, duplicate data is deleted by comparing whether the current and the subsequent data are the same. We propose a sliding window-based probability model and range constraint method which is consistent with the minimum change principle [5] and avoiding over-repairing. This method supports repairing the data that largely deviates from the correct one and fulfills the constraint condition.

This article makes the following contributions:

- We define five types of noises in GPS data.
- We develop a sorting method for GPS records achieving the time complexity of $O(N \cdot \lg c)$, in which N is the number of data points and c is the disorder offset constant in raw data.
- We propose a sliding window-based probabilistic detection model and a range constraint method to detect and repair GPS data anomalies. Furthermore, we propose an anomaly detection and repairing algorithm based on the driving direction and follow the minimum change principle to avoid over-repairing.
- We do the evaluation on large real datasets and the results demonstrate that the repairing accuracy of RCSWS is three times higher than SCREEN and nearly an order of magnitude higher than EWMA.

The rest of the article is organized as follows: Related work is discussed in Section 2. Basic concepts and the framework *GPSClean* are introduced in Section 3. In Section 4, we present the maximum disorder sorting algorithm. In Section 5, we introduce the nearest neighbor filling algorithm. We propose the algorithm combining range constraint and sliding window statistics, and analyze the abnormal repairing of data by direction in Section 6. The experimental evaluation is performed in Sections 7 and 8 concludes the article.

2 RELATED WORK

In this section, we make some introductions from two aspects of previous related work. We investigate the anomaly problems in different types of data and related detection algorithms and introduce relevant literature for mainstream anomaly repairing algorithms.

2.1 Data Anomaly Detection

Various fields have done relatively more work in data anomaly detection. In single-point anomaly detection, the predicted value is compared with the observed value to determine the abnormal data. The paper [4] uses the median of timestamp data as a prediction model to predict the value of the currently detected data. Then the difference between the predicted value and the observed value is compared with the error threshold to detect the abnormal situation of the current data. The paper [41] uses a cluster-based method to detect anomalies in power data in a distributed environment, and then uses an exponential weighted average method to repair the data. The same is to determine whether the data is abnormal by analyzing and comparing the predicted value and the observed value of the data. Jeung et al. [30] propose a flexible trajectory modeling method to infer the actual position and nondeterministic uncertainty ranges. In the detection of data anomalies, the range of data points can roughly estimate the abnormality of the data, and improve the data quality of the trajectory. Since trajectories are often uncertain, Ma et al. [35] propose top-*k* trajectory similarity query method to quantify the similarity between uncertain trajectories. The abnormal data in the uncertain trajectory can be detected by comparing the similar real trajectory data points that have been queried. Mautz et al. [37] propose a deep embedded cluster tree, which is a divisive hierarchical embedded clustering method. By combining the features of GPS data to perform clustering, the detected abnormal clusters where the number of points in the cluster after clustering is less than a certain threshold can be regarded as the detected abnormal cluster. Ichikawa et al. [28] propose a qualitative evaluation method based on binary features of products. In the detection of the dataset, the quality of the data can be checked by preliminarily scoring the GPS dataset. According to the quality of the dataset, the abnormal data points in the GPS data can be preliminarily judged. Patil et al. [40] propose an optimal number estimation method based on deep clustering, which can adjust the optimal number of clusters through different input parameters. The optimal number clustering of the GPS dataset can be used to detect data that deviates from the optimal data point as the detected abnormal data. **Moving Average (MA)** algorithm [7] is also widely used in the detection of single point anomalies. The average value of the recently observed time series is used as the predicted value of the current detection data to determine the abnormal point. For **Weighted Moving Average (WMA)** algorithm, the current prediction value is calculated by assigning different weights to the observation data at different positions in the time series data according to the influence on the current detection data. Song et al. [45] propose a speed constraint method that constrains the range of current data changes by changing the sequence data in the previous period. Determining abnormal data based on whether the currently detected data is within the constraints. **Autoregressive (AR)** model [27] and **Autoregressive with exogenous input (ARX)** model [8] are widely used in data anomaly detection in many fields. Alengrin et al. [2] propose **Autoregressive Moving Average (ARMA)** model based on AR model and MA model. Box et al. [6] propose a more complex **Autoregressive Integrated Moving Average (ARIMA)** model based on ARMA model for data anomaly detection.

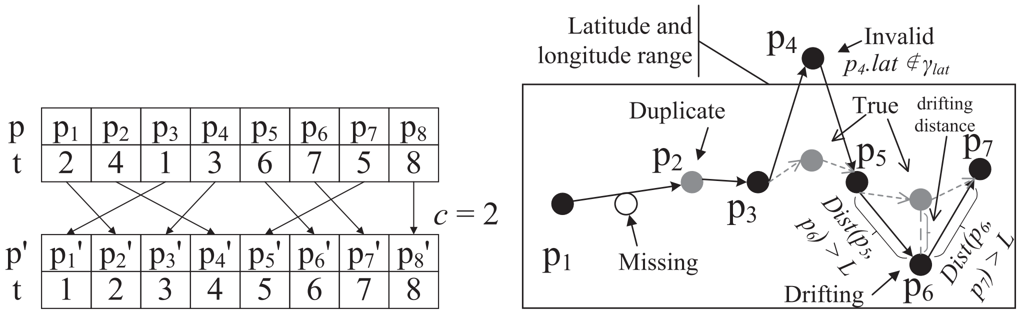
In terms of sequence data anomaly detection, Keogh et al. [31] propose a method for sequence anomaly detection. This method determines whether the subsequence is an abnormal sequence by calculating whether the matching distance between the subsequence and its closest sequence is greater than a certain threshold. In the density-based anomaly detection method, DBSCAN algorithm [13, 14] divides the data into a cluster by the density of the data. The outliers in the data are calculated through clustering, and the data with large deviations are regarded as abnormal data. Density-based methods can detect anomalies in dense data well, but discarding the detected anomalies will destroy the integrity of the original data. In terms of using machine learning methods to detect abnormal points in data, the method of **Generative Adversarial Networks (GANs)** [17, 32, 47] simultaneously trains two models, one is used to generate the distribution model of the data, and the other is used to generate the discriminant model to determine whether the data is abnormal. The errors of the two models are calculated by backpropagation and gradient descent methods and the corresponding weights are updated to minimize the loss of the generated two models. Gers et al. [20] improve **Recurrent Neural Network (RNN)** by using **Long Short-Term Memory (LSTM)** to avoid the disappearance of the RNN gradient, which makes RNN detect abnormal data in time series well.

2.2 Data Anomaly Repairing

To make full use of existing data to obtain high-quality data, data repairing methods in various fields are also emerging endlessly. Most of the data anomaly detection methods mentioned above can also perform data anomaly repairing. In terms of smoothing-based data repairing technology, MA sequence cleaning algorithm [7] cleans abnormal data by calculating the average value of the data over some time as the data repairing value. The data cleaning algorithm of AR model [27, 50] uses the current data as a regression variable, and describes the regression process of the current data through the weighted linear combination of the first k data. These papers [19, 36, 58] use the Kalman filter model to clean the data. The Kalman filter model can handle not only stationary signals but also non-stationary signals and multi-dimensional signals. Although the smoothing repairing algorithm has a high accuracy for repairing some data with relatively small outliers, for some data with large outliers, more data anomalies will be caused due to the smooth transition process.

In terms of constraint-based data cleaning technology, the paper [15] proposes the method of **Order Dependencies (ODs)**. This method determines whether the data violates the corresponding constraint through the dependence relationship between the data. For example, the fuel consumption of a car can only be a positive number. If the data is negative, the constraint is violated and the data is regarded as abnormal data. Then, the abnormal data is repaired by the normal data around the time series. Lopatenko et al. [34] propose a method based on **Denial Constraints (DCs)** to detect and repair anomalies in numerical time series data. The idea of DCs method and ODs method is roughly the same, and the data is determined and repaired through the dependence relationship between the data. The paper [22] proposes **Sequential Dependencies (SDs)**, which determine the abnormality of the data by comparing the difference between consecutive data in the time series data. This method makes a preliminary determination based on whether the difference between the current data point and the previous data point meets the threshold and then determines whether the data is in the corresponding interval to detect and repair the data. The paper [45] proposes the speed constraint method SCREEN, which detects anomalies by constraining the changes of the data and uses the maximum change boundary of the data to repair the abnormal data. SCREEN modifies the abnormal data beyond the change range to the data value of the maximum change boundary as the repair value of the abnormal data. Other constraint-based data cleaning methods include variance constraints [51] and similarity rule constraints [44].

In terms of statistics-based data cleaning technology, Zhang et al. [54] use the maximum likelihood estimation method to clean the abnormal data in the time series data. This method makes a probability aggregation function by counting the distribution of data changes in time-series data, and repairs abnormal data according to the maximum probability of data changes in the overall



(a) Maximum disorder offset c = 2 (b) Different kinds of anomalies

Fig. 2. Examples of GPS anomalies.

data. These papers [21, 48, 49] use the maximum likelihood method to detect and repair anomalies of different types of data. Jeffery et al. [29] propose an adaptive **Radio Frequency Identification (RFID)** data cleaning method, which uses techniques based on sampling and smoothing to improve the quality of RFID data. Gupta et al. [24] use **Hidden Markov Model (HMM)** to predict the price of stocks, and the paper [3] also uses HMM to detect and clean up some abnormal data in time series data. The paper [38] uses **Spatio-Temporal Probabilistic Model (STPM)** to learn some data characteristics of historical data, and then uses the model to clean the current data.

3 THE FRAMEWORK

We introduce basic concepts in Section 3.1 and outline the framework *GPSClean* in Section 3.2.

3.1 Basic Concepts

Let $P = \{p_1, \ldots, p_n\}$ be a set of GPS records. Each p consists of a set of items, among which essential items are t (time), lon (longitude) and lat (latitude). Other items are for speed, direction and acceleration. Preliminary concepts are introduced in the following.

Definition 1 (Maximum Disorder Offset). Let P' be an ordered set on P by time such that $\forall p_i \in P$: $\exists p'_j \in P', p_i = p'_j \wedge |j - i| \leq c$. We call c the maximum disorder offset.

The maximum disorder represents that the raw data finds the corresponding position in sorted data within a fixed range. Let P be the unordered set of original GPS records, the maximum disorder offset of P is c, $p_i \in P$, and P' is an ordered sequence of P, $p'_i \in P'$, hence $p_i \in \{p'_{i-c}, \ldots, p'_{i+c}\}$. For example, the maximum disorder offset of the dataset is 2, and p_3 is repaired to p'_1 in the set $\{p'_1, p'_2, p'_3, p'_4, p'_5\}$, as shown in Figure 2(a).

Definition 2 (Out-of-order Anomaly). The set $P = \{p_1, \ldots, p_n\}$ is called out-of-order anomaly if $\exists$ $p_i, p_j \in P$: $i < j \wedge p_i.t > p_j.t$.

Out-of-order abnormalities are caused by signal delays that cause the located data to be stored later such as p_3, p_4, and p_7 in Figure 2(a).

Definition 3 (Duplicate Anomaly). The set $P = \{p_1, \ldots, p_n\}$ is called duplicate anomaly if $\exists$ p_i, p_j $\in P$: $i \neq j \wedge p_i = p_j$.

Definition 4 (Invalid Anomaly). γ_{lon} and γ_{lat} are the longitude and latitude range of the collection site, respectively. Given a set P, where $\exists$ $p_i \in P$: $p_i.lon \notin \gamma_{lon} \vee p_i.lat \notin \gamma_{lat}$.

Definition 5 (Missing Anomaly). T_f is the data collection frequency. Given a set P, where $\exists$ p_i, $p_j \in P$: $i < j \wedge p_j.t - p_i.t \geq T_f * (j - i + 1)$.

Table 1. Notations

Symbol	Description
P	a set of GPS records containing multiple p
P'	repair of set P
p_i	i-th positioning data
v_i	approximate speed of point p_i
lon, lat	longitude and latitude
T_f	data collection frequency
L_e	error range of data collected by equipment
$\gamma_{lon}, \gamma_{lat}$	longitude and latitude range of the data collection place
$Dist(p_i, p_j)$	distance between p_i and p_j

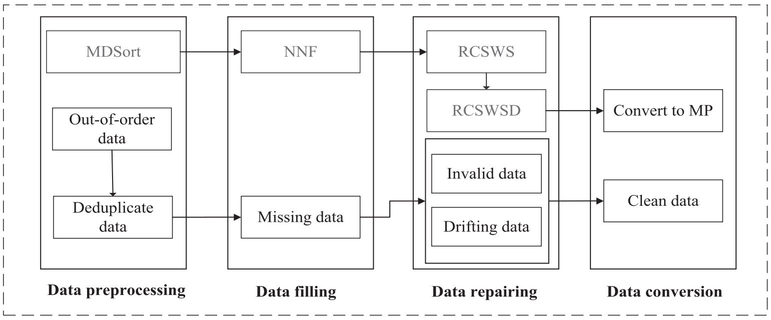


Fig. 3. Detailed description of primary components.

Definition 6 (Drifting Anomaly). $Dist(p_i, p_j)$ is the distance between the positions of point i and point j in P, and L is the threshold of the maximum distance traversed within the sampling frequency T_f. Given a set P, $\forall\, p_i \in P, p_i.lon \in \gamma_{lon} \wedge p_i.lat \in \gamma_{lat}$ where $\exists\, p_k \in P$: $Dist(p_{k-1}, p_k) > L \wedge Dist(p_k, p_{k+1}) > L$.

Definition 7 (Data Range). Given a data point p_i in the set P, the current speed of p_i is v_i, and the distance passed by the maximum acceleration is L within the fixed collection frequency T_f. The circular area formed by taking p_i as the center and L as the radius is the data range of p_i.

Figure 2(b) shows the remaining anomalies. The missing anomaly is shown by hollow point. Duplicate anomaly is multiple GPS records having the same data such as p_2. Invalid anomaly is the value greater than the valid range such as p_4. The drifting data is the value within the range, but the distance between the two points exceeds the set maximum threshold such as p_6. Table 1 summarizes frequently used notations in the article.

3.2 An Overview

We elaborate primary components of *GPSClean* in Figure 3. The framework solves the problems of out-of-order and duplicate by an algorithm named *MDSort*. Then, the *NNF* algorithm solves the missing abnormal data. To repair invalid and drifting data, we design two algorithms named *RCSWS* and *RCSWSD*, respectively, in which one is for the data without direction and the other is for the data with direction. Finally, the framework converts the repaired data into data formats in a database system.

We make full use of attributes in raw data for anomaly repairing. Depending on data characteristics, corresponding detection and repairing algorithms are called. Existing repairing algorithms may incur over-repair situations, which destroy the correct data in the original data. To avoid over-repairing, we follow the minimum change principle. The overall algorithm is given in Algorithm 1.

ALGORITHM 1: *GPSClean*

```
Input:  trajectory dataset $P$,
        maximum disorder offset $c$,
        probability threshold $\sigma$,
        direction change threshold $\alpha$
Output: $P'$
1:  if $P$ not sorted then
2:      $P \leftarrow MDSort(P, c)$ // Algorithm 2, MDSort
3:  $T_f \leftarrow$ get the frequency of the dataset $P$
4:  $hasDir \leftarrow$ get the direction of the dataset $P$
5:  $P_s \leftarrow$ segment the trajectory of the $P$
6:  for $P_i \in P_s$ do
7:      for $p_j \in P_i$ do
8:          while $p_j = p_{j+1}$ do
9:              delete $p_{j+1}$
10:         if $p_{j+1}.t - p_j.t \geq 2 \cdot T_f$ then
11:             $P_i' \leftarrow NNF(P_i, j, j+1)$ // Algorithm 3, NNF
12:         if $hasDir$ then
13:             $P_i' \leftarrow RCSWSD(P_i, j, T_f, \sigma, \alpha)$ // Algorithm 5, RCSWSD
14:         else
15:             $P_i' \leftarrow RCSWS(P_i, j, T_f, \sigma)$ // Algorithm 4, RCSWS
16:     $P_s' \leftarrow P_s' \cup P_i'$
17: $P' \leftarrow \cup P_s'$
18: return $P'$
```

4 PREPROCESSING

4.1 Data Dependency

A GPS record at time t_i is related to the record at time t_{i-1}. Taking into account the dependency between two consequent GPS records, the method utilizing the sliding window can solve the problem of data anomaly detection and repairing. To achieve high efficiency and accuracy of anomaly detection, we design a statistical method based on the sliding window. To support an automatic anomaly detection for different types of GPS datasets, we further adapt the algorithm when performing anomaly detection.

4.2 Sorting Data and Removing Duplication

Due to signal delay, the time storing the data is usually later than the actual time. This results in disordered GPS records. We design a sorting algorithm based on heap sorting, named **MDSort (Maximum Disorder Sorting),** to enhance the efficiency of processing disordered data. The MD-Sort algorithm sorts the data in the window. The size of the window is the size of the heap in the sorting algorithm. The idea of the algorithm is to pre-store the data collected by the device in advance through the characteristics of the heap structure, then wait for the delayed data to arrive and sort. We first determine the maximum time for data delay storage through an external device, and calculate the number of data points collected during the delay time based on the frequency. Then, we initialize the size of the heap to the number of data points saved in advance. In the process of sorting data, we compare the time of the heap vertex data with the time of the sorted data. If the compared data is not within the continuous collection frequency, we expand the heap size to two times the current size. After adding the expanded new data, we again compare the data at the top of the heap with the sorted data. If the time of the data is continuous, the data at the top of the heap will be taken out, and if the time is not continuous, the heap size will continue to be expanded to twice the current size. We provide the complete algorithm in Algorithm 2.

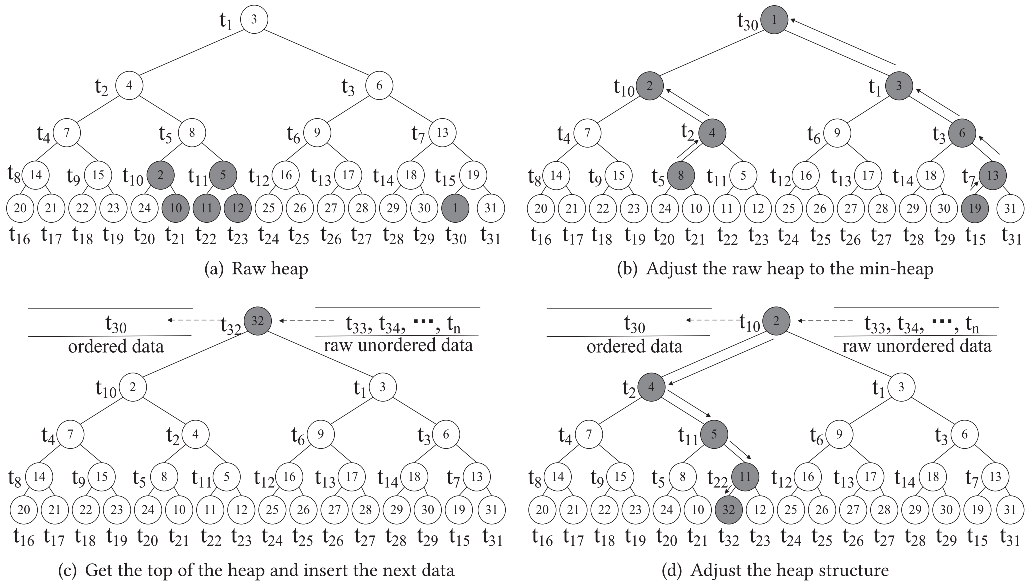
ALGORITHM 2: *MDSort*

```
Input:  trajectory dataset $P$,
        maximum disorder offset $c$
Output: $P'$
 1: $H \leftarrow$ build a min-heap sorted by time for the first $c$ records
 2: for $i \leftarrow c + 1$ to $N$ do
 3:     if $H.peek()$ is not next data then
 4:         $c \leftarrow c * 2$
 5:         while $i < N \wedge H.size < c$ do
 6:             $H.push(p_i)$
 7:             $++i$
 8:     else
 9:         $P' \leftarrow H.pop()$
10:         $H.push(p_i)$
11: while $H$ is not empty do
12:     $P' \leftarrow H.pop()$
13: return $P'$
```

Assume that the maximum delay is 60 seconds and the data acquisition frequency is 2 seconds. Therefore, the number of records during the period is $\frac{60}{2} = 30$. As a result, the maximum data disorder position is 30. For the sake of implementation, we take 31 data points to construct a min-heap. We use the first 31 records to build a heap, as shown in Figure 4(a). Numbers are to represent the time, and grey nodes represent delayed stored data. The min-heap is built on the time order, and grey nodes are the adjusted nodes, as shown in Figure 4(b). Since the maximum offset is 30, even if a record reaches the furthest offset such as the node t_{30}, the correct order can still be obtained after sorting the heap. Therefore, the top element in the heap is the first. We put the top of the heap into a queue, and push the next one into the heap, as shown in Figure 4(c). Then, we maintain the heap and have the data at the top be the second one in order, as shown in Figure 4(d). Repeating the steps, we obtain the top element from the heap and put the next one into the heap until all data are processed.

Lemma 1. *Given a set of N records, the time complexity of MDSort is $O(N \cdot \lg c)$, in which c is a constant representing the maximum disorder offset.*

Proof. Since the maximum disorder is a constant c, the correct data order can be found within the maximum offset c. Each time a new node is inserted, the time required to maintain the heap is $O(\log_2 c)$, in which c is the number of nodes in the heap. The number of adjustment



(a) Raw heap (b) Adjust the raw heap to the min-heap

(c) Get the top of the heap and insert the next data (d) Adjust the heap structure

Fig. 4. Maximum disorder sorting based on a heap.

operations is the number of all data points N, and therefore the time complexity of the algorithm is $O(N \cdot \lg c)$. □

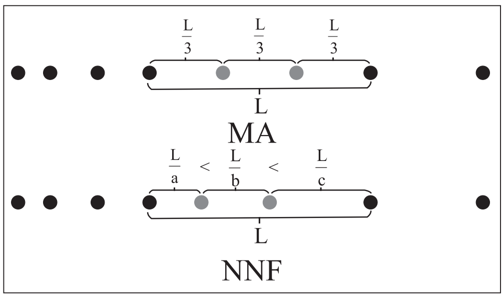
For ordered data, deleting duplicate data is simple. The algorithm determines whether the current detection record is the same as the next record. If they are equal, the duplicate data is deleted. If they are not equal, the algorithm continues to detect other types of abnormalities.

5 REPARING MISSING DATA

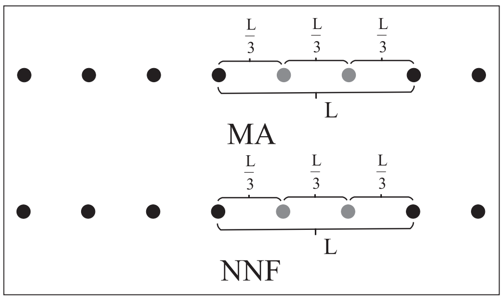
GPS devices may be accidentally shut down and therefore no data is collected afterward. For example, when the driver turns off the engine, no data is collected as the device is turned off. The vehicle may be parked for a long time, but the algorithm likely identifies the parking time as a large amount of missing data. To avoid misidentification, we partition the data whose time difference between the two consequent records is larger than 10 minutes. This method not only avoids loading a large amount of data into the memory, but also correctly handles misidentified data.

Since 100 consecutive data can fully reflect the frequency of dataset collection, before performing data missing detection on the dataset, we first count the time difference between the first 100 data in the dataset. The equipment that collects data usually has a fixed collection frequency. We take the time with the largest probability of different time differences among the first 100 data as the collection frequency of the current dataset. For a time-ordered dataset, we calculate the time difference between consecutive data records, and compare the time difference with the collection frequency to determine missing data. Two consecutive data with a time difference greater than twice the frequency or more are recorded as the leftmost and rightmost ends of the missing data.

We segment the data with large time differences in continuous data records, which effectively avoids the fact that the data is incorrectly recognized as a large amount of missing data due to engine shutdown. To achieve a good filling effect, we calculate the changes in normal data around the missing data. We use the change of normal data and the duration of missing data to solve the filling problem. This is done by the filling method based on neighbor data. To predict data points



(a) uniform acceleration



(b) uniform speed

Fig. 5. Comparison of different methods to repair missing data.

during this period, we approximately set this period of data as a uniform acceleration for analysis. According to the collection frequency of the original data obtained by the above statistics, we first calculate the number of missing data by Formula (1), where t_l is the leftmost time of the missing data, t_r is the rightmost time, and T_f is the frequency. Then, the approximate value of the speed before the missing data is obtained by the weighted average method of the normal data for a period of time before the missing data. The approximate speed of normal data for a period of time after missing data is also obtained by this method. According to the speed change in the neighboring data of the missing data, the running state of the car is vividly simulated. In the case of uniform acceleration, the acceleration of the car during the period of missing data can be calculated from the speed change value and the duration of the missing data. The driving distance within a fixed frequency can be calculated by Formula (2), where v_l is the approximate speed before the missing data, and v_r is the approximate speed after the missing data. We map the data of latitude and longitude to a two-dimensional plane through Gauss-Kruger, and (x_i, y_i) are the corresponding coordinates of the latitude and longitude mapping. The simultaneous equations in Formula (3) are obtained from the correct data connection on the two adjacent sides of the missing data and the left adjacent point as the center of the circle, and the travel distance as the radius. Through the simultaneous equations of Formula (3), the corresponding coordinate positions in Formula (4) can be obtained. The **NNF (Nearest Neighbor Filling)** algorithm is given in Algorithm 3. NNF first calculates the number of missing data points, then calculates the distance between the current position and the previous correct data by traversing the missing data, and finally calculates the coordinates of the current position and fills in the data.

$$num = \frac{t_r - t_l}{T_f} \tag{1}$$

$$L = v_l * T_f + \frac{\frac{v_r - v_l}{t_r - t_l} * T_f^2}{2} \tag{2}$$

$$\begin{cases} x = x_l + (y - y_l)(x_r - x_l)/(y_r - y_l) \\ (x - x_l)^2 + (y - y_l)^2 = L^2 \end{cases} \tag{3}$$

$$\begin{cases} (x_l + u(x_r - x_l)/(y_r - y_l), y_l + u), y_r > y_l \\ (x_l - u(x_r - x_l)/(y_r - y_l), y_l - u), y_r < y_l \\ (x_l + L, y_l), y_r = y_l \wedge x_l \le x_r \\ (x_l - L, y_l), y_r = y_l \wedge x_l > x_r \end{cases} \quad u = \left(\frac{L}{\sqrt{1 + [(x_r - x_l)/(y_r - y_l)]^2}} \right) \tag{4}$$

Figure 5(a) shows the results of different methods to fill in missing data under accelerated conditions. Based on the smoothed **moving average (MA)** model [7], the distance between normal data is evenly divided according to the number of missing points. MA method does not use the

ALGORITHM 3: *NNF*

```
Input: trajectory dataset $P$,
       left margin $l$,
       right margin $r$
Output: $P'$
1: $num \leftarrow (p_r.t - p_l.t)/T_f$
2: $v_l, v_r \leftarrow$ calculate the approximate speed on the left and right sides of the missing data
3: $L_f \leftarrow$ calculate the distance passed within the frequency according to Formula (2)
4: $L \leftarrow L_f$
5: while $num \neq 0$ do
6:     $p'.t \leftarrow$ calculate missing data time
7:     $(p'.x, p'.y) \leftarrow$ calculate the coordinates of filling data according to the position coordinates
       in Formula (4)
8:     $--num$
9:     $L \leftarrow$ calculate the distance passed in the next frequency
10:    $P' \leftarrow P.add(p')$
11: return $P'$
```

speed change of the neighboring correct data, and the change factors of different distances under different driving conditions. Therefore, in the case of accelerated driving, the distance allocated for data filling using MA is much larger than the distance allocated by NNF before the beginning of the missing data. At the same time, allocating the same driving distance also violates the law of driving, that is, in a straight line, the driving distance with higher speed should be longer than the driving distance with lower speed. The opposite of the acceleration situation is the deceleration situation, and the method of evenly distributing the distance also violates the law of driving. Figure 5(b) shows the results of different methods to fill in missing data at a constant speed. If there is no change in the adjacent speed of the missing data, the current missing data is missing when the car is traveling at approximately a constant speed. The method of smooth filling by MA can reflect the driving record very well. Since the car is at a constant speed, the acceleration is $0m/s^2$. At this time, the data filled by NNF also tends to be average, and the result of filling the data is similar to the MA method. NNF not only has the advantage of MA filling data equally, but also has a good filling advantage for gradually evolving data. In the changing data, NNF has a better filling effect than MA.

6 REPAIRING DRIFTING DATA

To detect and repair drifting data, we design an algorithm that combines range constraints and maximum likelihood estimation. Range constraint repairs large drifting anomalies to avoid using statistical methods to violate the minimum change principle. Repairing small anomalies by the maximum likelihood method also solves the problem of anomalies that are not detected by constraints. Since the vehicle cannot accelerate all the time, the current speed is used to set the acceleration change. The braking standards of the European Community and the United Nations Economic Commission for Europe are shown in Table 2.

M_1 refers to a passenger car with no more than 8 seats; M_2 refers to a passenger car with more than 8 seats and a total mass of not more than 5 tons; M_3 refers to a passenger car with more than 8 seats and a total mass of more than 5 tons; N_1 refers to a truck or tractor with a total mass of no more than 3.5 tons; N_2 refers to a truck with a total mass of $(3.5tons, 12tons]$; N_3 refers to a truck

Table 2. Braking Standard

Car type	M_1	M_2 M_3	N_1 N_2 N_3
Braking Deceleration (m/s^2)	5.8	5	4.4
Braking Time (s)	0.36	0.54	0.54
Braking Distance (m)	$\leq 0.1V + \frac{V^2}{150}$	$\leq 0.15V + \frac{V^2}{135}$	$\leq 0.15V + \frac{V^2}{115}$

Table 3. Raw Data Collected by Different Devices

	Id	Time	Longitude	Latitude	...	Direction	Speed
No. 1	1	2008-02-05 22:05:42	116.69155	39.85164			
	1	2008-02-05 22:15:41	116.69155	39.8518			
	1	...	...	...			
	1	2008-02-05 23:55:41	116.69159	39.85182			
No. 2	AA00001	2018/8/4 1:23:54	115.944571	28.65114	...	114	11
	AA00001	2018/8/4 1:23:55	115.944603	28.651126	...	110	12
	AA00001	...	...	...	...	...	...
	AA00001	2018/8/4 1:25:02	115.950913	28.648358	...	117	42

with a total mass of more than 12 tons. Our dataset is mainly collected by M_1 and N_2 types of vehicles. The speed change rate of taxis generally does not exceed 3.5 m/s^2, and the speed change rate of transport vehicles does not exceed 2.5 m/s^2. The reference thresholds for different data speed change rates are set by $[-5.8m/s^2, 3.5m/s^2]$ and $[-4.4m/s^2, 2.5m/s^2]$, respectively.

To detect and repair drifting anomalies, direction is an important factor for the algorithm. Different data collection sources make the original data have some subtle differences in attributes. Table 3 shows the raw data information collected by No.1 equipment and No.2 equipment. Data collected by No.1 device only includes *id*, *time*, *longitude*, and *latitude*. In addition to the above-mentioned four attributes, data collected by equipment No. 2 has additional attributes such as *direction* and *speed*. To make full use of the information, we perform the analysis on the direction to further improve the algorithm if the data contains *direction*.

6.1 Range Constraints and Statistics

For the data anomaly repairing without *direction*, we perform anomaly repairing through a combination of restricting data changes and maximum likelihood in a sliding window. The constraint-based method SCREEN [45] constrains the data through fixed changes (s_{min}, s_{max}), but the speed cannot always increase under normal circumstances, in which s_{min} is the maximum decrease change and s_{max} is the maximum increase change of the data. Constraint-based methods do not well detect small anomalies and do the repairing. Using the maximum likelihood method to count acceleration will make the entire algorithm run slowly. Using a local sliding window can improve the algorithm's efficiency. To reduce the task of modification, we design a range constraint method to repair large anomalies. For abnormal situations with small deviations, we design a method based on sliding window statistics to detect and repair data. The **RCSWS (Range Constraints and Sliding Window Statistics)** is given in Algorithm 4.

Based on speed constraints, range constraints can well implement the repairing of detected abnormalities. The data range refers to the circular area formed by the maximum distance passed by the current data point within the frequency. When detecting a data point, not only the range of the previous correct data point, but also the range of the next correct data point is required. To

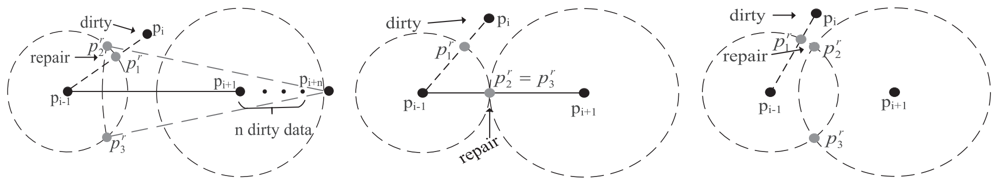
ALGORITHM 4: *RCSWS*

```
Input: trajectory dataset $P$,
       detection location $i$,
       frequency $T_f$,
       probability threshold $\sigma$
Output: $P'$
1:  $count \leftarrow 0$
2:  $a \leftarrow$ calculate the acceleration of $p_{i-1}$
3:  $t \leftarrow T_f$
4:  $dis_{max} \leftarrow$ calculate the distance of $p_{i-1}$ with acceleration $a$ in time $t$
5:  $dis \leftarrow Dist(p_{i-1}, p_i)$
6:  while $dis \geq dis_{max}$ do
7:      $count++$
8:      $t \leftarrow$ calculate the time between $p_{i-1}$ and $p_{i+count}$
9:      $dis_{max} \leftarrow$ calculate the distance of $p_{i-1}$ with acceleration $a$ in time $t$
10:     $dis \leftarrow Dist(p_{i-1}, p_{i+count})$
11: if $count > 1$ then
12:     $P' \leftarrow$ evenly divide the data changes from $p_{i-1}$ to $p_{i+count}$ to repair continuous anomalies
13:     return $P'$
14: if $count = 1$ then
15:     $P' \leftarrow$ repair $p_i$ through range constraints
16:     return $P'$
17: else
18:     $P \leftarrow$ count changes in data acceleration of sliding window
19:     $P(i) \leftarrow$ probability of speed change at $p_i$
20:     if $P(i) < \sigma$ then
21:         $P' \leftarrow$ use a change value with a probability greater than $\sigma$ to repair $p_i$
22:         return $P'$
23: return $P$
```

reflect the speed change between data, we assume that the speed of the latter data point is greater than the speed of the previous data point. Therefore, the range radius of the previous data point of the detection point is smaller than the range radius of the next data point. At the same time, the detection point should be within the intersection of the previous correct data range and the next correct data range. For data not in the intersection, we follow the minimum change principle to repair abnormal data. There are three cases when detecting the repairing situation:

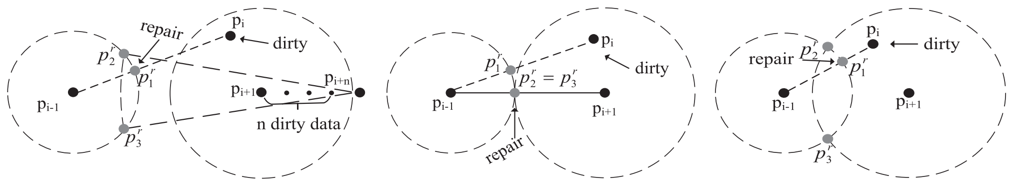
Case (1): The detected data point is not within the range of the surrounding data. In this case, data points deviate greatly from the original one. To fulfill the minimum change principle, we repair the abnormal data points by setting them the closest normal data range. If two data points are centers of the circles, the radius of the range depends on the distance of the data point within the sampling frequency. If there is no intersection between the two ranges, as shown in Figure 6(a), the next data is determined by evaluating whether the range constraint is violated.

If the range constraint is still violated, the determination is performed backward until the data meets the constraint condition. We connect the detection point p_i and the previous point p_{i-1}. The intersection of the line and the range of p_{i-1} is denoted as p_1^r. p_2^r and p_3^r are the intersecting points of two circles satisfying the constraint condition. The repairing p_i can be calculated using



(a) No intersection (b) One intersection point (c) Two intersection points

Fig. 6. The abnormal point violates the range constraint of surrounding data.



(a) No intersection (b) One intersection point (c) Two intersection points

Fig. 7. The abnormal point violates the range constraint of the previous data point.

Formula (5). If p_1^r is in the range at which two circles intersect, then p_1^r is used as the repair of p_i. In other cases, the point closest to p_i in p_2^r and p_3^r is used for repairing. The remaining abnormal points are detected and repaired according to newly repaired points. If two circles centered on two data points intersect, as shown in Figure 6(b), the intersecting point is the repairing point of the abnormal data point ($p_2^r = p_3^r$). If two circles centered on two data points have two intersecting points, as shown in Figure 6(c), according to the location of the abnormal data, the one in p_2^r and p_3^r that is closest to p_i is used for abnormal repairing.

$$p_i' = \begin{cases} p_1^r & if\ p_1^r in\ intersecting\ range \\ p_2^r & Dist(p_i, p_2^r) \leq Dist(p_i, p_3^r) \\ p_3^r & Dist(p_i, p_2^r) > Dist(p_i, p_3^r) \end{cases} \tag{5}$$

Case (2): For the case where the detected data point is only within the normal range of one data, p_{i-1} is repaired and p_i violates the range constraint of p_{i-1}. The current detection point p_i exceeds the constraint range of p_{i-1}, but is within the constraint range of p_{i+1}, as shown in Figure 7. If p_{i-1} and p_{i+1} also violate the constraints, we continue to set the subsequent data until the two range constraints intersect. As case (1), we obtain three candidate repair points and repair p_i according to Formula (5), as shown in Figure 7.

Case (3): The case when the detected data point p_i is within the range of the two data. Since the range constraints are satisfied, constraint-based methods cannot detect and repair abnormal data. To accurately detect and repair anomalies in a small range, we employ the sliding window maximum likelihood estimation. Compared with the overall data statistics, sliding window statistics not only reduce the amount of statistical data, but also avoid the interference of long intervals and many incorrect data. Since the first half of the data in the window has been repaired, the statistical results can correctly reflect the data changes in the window. If the current data change occupies a small probability in the statistical results, we treat the current data as an abnormal point with a small deviation from the trajectory. Since half of the data is already corrected, based on the minimum change principle, we use the probability changes 50% in statistical results to repair abnormalities.

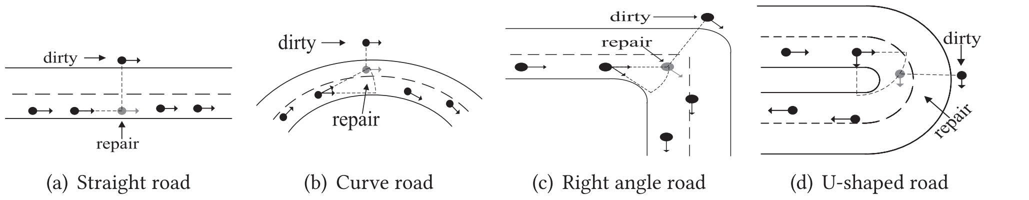


Fig. 8. Repair anomalies corresponding to different driving directions.

When detecting small abnormalities, existing statistical methods will count the entire dataset, making the algorithm running slowly. If there are a lot of abnormal data, the statistical results cannot reflect the true data phenomenon and lead to bad abnormal repairing results.

6.2 Direction Analysis

For the data with *direction*, the changes in the speed and direction of previous and following data are used to accurately detect and repair abnormal data. The analysis of driving direction in anomaly detection and repairing is divided into three parts.

Case (1): The driving direction of data in previous and following segments in the window is almost similar. Such trajectory data is usually collected when the vehicle is moving along a straight road, as shown in Figure 8(a). In the case of straight driving, the driving direction is almost the same. When the driving direction of the current detection point is merged with the previous correct data point, the two directions coincide. According to the approximate speed and acceleration of the previous point, the distance passed in the frequency can be obtained. The end point of the distance is used as the repairing point of the drifting data when driving in a straight line.

Case (2): There is a certain deflection in the driving direction of the data in previous and following sections in the window. This type of trajectory data is usually caused by a relatively small curve, as shown in Figure 8(b) and 8(c). We merge the driving direction of the current point with the previous data point, and there will be a certain deviation between the two directions. We use the approximate speed and acceleration of the previous data to calculate the distance passed in these two directions within the frequency. The fan-shaped area formed by the two directions is the area where the current detection point is located. To follow the minimum change principle [5], we take the point closest to the abnormal point in the fan-shaped area as the data repairing point. If the speed of the car is slow on the curve and the direction of the speed change in a short time is relatively small, this situation is transformed into case (1).

Case (3): The driving direction of the data in previous and following sections in the window has a deflection close to 180 degrees. For this type of trajectory, the data is usually collected on a U-shaped road, as shown in Figure 8(d). In this case, we also merge the direction of the current detection point with the direction of the previous data point. According to the formed fan-shaped area, the data closest to the abnormal point is selected as the repairing point. In the case of slow speed, the data is analyzed according to the specific observation data and converted into cases (1) or (2).

In repairing abnormal data with directions, we analyze the range of repairing points in different road scenes. In different scenarios, the fan-shaped area formed by the driving distance and the angle between the two directions can always contain the correct data points. Compared with the entire circular area, the fan-shaped area has a smaller range of selectable repairing points. In data repairing, the use of fan-shaped areas can not only satisfy the minimum change principle, but also reduce the scope of the determined data and improve the accuracy of data repairing. The algorithm named **RCSWSD (Range Constraints and Sliding Window Statistics with Direction)** is introduced in Algorithm 5.

Table 4. Description on Datasets

Type	#Attributes	Time Range	Longitude Range	Latitude Range	#Vehicles	#Records	City	Frequency
taxi	4	[2008-02-02, 2008-02-08]	(115.117, 117.067)	(39.067, 41.1)	10357	15 million	Beijing, China	10 s
truck	13	[2018-07-30, 2018-10-26]	(109.418705, 121.99981)	(22.100031, 31.9953)	450	45 million	Nanchang, China	1 s

ALGORITHM 5: *RCSWSD*

```
Input: trajectory dataset $P$,
       detection location $i$,
       frequency $T_f$,
       probability threshold $\sigma$,
Output: $P'$
1:  $P' \leftarrow RCSWS(P, i, T_f, \sigma)$
2:  $a_{i-1} \leftarrow$ calculate the acceleration of $p_{i-1}$
3:  $v_{i-1} \leftarrow$ calculate the speed of $p_{i-1}$
4:  $L \leftarrow$ calculate the passing distance of $p_{i-1}$ within the frequency $T_f$
5:  $\alpha \leftarrow$ calculate the angle between $p_{i-1}$ and $p_i$ driving direction
6:  $A \leftarrow$form the fan-shaped area and construct an equation
7:  if $p_i \notin A$ then
8:      $p_i' \leftarrow$ calculate the point closest to $p_i$ in the fan-shaped area
9:      $P' \leftarrow p_i'$
10: return $P'$
```

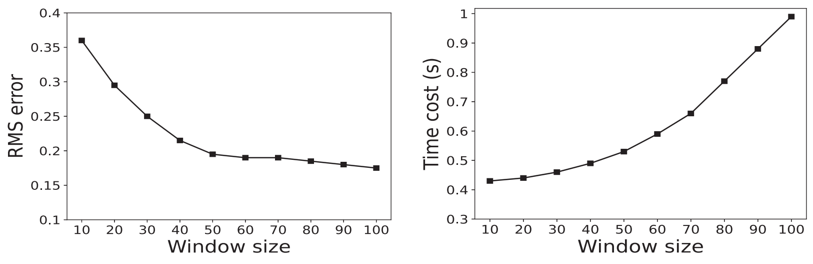
7 EXPERIMENTAL EVALUATION

The evaluation is conducted in a desktop PC (Intel(R) Core(TM) i7-4790CPU, 3.6 GHz, 8 GB memory, 1 TB disk) running Ubuntu 14.04 (64 bits, kernel version 4.8.2-19). Using C/C++, we develop the framework in an extensible database system SECONDO [26].

7.1 Settings

7.1.1 Datasets. We collect GPS data from two different vehicles. Some information about these two datasets is given in Table 4. One is the taxi dataset collected by Microsoft [52, 53], including *id*, *time*, *longitude*, and *latitude*. This dataset contains the GPS trajectories of 10,357 taxis in Beijing from Feb. 2 to Feb. 8, 2008. The total number of points in this dataset is about 15 million, and the total distance of the trajectories reaches 9 million kilometers. Another dataset is the trajectory data of transport vehicles (truck) provided by the Ministry of Transport of China [1], which contains 13 attributes such as *device_num*, *direction_angle*, *lng*, *lat*, *acc_state*, *location_time*, *gps_speed*, and other attributes. This dataset contains the trajectory data of a total of 450 transport vehicles from July 30, 2018 to October 26, 2018, with approximately 45 million data points, and the sampling frequency is 1 second.

7.1.2 Evaluation Criteria. **Root-Mean-Square error (RMS)** [29] is employed to evaluate the accuracy of data repairing. RMS represents the distance between the repaired data and the real data. The smaller the RMS, the closer the repaired data is to the real data, and the more accurate the results of data repairing. Let x^{truth} be the ground truth of clean GPS sequence, and x^{repair} be the repaired GPS sequence. The RMS error can be calculated by Formula (6). Since there is no groundtruth trajectory dataset, the groundtruth trajectory dataset used in the experiment is



(a) RMS (b) Time cost

Fig. 9. RMS and time consumption of different window sizes on a dataset size of 100k.

referenced by selecting approximate points in the trajectory. For the dataset without anomalies, we manually remove the normal data and add some noise data accordingly.

$$\Delta\left(x^{truth}, x^{repair}\right) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(x_i^{truth} - x_i^{repair}\right)^2} \tag{6}$$

7.1.3 Effect of Window Size. The size of the sliding window plays a quite important role in the detection and repairing of data with small abnormal values in the original data. To obtain an appropriate window size, We use 100k real datasets to test RCSWS. Experiment with different window sizes and compare the RMS error results of data repairing with the corresponding time consumption to balance cleaning accuracy and efficiency. As the window size increases, the value of RMS error also gradually decreases and shows a slower and slower trend, as shown in Figure 9(a). In terms of the time consumption of the algorithm, as the window size increases, the corresponding algorithm time becomes larger and larger, as shown in Figure 9(b). When the window size is greater than 50, the RMS error value does not decrease significantly but occasionally increases. This phenomenon is since the window setting is too large, which may cause the statistical results to fail to reflect the data status near the detection point. The statistical results of a large amount of data are also easy to make the data repairing tend to be consistent, thereby forming inaccurate data repairing. To obtain a balance between the accuracy and efficiency of data repairing and obtain a high accuracy rate in a short time, we set the window size to 50 and use this value in subsequent comparison experiments.

7.2 Evaluation

We compare the time consumption of MDSort and ordinary heap sort in terms of maximum out-of-order offset data. We compare our proposed algorithms RCSWS and RCSWSD with two state-of-the-art methods including: (1) EWMA [19], and (2) SCREEN [45]. EWMA sets the weighting coefficient of each observation to a value that decreases exponentially with time. The closer the observation time is to the current moment, the larger the numerical weighting coefficient. EWMA determines the predicted value according to the final moving average to repair the data. SCREEN uses the maximum and minimum constraints of the previous data point to find the current data range. The candidate points are obtained through the constraints of the following data points on the current point. Then, SCREEN compares the median of the candidate points with the current data range to repair the data.

7.2.1 Comparison Algorithms Adjustment. To obtain the optimal experimental results of EWMA and SCREEN methods on the taxi dataset, we use real data for experiments and adjust the parameters of the two methods, respectively. In the adjustment of EWMA parameters, we experiment

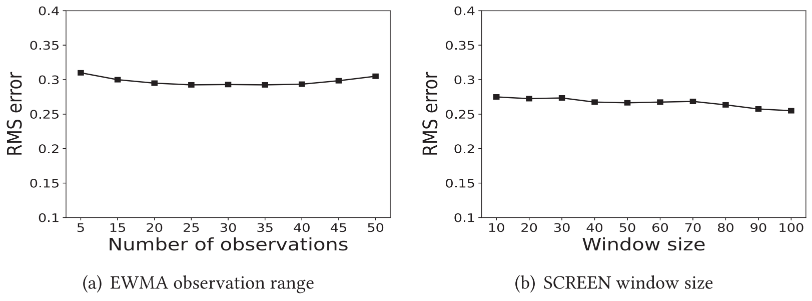


Fig. 10. Algorithms adjustment.

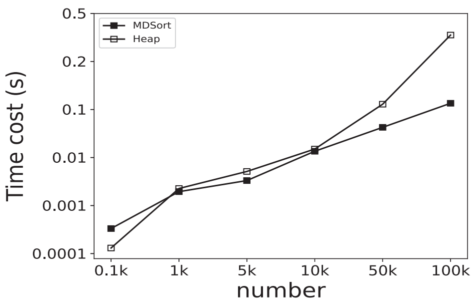
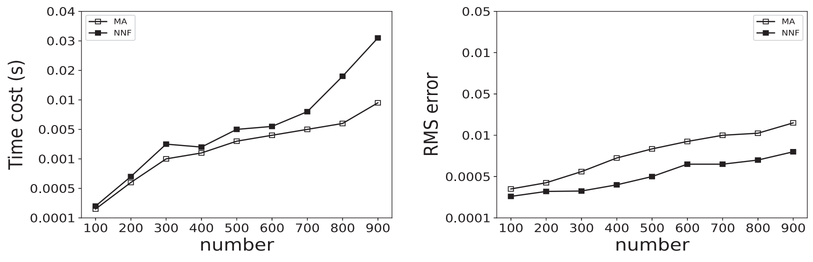


Fig. 11. MDSort and ordinary Heap sort time comparison.

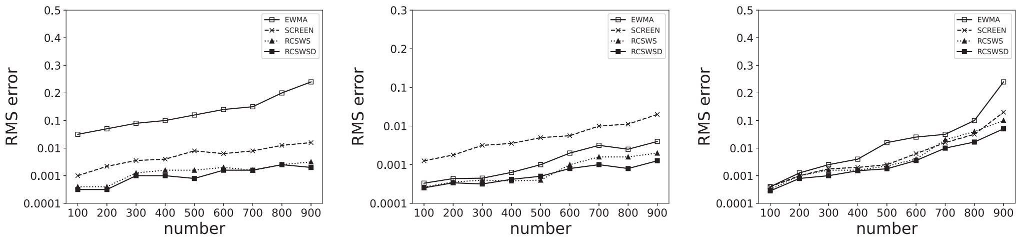
with the selection of the observation data range, as shown in Figure 10(a). The observation data of EWMA show the best repairing result within 40 seconds. As the amount of observation data increases, the data repairing will tend to be smooth so that the correct data is also modified. For the SCREEN setting, we use the local optimal algorithm *Local*. We set the maximum speed of a car as $120km/h \approx 33.33m/s$, which means the maximum distance in 1 second is 33.33 meters, and the corresponding longitude change is 0.0003902 latitude change is 0.0002998, hence constraints in *Local* are $s_{lon} = (-0.0003902, 0.0003902)$ and $s_{lat} = (-0.0002998, 0.0002998)$. The experimental result of SCREEN window size adjustment is shown in Figure 10(b). As the window size increases, the local optimal repairing results of SCREEN also slightly improve. The increase of the window will cause SCREEN's local optimal algorithm to become a global optimal algorithm, which will waste time. To obtain a good local optimal algorithm, we use the experimental value of 60 as the window size of SCREEN's next experiment.

7.2.2 Sorting Time. We use real datasets to conduct experiments on MDSort with maximum disorder offset and ordinary heap sort. After analyzing the data, since the maximum delay of data collection is 1 min and the frequency of data collection is 1 s, our maximum offset in MDSort is set to 60. The time experiment results of MDSort and ordinary heap sort are shown in Figure 11. When the size of dataset is 100, the original data basically maintains the correct time series due to the small amount of data, and the redundant operations of MDSort cause some waste of time, making the time consumption of Heap slightly less than that of MDSort. Since the dataset increases, the advantages of MDSort over Heap become more and more obvious. When the dataset reaches 100k, the efficiency of MDSort is three times faster than Heap. The larger the dataset, the more out-of-order data in the original data. Since the cost of adjusting the heap in the large-scale data of MDSort is much smaller than that of Heap, the efficiency improvement is very considerable.



(a) Time cost (b) RMS

Fig. 12. Filling algorithm comparison.

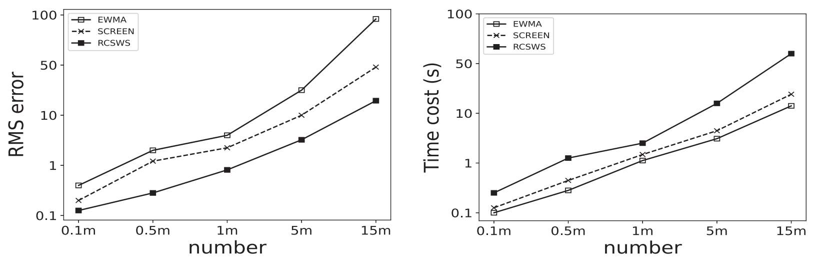


(a) Large anomaly datasets (b) Small anomaly datasets (c) Various anomaly datasets

Fig. 13. The impact of different types of abnormalities.

7.2.3 Data Filling. We remove some correct data points from the real dataset containing 1,000 data points for data filling comparison. Since MA method fills in the missing data by calculating the average value of the surrounding data, the efficiency of the algorithm is higher than that of the NNF, as shown in Figure 12(a). However, in terms of the accuracy of data filling, NNF uses the change relationship of the neighboring data to calculate the approximate location of the missing point based on the change of the speed around the missing data. Compared with the filling method of MA, the method of NNF is more in line with the operation law of driving, such that NNF is better than MA in accuracy, as shown in Figure 12(b).

7.2.4 Effect of Different Abnormalities. We manually add three different types of abnormal datasets to verify the repair effect of the algorithm. The three datasets contain many large anomalies, many small anomalies, and various types of anomalies, respectively. Figure 13(a) shows the experimental results with large abnormal data. According to the value of RMS error, we can intuitively determine that RCSWS, RCSWSD, and SCREEN are better than EWMA in repairing large abnormal data. Since small anomalies are difficult to repair with constraint-based methods, we add statistics to RCSWS to avoid the disadvantages of repairing small anomalies. The data before the detection point tends to be stable so that the predicted value obtained by the sum of the products of different weights can still be used to repair small anomalies. Therefore, EWMA is better than SCREEN in repairing minor anomalies, and the results of RCSWS and RCSWSD repair are not much different, as shown in Figure 13(b). For the repair of many different types of anomalies, due to the interference of some invalid points, such as a value of 0, the smooth EWMA over-repairing causes the correct data to be modified to show the bad repair results, as shown in Figure 13(c). Since RCSWSD uses direction analysis to repair special conditions in a variety of different abnormalities, the repair result is better than RCSWS.



(a) RMS (b) Time cost

Fig. 14. Datasets of different sizes without direction (taxi).

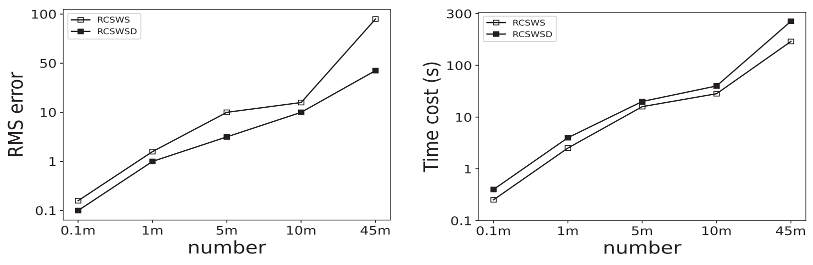
7.2.5 Effect of Dataset Size. We first use taxi datasets of different sizes for comparative experiments. In the case of a small amount of data, the RMS error obtained by the three algorithms has little difference, as shown in Figure 14(a). When the amount of data reaches 0.1m, the RMS error value of EWMA and SCREEN increases slightly. The RMS error fluctuation of RCSWS is small and still has a good data repairing effect. When the amount of data reaches 15m, the shortcoming that SCREEN cannot handle anomalies of meeting the constraints is magnified. EWMA handles some large abnormal data too smoothly so that the correct data in the data is also modified, resulting in a larger RMS error value than SCREEN. RCSWS not only uses the range constraint method to solve the big anomaly, but also repairs the small anomaly that satisfies the constraint through the statistical method so that the overall repair accuracy is better than the other two methods. Since RCSWS needs to perform constraint detection on the original data and perform data change statistics on the data that meets the constraints, RCSWS takes the most time to repair large-scale data, as shown in Figure 14(b). SCREEN needs to count the candidate points of the following data corresponding to the current point, while EWMA only needs the sum of the products of the previous data corresponding to different weights, hence EWMA takes the least time.

7.2.6 Effect of Different Attributes. Different data attributes in the original data also affect the cleaning effect of the algorithm. We experiment with the RCSWS and RCSWSD algorithms using the data of the transport vehicle, and the RMS of the experiment is shown in Figure 15(a). Since the dataset of transport vehicles contains the driving direction of the vehicle, RCSWSD can make good use of this attribute. When the amount of data is small, there is not much difference between the two algorithms. When the amount of data reaches 45m, the RMS of the RCSWSD algorithm is significantly lower than RCSWS. Since RCSWSD increases the directional analysis of data, the algorithm time consumption is higher than RCSWS, as shown in Figure 15(b). With a 45m dataset, the time consumption of RCSWSD is almost double that of RCSWS.

RCSWS not only has the advantages of SCREEN's detection and repairing of larger abnormal data, but also uses statistical methods to make up for SCREEN's repairing of abnormalities within constraints. The sliding window approach not only improves the operating efficiency of the algorithm but also avoids the single direction of data repairing due to the statistical changes in the overall data. At the same time, the characteristics of the current detection data can be obtained by counting the changes of the neighboring data to implement accurate data cleaning.

8 CONCLUSION

This paper studies anomaly detection and repairing of GPS data. Different types of abnormal data are thoroughly analyzed, and detection and repairing methods are proposed. We evaluate our method *GPSClean* by comparing with two data cleaning algorithms EWMA and SCREEN.



(a) RMS (b) Time cost

Fig. 15. Datasets of different sizes with direction (truck).

Experimental results demonstrate that our method is comprehensive in detecting abnormal types. In terms of the accuracy of anomaly repairing, our method is significantly better than EWMA and SCREEN. By performing the evaluation in different settings, we find that the window size has significant effect on algorithm efficiency. The future work is to design a model that evaluates GPS data quality.

REFERENCES

[1] 2019. http://www.tipdm.org/bdrace/tzjingsai/20181226/1544.html#sHref.

[2] Gérard Alengrin and Gérard Favier. 1978. New stochastic realization algorithms for identification of ARMA models. In *IEEE ICASSP*. 208–213.

[3] Asif Iqbal Baba, Manfred Jaeger, Hua Lu, Torben Bach Pedersen, Wei-Shinn Ku, and Xike Xie. 2016. Learning-based cleansing for indoor RFID data. In *SIGMOD*. 925–936.

[4] Sabyasachi Basu and Martin Meckesheimer. 2007. Automatic outlier detection for time series: An application to sensor data. *Knowl. Inf. Syst.* 11, 2 (2007), 137–154.

[5] Philip Bohannon, Michael Flaster, Wenfei Fan, and Rajeev Rastogi. 2005. A cost-based model and effective heuristic for repairing constraints by value modification. In *ACM SIGMOD*. 143–154.

[6] George E. P. Box and David A. Pierce. 1970. Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *Journal of the American Statistical Association* 65, 332 (1970), 1509–1526.

[7] David R. Brillinger. 2001. Time series - data analysis and theory. Classics in applied mathematics, Vol. 36. SIAM.

[8] Peter J. Brockwell, Richard A. Davis, and Matthew V. Calder. 2002. *Introduction to Time Series and Forecasting*. Vol. 2.

[9] Chao Chen, Shuhai Jiao, Shu Zhang, Weichen Liu, Liang Feng, and Yasha Wang. 2018. TripImputor: Real-time imputing taxi trip purpose leveraging multi-sourced urban data. *IEEE Trans. Intell. Transp. Syst.* 19, 10 (2018), 3292–3304.

[10] Chao Chen, Daqing Zhang, Xiaojuan Ma, Bin Guo, Leye Wang, Yasha Wang, and Edwin Hsing-Mean Sha. 2017. Crowd-deliver: Planning city-wide package delivery paths leveraging the crowd of taxis. *IEEE Trans. Intell. Transp. Syst.* 18, 6 (2017), 1478–1496.

[11] Longbiao Chen, Daqing Zhang, Gang Pan, Xiaojuan Ma, Dingqi Yang, Kostadin Kushlev, Wangsheng Zhang, and Shijian Li. 2015. Bike sharing station placement leveraging heterogeneous urban open data. In *ACM*. 571–575.

[12] Longbiao Chen, Daqing Zhang, Leye Wang, Dingqi Yang, Xiaojuan Ma, Shijian Li, Zhaohui Wu, Gang Pan, Thi Mai Trang Nguyen, and Jérémie Jakubowicz. 2016. Dynamic cluster-based over-demand prediction in bike sharing systems. In *ACM*. 841–852.

[13] Roberto Corizzo, Michelangelo Ceci, and Nathalie Japkowicz. 2019. Anomaly detection and repair for accurate predictions in geo-distributed big data. *Big Data Res.* 16 (2019), 18–35.

[14] Yinglong Diao, Ke-yan Liu, Xiaoli Meng, Xueshun Ye, and Kaiyuan He. 2015. A big data online cleaning algorithm based on dynamic outlier detection. In *CyberC*. 230–234.

[15] Jirun Dong and Richard Hull. 1982. Applying approximate order dependency to reduce indexing space. In *ACM SIGMOD*. 119–127.

[16] Uwe Draisbach, Felix Naumann, Sascha Szott, and Oliver Wonneberg. 2012. Adaptive windows for duplicate detection. In *IEEE ICDE*. 1073–1083.

[17] Chenguang Fang, Shaoxu Song, Zhiwei Chen, and Acan Gui. 2019. Fine-grained fuel consumption prediction. In *ACM CIKM*. 2783–2791.

[18] Stefan Funke and Sabine Storandt. 2015. Personalized route planning in road networks. In *SIGSPATIAL*. 45:1–45:10.

[19] Everette S. Gardner Jr. 2006. Exponential smoothing: The state of the art–Part II. *International Journal of Forecasting* 22, 4 (2006), 637–666.

[20] Felix A. Gers, Jürgen Schmidhuber, and Fred A. Cummins. 2000. Learning to forget: Continual prediction with LSTM. *Neural Comput.* 12, 10 (2000), 2451–2471.

[21] Tomasz Gogacz and Szymon Torunczyk. 2017. Entropy bounds for conjunctive queries with functional dependencies. In *ICDT*, Vol. 68. 15:1–15:17.

[22] Lukasz Golab, Howard J. Karloff, Flip Korn, Avishek Saha, and Divesh Srivastava. 2009. Sequential dependencies. *Proc. VLDB Endow.* 2, 1 (2009), 574–585.

[23] Bin Guo, Yan Liu, Wenle Wu, Zhiwen Yu, and Qi Han. 2017. ActiveCrowd: A framework for optimized multitask allocation in mobile crowdsensing systems. *IEEE Trans. Hum. Mach. Syst.* 47, 3 (2017), 392–403.

[24] Aditya Gupta and Bhuwan Dhingra. 2012. Stock market prediction using hidden Markov models. IEEE, 1–4.

[25] Manish Gupta, Jing Gao, Charu C. Aggarwal, and Jiawei Han. 2014. *Outlier Detection for Temporal Data.*

[26] Ralf Hartmut Güting, Thomas Behr, and Christian Düntgen. 2010. SECONDO: A platform for moving objects database research and for publishing and integrating research implementations. *IEEE Data Eng. Bull.* 33, 2 (2010), 56–63.

[27] David J. Hill and Barbara S. Minsker. 2010. Anomaly detection in streaming environmental sensor data: A data-driven modeling approach. *Environ. Model. Softw.* 25, 9 (2010), 1014–1022.

[28] Koji Ichikawa and Hiroshi Tamano. 2020. Unsupervised qualitative scoring for binary item features. *Data Sci. Eng.* 5, 3 (2020), 317–330.

[29] Shawn R. Jeffery, Minos N. Garofalakis, and Michael J. Franklin. 2006. Adaptive cleaning for RFID data streams. In *VLDB*. 163–174.

[30] Hoyoung Jeung, Hua Lu, Saket Sathe, and Man Lung Yiu. 2014. Managing evolving uncertainty in trajectory databases. *IEEE Trans. Knowl. Data Eng.* 26, 7 (2014), 1692–1705.

[31] Eamonn J. Keogh, Jessica Lin, Sang-Hee Lee, and Helga Van Herle. 2007. Finding the most unusual time series subsequence: Algorithms and applications. *Knowl. Inf. Syst.* 11, 1 (2007), 1–27.

[32] Dan Li, Dacheng Chen, Baihong Jin, Lei Shi, Jonathan Goh, and See-Kiong Ng. 2019. MAD-GAN: Multivariate anomaly detection for time series data with generative adversarial networks. In *ICANN*, Vol. 11730. 703–716.

[33] L. Li, X. Chen, Q. Liu, and Z. Bao. 2020. A data-driven approach for GPS trajectory data cleaning. In *DASFAA (Lecture Notes in Computer Science, Vol. 12112)*. Springer, 3–19.

[34] Andrei Lopatenko and Loreto Bravo. 2007. Efficient approximation algorithms for repairing inconsistent databases. In *ICDE*. 216–225.

[35] Chunyang Ma, Hua Lu, Lidan Shou, and Gang Chen. 2013. KSQ: Top-(k) similarity query on uncertain trajectories. *IEEE Trans. Knowl. Data Eng.* 25, 9 (2013), 2049–2062.

[36] Martyna Marczak, Tommaso Proietti, and Stefano Grassi. 2018. A data-cleaning augmented Kalman filter for robust estimation of state space models. *Econometrics and Statistics* 5 (2018), 107–123.

[37] Dominik Mautz, Claudia Plant, and Christian Böhm. 2020. DeepECT: The deep embedded cluster tree. *Data Sci. Eng.* 5, 4 (2020), 419–432.

[38] Mostafa Milani, Zheng Zheng, and Fei Chiang. 2019. CurrentClean: Spatio-Temporal cleaning of stale data. In *IEEE ICDE*. 172–183.

[39] Yi Ouyang, Bin Guo, Xinjiang Lu, Qi Han, Tong Guo, and Zhiwen Yu. 2019. CompetitiveBike: Competitive analysis and popularity prediction of bike-sharing apps using multi-source data. *IEEE Trans. Mob. Comput.* 18, 8 (2019), 1760–1773.

[40] Channamma Patil and Ishwar Baidari. 2019. Estimating the optimal number of clusters k in a dataset using data depth. *Data Sci. Eng.* 4, 2 (2019), 132–140.

[41] Z. Qu, Y. Wang, Chong Wang, Nan Qu, and Jia Yan. 2016. A data cleaning model for electric power big data based on spark framework. *International Journal of Database Theory and Application* 9, 3 (2016), 137–150.

[42] Jingbo Shang, Yu Zheng, Wenzhu Tong, Eric Chang, and Yong Yu. 2014. Inferring gas consumption and pollution emission of vehicles throughout a city. In *ACM SIGKDD*. 1027–1036.

[43] Shaoxu Song, Chunping Li, and Xiaoquan Zhang. 2015. Turn waste into wealth: On simultaneous clustering and cleaning over dirty data. In *ACM SIGKDD*. 1115–1124.

[44] Shaoxu Song, Yu Sun, Aoqian Zhang, Lei Chen, and Jianmin Wang. 2020. Enriching data imputation under similarity rule constraints. *IEEE Trans. Knowl. Data Eng.* 32, 2 (2020), 275–287.

[45] Shaoxu Song, Aoqian Zhang, Jianmin Wang, and Philip S. Yu. 2015. SCREEN: Stream data cleaning under speed constraints. In *ACM SIGMOD*. 827–841.

[46] Xuan Song, Quanshi Zhang, Yoshihide Sekimoto, Ryosuke Shibasaki, Nicholas Jing Yuan, and Xing Xie. 2017. Prediction and simulation of human mobility following natural disasters. *ACM Trans. Intell. Syst. Technol.* 8, 2 (2017), 29:1–29:23.

[47] Yuqiang Sun, Lei Peng, Huiyun Li, and Min Sun. 2018. Exploration on spatiotemporal data repairing of parking lots based on recurrent GANs. In *ITSC*. 467–472.

[48] Dong Wang, Lance M. Kaplan, and Tarek F. Abdelzaher. 2014. Maximum likelihood analysis of conflicting observations in social sensing. *ACM Trans. Sens. Networks* 10, 2 (2014), 30:1–30:27.

[49] Mohamed Yakout, Laure Berti-Équille, and Ahmed K. Elmagarmid. 2013. Don't be SCAREd: Use scalable automatic repairing with maximal likelihood and bounded changes. In *ACM SIGMOD*. 553–564.

[50] Kenji Yamanishi and Jun'ichi Takeuchi. 2002. A unifying framework for detecting outliers and change points from non-stationary time series data. In *ACM SIGKDD*. 676–681.

[51] Wei Yin, Tianbai Yue, Hongzhi Wang, Yanhao Huang, and Yaping Li. 2018. Time series cleaning under variance constraints. In *DASFAA*, Vol. 10829. 108–113.

[52] Jing Yuan, Yu Zheng, Xing Xie, and Guangzhong Sun. 2011. Driving with knowledge from the physical world. In *ACM SIGKDD*. 316–324.

[53] Jing Yuan, Yu Zheng, Chengyang Zhang, Wenlei Xie, Xing Xie, Guangzhong Sun, and Yan Huang. 2010. T-drive: Driving directions based on taxi trajectories. In *ACM SIGSPATIAL ACM-GIS*. 99–108.

[54] Aoqian Zhang, Shaoxu Song, and Jianmin Wang. 2016. Sequential data cleaning: A statistical approach. In *SIGMOD*. 909–924.

[55] Qi Zhang, Jianlong Chang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. 2020. Spatio-Temporal graph structure learning for traffic forecasting. In *AAAI*. 1177–1185.

[56] Yu Zheng and Xing Xie. 2011. Learning travel recommendations from user-generated GPS traces. *ACM Trans. Intell. Syst. Technol.* 2, 1 (2011), 2:1–2:29.

[57] Yu Zheng, Xiuwen Yi, Ming Li, Ruiyuan Li, Zhangqing Shan, Eric Chang, and Tianrui Li. 2015. Forecasting fine-grained air quality based on big data. In *ACM SIGKDD*. 2267–2276.

[58] Yongzhen Zhuang, Lei Chen, Xiaoyang Sean Wang, and Jie Lian. 2007. A weighted moving average-based approach for cleaning sensor data. In *IEEE ICDCS*. 38.

Received January 2021; revised April 2021; accepted May 2021