Stream Data Cleaning under Speed and Acceleration Constraints

SHAOXU SONG and FEI GAO, Tsinghua University, China
AOQIAN ZHANG, Beijing Institute of Technology, China
JIANMIN WANG, Tsinghua University, China
PHILIP S. YU, University of Illinois at Chicago, USA

10

Stream data are often dirty, for example, owing to unreliable sensor reading or erroneous extraction of stock prices. Most stream data cleaning approaches employ a smoothing filter, which may seriously alter the data without preserving the original information. We argue that the cleaning should avoid changing those originally correct/clean data, a.k.a. the *minimum modification rule* in data cleaning. To capture the knowledge about *what is clean*, we consider the (widely existing) constraints on the speed and acceleration of data changes, such as fuel consumption per hour, daily limit of stock prices, or the top speed and acceleration of a car. Guided by these semantic constraints, in this article, we propose the constraint-based approach for cleaning stream data. It is notable that existing data repair techniques clean (a sequence of) data *as a whole* and fail to support stream computation. To this end, we have to relax the global optimum over the entire sequence to the local optimum in a window. Rather than the commonly observed NP-hardness of general data repairing problems, our major contributions include (1) polynomial time algorithm for global optimum, (2) linear time algorithm towards local optimum under an efficient *median-based solution*, and (3) experiments on real datasets demonstrate that our method can show significantly lower L1 error than the existing approaches such as smoother.

CCS Concepts: • **Information systems → Data cleaning;**

Additional Key Words and Phrases: Data repairing, speed constraints, acceleration constraints

ACM Reference format:
Shaoxu Song, Fei Gao, Aoqian Zhang, Jianmin Wang, and Philip S. Yu. 2021. Stream Data Cleaning under Speed and Acceleration Constraints. *ACM Trans. Database Syst.* 46, 3, Article 10 (September 2021), 44 pages.
https://doi.org/10.1145/3465740

This work is supported in part by the National Key Research and Development Plan (2019YFB1705301, 2019YFB1707001), the National Natural Science Foundation of China (62072265, 71690231), the MIIT High Quality Development Program 2020, NSF under grants III-1763325, III-1909323, and SaTC-1930941.
Authors' addresses: S. Song, F. Gao, and J. Wang, Tsinghua University, Beijing Key Laboratory for Industrial Bigdata System and Application, School of Software, Tsinghua University, Beijing, China; emails: sxsong@tsinghua.edu.cn, gaof16@mails.tsinghua.edu.cn, jimwang@tsinghua.edu.cn; A. Zhang, Beijing Institute of Technology, Beijing, China; email: aoqian.zhang@uwaterloo.ca; P. S. Yu, University of Illinois at Chicago, Chicago, USA; email: psyu@uic.edu.

0362-5915/2021/09-ART10 $15.00
https://doi.org/10.1145/3465740

1 INTRODUCTION

Dirty values commonly exist in data streams, for example, in traditional sensor data, due to the unreliable readers [19]. A large amount of inconsistent data is surprisingly observed in the domains of stock and flight [22]. According to the study, the accuracy of stock in *Yahoo! Finance* is 0.93, and the flight data accuracy in *Travelocity* is 0.95. Reasons for imprecise values include ambiguity in information extraction, unit error or pure mistake. For instance, the price of SALVEPAR (SY) is misused as the price of SYBASE, which is denoted by SY as well in some sources. (See more examples of data errors in the following.) Such inaccurate values, e.g., taken as the 52-week low price, may seriously mislead business investment.

A temporal smoothing filter, such as **exponentially weighted moving average (EWMA)** [14], may modify almost all the data values, most of which are originally correct/clean. It thus seriously damages the precision of individual data points (such as daily stock prices). Indeed, to preserve the original clean information as much as possible, the *minimum modification rule* is widely considered in improving data quality [5].

To capture the knowledge about *what is clean*, we notice that the "jump" of values in a stream is often constrained, so-called *speed constraints* and *acceleration constraints*. For the example of speed constraints, in financial and commodity markets, prices are only permitted to rise or fall by a certain number of ticks per trading session. In environment monitoring, temperature difference of any two days in a week should not be greater than 20 degrees. The fuel consumption of a crane should not be negative and not exceed 40 liters per hour. Moreover, for the example of acceleration constraints, we consider the trajectory of a van. The speed constraints state that the GPS value change of two points should not exceed 100 km/h, while the acceleration constraints further require that the difference on speeds between two consecutive points in a second is no greater than 10 km/h. That is, the increase/decrease of speeds in a second cannot be greater than 10 km/h. We believe that with these meaningful constraints on value change speed and acceleration, the cleaning could be more accurate.

Example 1.1. Consider the prices of a stock in 32 trading days, in Figure 1. As illustrated, large spikes appear in the dirty data (in black), e.g., in day 15, owing to ambiguity in information extraction as discussed or pure mistake. It may also be raised by temporary loss of data (days 23 to 26) and the subsequent coding of these missing values as zero by the data collection system.

The smoother method (in orange) modifies almost all the price values, most of which are indeed accurate. Without preserving the original clean price of each day, the modified data values become useless. It is obviously not the best way for cleaning the stream data.

The speed constraints derived from price limit[1] state that the price difference of two consecutive trading days should not be greater than 0.15. The maximum speed $s_{\max} = 0.15$ specifies that the increase amount is no larger than 0.15 in a single trading day from the previous day's settlement price. The minimum speed $s_{\min} = -0.15$ indicates that the decrease should be within 0.5. Moreover, the acceleration constraints can also be obtained, e.g., discovered from data.[2] The maximum acceleration $a_{\max} = 0.1$ indicates that the increase of speed should be within 0.1, and the minimum acceleration $a_{\min} = -0.1$ means that the decrease of speed should be no faster than 0.1.

With speed and acceleration constraints, the imprecise value of day 15 can be detected. It obviously increases too much from the price of the previous day 14. As shown, the speed constraint-based repair preserves more originally clean price values (in blue). Moreover, with both speed and acceleration constraints, since more constraints are utilized, the corresponding repair (in red) is

[1] In some markets, the price limit is specified by a certain percentage.
[2] See Section 5.3 for obtaining the max/min speeds and accelerations on price limit.

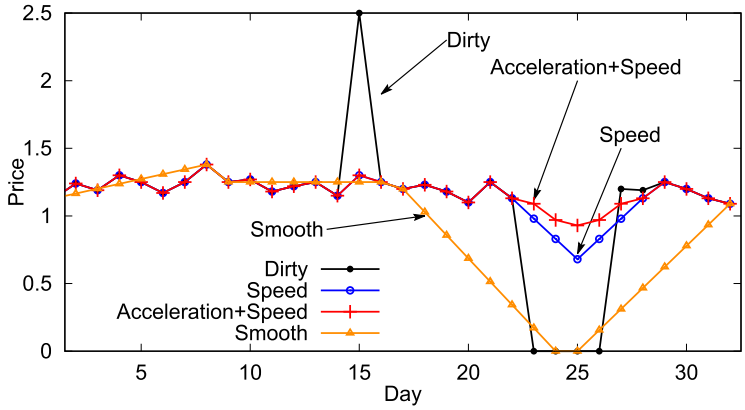


Fig. 1. Smoothing filter seriously alters the original correct data, while the minimum repair under speed (and acceleration) constraints aim to preserve the original information as much as possible.

more accurate than considering the speed constraints solely. In Figure 1, only one point in day 27 is erroneously modified, which is indeed correct. The point in day 28 keeps unchanged when given both speed and acceleration constraints. Such a point, however, is erroneously modified when given only speed constraints.

1.1 Challenges

Unlike the existing techniques on smoothing time series [14], we propose to minimally modify the data values such that the declared speed and acceleration constraints are satisfied. This constraint-based cleaning, however, is non-trivial and challenging especially in the following aspects:

(1) *Soundness.* Owing to the inherent hardness of general data repair problems, a greedy strategy is employed in the existing repair [8]. It modifies values to eliminate currently observed violations (w.r.t. the given constraints) in each round, which may introduce new violations to other data points, and thus evokes another round of repairing. In particular, the greedy repair could be trapped in local optima, and thus cannot eliminate all the violations. In other words, the soundness w.r.t. satisfaction of (speed and acceleration) constraints is not guaranteed.

(2) *Online Computing.* Typically, data repair techniques consider a global optimization function on modifying the entire data [5]. It has to first collect all the data, and then repair them as a whole. Online cleaning on the streaming data is not supported. To enable streaming computation, we have to decompose the global optimum into a list of local optimums on each data point, respectively. Integral cleaning can thus be applied by incrementally computing the local optimal repair on every data point of the sequence in turn.

1.2 Contributions

The preliminary conference version of this article [32] focuses on cleaning the dirty stream data under speed constraints. In this study, along with the speed constraints, we further consider the constraints on acceleration of value changes. A linear time, constant space cleaning approach is presented under both the speed and acceleration constraints. Our main contributions are summarized as follows:

(1) We formalize the repair problem under both the speed and acceleration constraints (in Section 2). By considering the entire sequence as a whole, the monolithic cleaning finds a repaired sequence that minimally differs from the input. Unlike NP-hardness of general data repair

problems [21, 25], we show that stream data cleaning under speed and acceleration constraints can be modeled as a linear programming problem, i.e., polynomial time solvable.

(2) We devise an online cleaning algorithm (in Section 3). To support integral cleaning (i.e., incrementally repair one data point a time in the sequence rather than monolithic cleaning as a whole), we relax the global optimum over the entire sequence to the local optimum in a window. The main idea is to locally compute a data point repair, which is minimal w.r.t. the upcoming data points in a window and also compatible with the previously repaired data points. In particular, to efficiently compute the local optimum, we propose a novel *median-based solution*, following the intuition that a solution with the minimum distance (i.e., as close as possible to each point) probably lies in the middle of the data point candidates. It is notable that soundness w.r.t. speed and acceleration constraints satisfaction is guaranteed in the devised algorithm.

(3) Experiments on real data (in Section 5) demonstrate that our proposal achieves significantly lower L1 error than the existing smoother method [14]. Moreover, compared to the state-of-the-art data repair method [8], the proposed method with local optimum shows up to two orders of magnitude improvement in time costs and much lower L1 error. In addition to evaluating directly the repair performance with synthetic and real-world errors, we further investigate the classification accuracy over the data without/with cleaning. A method improving most the classification accuracy indicates that its repairing is more effective.

1.3 Extensions with Acceleration Constraints

The versions of Proposition 3.1, Proposition 3.3, Lemma 3.4, Proposition 3.5, and Proposition 3.6 appear in Reference [32] for only speed constraints. All the proofs are given in the journal version for the first time. The Local Algorithm 1 is also extended where the returned repair satisfies both the given speed and acceleration constraints. Proposition 3.8 analyzes the correctness of Algorithm 1 w.r.t. the definition of Problem 2. Moreover, in Section 3.3.2, we present a method of online determining dynamic constraints such that the repairing could adapt to the constraint changes in a streaming setting.

The extensions on the proofs w.r.t. acceleration constraints are significant and heavy. (1) In the proof of Proposition 3.1, we show that considering the latest point is sufficient to determine the candidate range. While the case of speed constraint is straightforward using the first two formulas, the proof for acceleration constraints is more complicated, taking almost one page, starting from page 10. (2) In the proof of Proposition 3.3, we illustrate that the optimal solution of other points x_i in a window could be derived from the current x_k not only for speeds (Formulas 23 and 24) and accelerations (Formulas 25 and 26) individually, but also for the case with both speed and accelerations (Formulas 27 and 28). (3) In the proof of Lemma 3.4, we prove that an optimal solution can always be found from the defined set of candidates. When given speed or acceleration constraints only, as illustrated by dot or solid lines in Figure 9, the candidate suggested by c_{j+1} is always larger the corresponding candidate by c_j. However, if both speed and acceleration constraints are specified, then the candidates by c_{j+1} and c_j interact with each other. The problem of whether an optimal solution can be found from the candidates is still open. (4) In the proof of Proposition 3.5, we illustrate the monotonicity of repair costs w.r.t. the corresponding repair values. Similar to the proof of Lemma 3.4, when given speed or acceleration constraints only, the monotonicity can be proved by considering the three cases presented in Figures 11. Again, if both speed and acceleration constraints are specified, then the candidates w.r.t. speed and acceleration interact with each other. The problem of whether the monotonicity still holds is an open problem.

Experiments are conducted over two more datasets, OliveOil and Trace, in addition to Stock and GPS reported in the conference version [32].

Table 1. Notations

Symbol	Description
x	sequence
$x[i]$, x_i	value of the ith data point in x
$t[i]$, t_i	timestamp of the ith data point in x
s	speed constraints
a	acceleration constraints
w	window size of speed and acceleration constraints
n	length of a finite sequence
x'	repair of sequence x
X_i	a set of candidates for x'_i

2 MONOLITHIC CLEANING

In this section, we consider all the data in a sequence as a whole, and perform the monolithic repair towards a globally minimum repair distance. Table 1 lists the notations frequently used in this article.

2.1 Preliminary

Consider a sequence $x = x[1], x[2], \ldots$, where each $x[i]$ is the value of the ith data point. Each $x[i]$ has a timestamp $t[i]$. For brevity, we write $x[i]$ as x_i, and $t[i]$ as t_i. By default, the sequence is ordered by timestamp, i.e., for any $i < j$, we have $t_i < t_j$.

The *speed constraints* $s = (s_{\min}, s_{\max})$ with window size w specify the minimum speed $s_{\min}$ and maximum speed $s_{\max}$ over the sequence x. Likewise, the *acceleration constraints* $a = (a_{\min}, a_{\max})$ denote the minimum acceleration $a_{\min}$ and maximum acceleration $a_{\max}$.

We say that a sequence x *satisfies* the speed constraints s, denoted by $x \vDash s$, if for any x_i, x_j in a window, i.e., $t_i < t_j \leq t_i + w$, the corresponding speed has

$$s_{\min} \leq \frac{x_j - x_i}{t_j - t_i} \leq s_{\max}. \tag{1}$$

Similarly, a sequence x *satisfies* the acceleration constraints a, denoted by $x \vDash a$, if for any x_i, x_j in a window, i.e., $t_i < t_j \leq t_i + w$, the acceleration has

$$a_{\min} \leq \frac{\frac{x_j - x_i}{t_j - t_i} - \frac{x_i - x_{i-1}}{t_i - t_{i-1}}}{t_j - t_i} \leq a_{\max}, \tag{2}$$

where x_{i-1} and x_i are in the same window as well, $t_i - w \leq t_{i-1} < t_i$. Intuitively, the speed constraints restrict the value changes (from x_i to x_j) over time, while the acceleration constraints limit the speed changes (from $\frac{x_i - x_{i-1}}{t_i - t_{i-1}}$ to $\frac{x_j - x_i}{t_j - t_i}$) over time.

Intuitively, the acceleration constraint on a point x_i is not defined symmetrically for several reasons. (1) The considered sequence x is often with irregular time intervals. For example, in Figure 2, we cannot find a point at time 4 to define symmetrically the acceleration of point x_3 at time 3, given point x_2 at time 2. (2) The repair of x_i is determined not only by the subsequent x_{i+1}, symmetric to x_{i-1}, but also $x_{i+2}, x_{i+3}, \ldots$ in a window, as illustrated in Figure 7. (3) Symmetric definition w.r.t. the index i is not always possible. For instance, again in Figure 2, we cannot find a point x_0 to define symmetrically the acceleration of x_3, given the subsequent x_6.

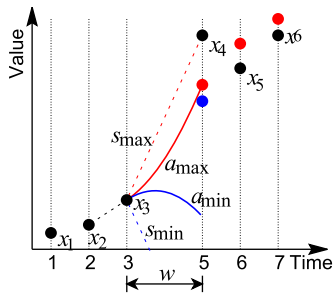


Fig. 2. Possible repairs under speed and acceleration constraints, where red dot line denotes $s_{\max}$, red solid line denotes $a_{\max}$, blue solid line means $a_{\min}$, and blue dot line means $s_{\min}$. The blue point is repaired by Global (Speed+Acceleration), and red points are the repairs of Local (Speed+Acceleration).

The window w denotes a period of time. In real settings, speed and acceleration constraints are often meaningful within a certain period. For example, it is reasonable to consider the maximum walking speed in hours (rather than the speed between two arbitrary observations in different years), since a person usually cannot keep on walking in his/her maximum speed for several years without a break. In other words, it is sufficient to validate the speed (and acceleration) w.r.t. the points x_i, x_j in a window $w = 24$ hours, i.e., whether $s_{\min} \le \frac{x_j - x_i}{t_j - t_i} \le s_{\max}$ (and $a_{\min} \le \frac{\frac{x_j - x_i}{t_j - t_i} - \frac{x_i - x_{i-1}}{t_i - t_{i-1}}}{t_j - t_i} \le a_{\max}$) for $t_i < t_j \le t_i + w$ (and $t_i - w \le t_{i-1} < t_i$). In contrast, considering the speed and acceleration w.r.t. two points in an extremely large period (e.g., two observation points in different years) is meaningless and unnecessary. Similar examples include the speed constraints on stock price whose daily limit is directly determined by the price of the *last* trading day, i.e., with window size 1.[3]

The speed constraints s and acceleration constraints a can be either positive (restricting increase) or negative (on decrease). In practice, the speeds (e.g., running or driving) usually do not keep on increasing or decreasing forever, i.e., $a_{\min} \le 0 \le a_{\max}$. In most scenarios, the speed and acceleration constraints are natural, e.g., the fuel consumption of a crane should not be negative and not exceed 40 liters per hour, while some others could be derived. (See Section 5.3 for a discussion on obtaining appropriate constraints.)

A *repair* x' of x is a modification of the values x_i to x'_i where $t'_i = t_i$. Referring to the minimum modification rule in data repairing [5], the repair distance is evaluated by the difference between the original x and the repaired x',

$$\Delta(x, x') = \sum_{x_i \in x} |x_i - x'_i|. \tag{3}$$

As illustrated in Formula 3, we use L1 norm to define the repair cost on value modification. The reason is that the repairing problem formulated by L1 norm in Figure 3 below can thus be transformed to a linear programming problem in Figure 4. Efficient algorithms apply. We time-align the cleaned series x' with the dirty one x first before computing the distance. Note that we assume the timestamp t_i of each point x_i to be clean in this study. Therefore, the repaired point x'_i naturally aligns with the original x_i referring to the index i. In other words, no lead and lag effects might happen during cleaning. Nevertheless, it is an interesting topic to consider both dirty values and dirty timestamps in the future study, where lead and lag effects need to be considered.

Example 2.1 (Constraints, Violations, and Repairs). Consider a sequence $x = \{0, 0.5, 2, 12, 10, 12\}$ of six data points, with timestamps $t = \{1, 2, 3, 5, 6, 7\}$. Figure 2 illustrates the data points (in black).

[3]The window size may be fixed, e.g., $w = 1$ for stock price daily limit.

$$\min \quad \sum_{i=1}^{n} |x_i - x_i'| \tag{4}$$

$$\text{s.t.} \quad \frac{x_j' - x_i'}{t_j - t_i} \le s_{\max}, \qquad t_i < t_j \le t_i + w, 1 \le i \le n-1, 2 \le j \le n \tag{5}$$

$$\frac{x_j' - x_i'}{t_j - t_i} \ge s_{\min}, \qquad t_i < t_j \le t_i + w, 1 \le i \le n-1, 2 \le j \le n \tag{6}$$

$$\frac{\frac{x_j'-x_i'}{t_j-t_i} - \frac{x_i'-x_{i-1}'}{t_i-t_{i-1}}}{t_j - t_i} \le a_{\max}, \qquad t_i < t_j \le t_i + w, t_i - w \le t_{i-1} < t_i, \quad 2 \le i \le n-1, 3 \le j \le n \tag{7}$$

$$\frac{\frac{x_j'-x_i'}{t_j-t_i} - \frac{x_i'-x_{i-1}'}{t_i-t_{i-1}}}{t_j - t_i} \ge a_{\min}, \qquad t_i < t_j \le t_i + w, t_i - w \le t_{i-1} < t_i, \quad 2 \le i \le n-1, 3 \le j \le n \tag{8}$$

Fig. 3. Global optimal repair.

$$\min \quad \sum_{i=1}^{n} u_i + v_i \tag{9}$$

$$\text{s.t.} \quad \frac{u_j - v_j + v_i - u_i - x_i + x_j}{t_j - t_i} \le s_{\max}, \qquad t_i < t_j \le t_i + w, \quad 1 \le i \le n-1, 2 \le j \le n$$

$$\frac{u_j - v_j + v_i - u_i - x_i + x_j}{t_j - t_i} \ge s_{\min}, \qquad t_i < t_j \le t_i + w, \quad 1 \le i \le n-1, 2 \le j \le n$$

$$\frac{\frac{u_j-v_j+v_i-u_i-x_i+x_j}{t_j-t_i} - \frac{u_i-v_i+v_{i-1}-u_{i-1}-x_{i-1}+x_i}{t_i-t_{i-1}}}{t_j - t_i} \le a_{\max}, \qquad t_i < t_j \le t_i + w, t_i - w \le t_{i-1} < t_i, \quad 2 \le i \le n-1, 3 \le j \le n$$

$$\frac{\frac{u_j-v_j+v_i-u_i-x_i+x_j}{t_j-t_i} - \frac{u_i-v_i+v_{i-1}-u_{i-1}-x_{i-1}+x_i}{t_i-t_{i-1}}}{t_j - t_i} \ge a_{\min}, \qquad t_i < t_j \le t_i + w, t_i - w \le t_{i-1} < t_i, \quad 2 \le i \le n-1, 3 \le j \le n$$

$$u_i \ge 0, v_i \ge 0, \qquad 1 \le i \le n$$

Fig. 4. LP transformation.

Suppose that the speed constraints are $s_{\max} = 5$ and $s_{\min} = -5$, the acceleration constraints are $a_{\max} = 1$ and $a_{\min} = -1$.

For a window size $w = 2$, data points x_3 and x_4, with timestamp distance $5 - 3 \le 2$ in a window, satisfy the speed constraints but are identified as violations to $a_{\max} = 1$, i.e., the speed has $-5 < \frac{12-2}{5-3} = 5$ but the acceleration is $\frac{\frac{12-2}{5-3} - \frac{2-0.5}{3-2}}{5-3} = 1.75 > 1$. Similarly, x_4 and x_5 with speed $-5 < \frac{10-12}{6-5} = -2 < 5$ but the acceleration $\frac{\frac{10-12}{6-5} - \frac{12-2}{5-3}}{6-5} = -7 < -1$ also violates $a_{\min} = -1$.

To remedy the violation, a repair x' can be performed, i.e., $x'_4 = 8$ (the blue solid data point). As illustrated in Figure 2, the repaired sequence satisfies both the speed and acceleration constraints. The repair distance is $\Delta(x, x') = |12 - 8| = 4$.

Note that if the window size is too small such as $w = 1$, then the violations between x_3 and x_4 could not be detected, since their timestamp distance is 2. However, if the window size is too large, say $w = 10$, then all the pairs of data points in x have to be compared. Although the same repair x' is obtained, the computation overhead is obviously higher (and unnecessary).

2.2 Global Optimum

The cleaning problem is to find a repaired sequence that satisfies the speed and acceleration constraints and minimally differs from the original sequence, called *global optimum.*

Problem 1. *Given a finite sequence x of n data points, speed constraints s, and acceleration constraints a, the* global optimal repair *problem is to find a repair x' such that $x' \vDash s$, $x' \vDash a$ and $\Delta(x, x')$ is minimized.*

A broad class of repair problems have been found to be NP-hard, for instance, repairing under functional dependencies for categorized data [21], or repairing under denial constraints that supports numeric data [25]. It is not the case for repairing under speed and acceleration constraints.

The global optimal repair problem is formulated in Figure 3, where $x'_i, 1 \le i \le n$, are variables in problem solving. The correctness of the result x' in the aforesaid problem is obvious. Formula 4 is exactly the repair distance in Formula 3 to minimize. The speed constraints in Formula 1 are ensured in Formulas 5 and 6 by considering all the t_j in the window starting from t_i, for each data point i in the sequence. And the acceleration constraints in Formula 2 are specified in Formulas 7 and 8 by considering all the t_j in the window starting from t_i, for each data point i in the sequence.

2.3 Transformation to LP

We transform the global optimal repair problem in Formula 4 to a **linear programming (LP)** problem, so existing solvers can directly be employed.

Let $u_i = \frac{|x'_i - x_i| + (x'_i - x_i)}{2}$ and $v_i = \frac{|x'_i - x_i| - (x'_i - x_i)}{2}$. We have $|x'_i - x_i| = u_i + v_i$ and $x'_i - x_i = u_i - v_i$. It follows the LP transformation in Figure 4, where u_i, v_i are variables in problem solving.

Example 2.2 (Global Optimum, Example 2.1 Continued). Consider again the sequence x in Example 2.1, the speed constraints $s_{\max} = 5$, $s_{\min} = -5$, and the acceleration constraints $a_{\max} = 1$, $a_{\min} = -1$ with window size $w = 2$, as illustrated in Figure 2.

According to Formulas 4 to 8, the constraint predicates declared w.r.t. $s_{\max} = 5$, $s_{\min} = -5$, $a_{\max} = 1$, $a_{\min} = -1$ are

$$\ldots, \quad \frac{x'_4 - x'_3}{5-3} \le 5, \qquad \frac{\frac{x'_4 - x'_3}{5-3} - \frac{x'_3 - x'_2}{3-2}}{5-3} \le 1, \qquad \ldots$$

$$\ldots, \quad \frac{x'_4 - x'_3}{5-3} \ge -5, \qquad \frac{\frac{x'_4 - x'_3}{5-3} - \frac{x'_3 - x'_2}{3-2}}{5-3} \ge -1. \qquad \ldots$$

The corresponding transformation is as follows:

$$\ldots, \quad \frac{u_4 - v_4 + v_3 - u_3 - 2 + 12}{5-3} \le 5, \qquad \frac{\frac{u_4 - v_4 + v_3 - u_3 - 2 + 12}{5-3} - \frac{u_2 - v_2 + v_1 - u_1 - 0.5 + 2}{3-2}}{5-3} \le 1, \qquad \ldots$$

$$\ldots, \quad \frac{u_4 - v_4 + v_3 - u_3 - 2 + 12}{5-3} \ge -5, \qquad \frac{\frac{u_4 - v_4 + v_3 - u_3 - 2 + 12}{5-3} - \frac{u_2 - v_2 + v_1 - u_1 - 0.5 + 2}{3-2}}{5-3} \ge -1, \qquad \ldots$$

where $\ldots, u_4 = \frac{|x'_4 - x_4| + (x'_4 - x_4)}{2}, v_4 = \frac{|x'_4 - x_4| - (x'_4 - x_4)}{2}, \ldots$.

By solving the problem with these constraint predicates (using LP solvers), the global optimal solution is exactly the repair x' in Example 2.1, with $x'_4 = 8$ and the minimum repair distance 4.

Referring to Karmarkar's algorithm [20], it is sufficient to conclude that the global optimal repair problem is polynomial time solvable.

Corollary 2.3. *The global optimal repair can be computed in $O(n^{3.5}L)$ time, where n is the size of sequence, and L is the number of bits of input.*

3 INTEGRAL CLEANING

The global optimum considers the entire sequence as a whole and does not support online cleaning over streaming data. To support integral repair w.r.t. the current short period in a stream, we study the local optimum, which concerns only the constraints locally in a window. By sliding windows in the sequence, the result of local optimum x_{local} guarantees to satisfy the speed and acceleration constraints in the entire sequence, i.e., also a feasible solution to the constraints in Formulas 5, 6, 7, and 8 of global optimum. Compared to the global optimum, the local optimum approach can show significant improvement in time costs (about two orders of magnitude improvement in Figures 17, 18, 19) but without introducing much worse L1 error.

In Section 3.1, we show that to determine the range of candidate values for a data point t_k, it is sufficient to consider the value of only the previous data point t_{k-1}. Then, taking this candidate range into account, we define the local optimum repair problem. Next, in Section 3.2, we prove that an optimal solution can always be constructed w.r.t. the speed and acceleration constraints. Finally, with the method for finding candidates set, we prove that the median of the candidates is optimal given only speed or acceleration constraints.

3.1 Local Optimum

The integral cleaning algorithm iteratively determines the local optimal x'_k, for $k \geq 1$. We assume that data points come in-order, i.e., $t_j < t_i$ for any $j < i$.[4]

3.1.1 Candidate Range. Consider x_k, where $x'_1, \ldots, x'_{k-1}$ have been determined in the previous steps. Referring to the speed constraints, each fixed x'_j, $t_k - w \leq t_j < t_k$, $1 \leq j < k$, indicates a range of candidates for x'_k, i.e., $[x^{\min}_{k,j,s}, x^{\max}_{k,j,s}]$, where

$$x^{\min}_{k,j,s} = x'_j + s_{\min}(t_k - t_j), \tag{10}$$

$$x^{\max}_{k,j,s} = x'_j + s_{\max}(t_k - t_j). \tag{11}$$

Likewise, for acceleration constraints, each fixed x'_j with $t_j - w \leq t_{j-1} < t_j$ also indicates a range of candidates for x'_k, i.e., $[x^{\min}_{k,j,a}, x^{\max}_{k,j,a}]$, where

$$x^{\min}_{k,j,a} = \left(a_{\min}(t_k - t_j) + \frac{x'_j - x'_{j-1}}{t_j - t_{j-1}}\right)(t_k - t_j) + x'_j, \tag{12}$$

$$x^{\max}_{k,j,a} = \left(a_{\max}(t_k - t_j) + \frac{x'_j - x'_{j-1}}{t_j - t_{j-1}}\right)(t_k - t_j) + x'_j. \tag{13}$$

[4]The handling of out-of-order arrival was introduced in the conference version [32]. In short, to handle an out-of-order arrival x_k, $t_k < t_{k-1}$, we reorder the sequence by timestamps, i.e., removing x_k and inserting it as a new x_l where $x_l = x_k$, $t_{l-1} < t_l < t_{l+1}$, $l < k$. The updates introduced by x_l include two aspects: (1) for x_j, $j < l$, where x_l suggests candidates for determining x^{mid}_j; and (2) for x_i, $i > l$, whose candidate range $[x^{\min}_i, x^{max}_i]$ is influenced (directly or indirectly) by x'_l.

To satisfy both speed and acceleration constraints, we define the joint range of candidates for x'_k, $[x^{\min}_{k,j}, x^{\max}_{k,j}]$, where

$$x^{\min}_{k,j} = \max\left(x^{\min}_{k,j,s}, x^{\min}_{k,j,a}\right), \tag{14}$$

$$x^{\max}_{k,j} = \min\left(x^{\max}_{k,j,s}, x^{\max}_{k,j,a}\right). \tag{15}$$

The following proposition states that considering the last x'_{k-1} is sufficient to determine the candidate range of possible repairs for x'_k. The rationale is that for any $1 \le j < i < k$, x'_i should be in the range specified by x'_j as well. In other words, the candidate range of x'_k specified by x'_i is subsumed in the range by x'_j.

Proposition 3.1. *For any* $1 \le j < i < k, t_k - w \le t_j < t_i < t_k$, *we have* $x^{\min}_{k,j} \le x^{\min}_{k,i}$, *and* $x^{\max}_{k,i} \le x^{\max}_{k,j}$.

Proof. The proposition can be iteratively proved by showing $x^{\min}_{k,j} \le x^{\min}_{k,j+1}$ and $x^{\max}_{k,j+1} \le x^{\max}_{k,j}$, where $t_k - w \le t_j < t_{j+1} < t_k, 1 \le j < k$.

According to the definitions of $x^{\min}_{k,j,s}, x^{\max}_{k,j,s}$ and $x^{\min}_{k,j,a}, x^{\max}_{k,j,a}$ in Formulas 10 to 13, we can find that

$$x^{\min}_{k,j,s} - x^{\min}_{k,j+1,s} = x'_j + s_{\min}(t_{j+1} - t_j) - x'_{j+1} \le 0,$$
$$x^{\max}_{k,j,s} - x^{\max}_{k,j+1,s} = x'_j + s_{\max}(t_{j+1} - t_j) - x'_{j+1} \ge 0,$$

i.e., $x^{\min}_{k,j,s} \le x^{\min}_{k,j+1,s}$ and $x^{\max}_{k,j,s} \ge x^{\max}_{k,j+1,s}$. For acceleration, we have

$$\begin{aligned} x^{\min}_{k,j+1,a} - x^{\min}_{k,j,a} &= \left(a_{\min}(t_k - t_{j+1}) + \frac{x'_{j+1} - x'_j}{t_{j+1} - t_j}\right)(t_k - t_{j+1}) + x'_{j+1} - \\ &\quad \left(a_{\min}(t_k - t_j) + \frac{x'_j - x'_{j-1}}{t_j - t_{j-1}}\right)(t_k - t_j) - x'_j \\ &= a_{\min}((t_k - t_{j+1})^2 - (t_k - t_j)^2) + (t_k - t_j)\left(\frac{x'_{j+1} - x'_j}{t_{j+1} - t_j} - \frac{x'_j - x'_{j-1}}{t_j - t_{j-1}}\right). \end{aligned}$$

The acceleration constraint implies $\frac{x'_{j+1} - x'_j}{t_{j+1} - t_j} - \frac{x'_j - x'_{j-1}}{t_j - t_{j-1}} \ge a_{\min}(t_{j+1} - t_j)$. It follows

$$\begin{aligned} x^{\min}_{k,j+1,a} - x^{\min}_{k,j,a} &\ge a_{\min}((t_k - t_{j+1})^2 - (t_k - t_j)^2 + (t_k - t_j)(t_{j+1} - t_j)) \\ &= a_{\min}(t_{j+1} - t_j)(t_{j+1} - t_k) \ge 0, \end{aligned}$$

i.e., $x^{\min}_{k,j,a} \le x^{\min}_{k,j+1,a}$. Similarly, we have

$$\begin{aligned} x^{\max}_{k,j+1,a} - x^{\max}_{k,j,a} &= \left(a_{\max}(t_k - t_{j+1}) + \frac{x'_{j+1} - x'_j}{t_{j+1} - t_j}\right)(t_k - t_{j+1}) + x'_{j+1} - \\ &\quad \left(a_{\max}(t_k - t_j) + \frac{x'_j - x'_{j-1}}{t_j - t_{j-1}}\right)(t_k - t_j) - x'_j \\ &= a_{\max}((t_k - t_{j+1})^2 - (t_k - t_j)^2) + (t_k - t_j)\left(\frac{x'_{j+1} - x'_j}{t_{j+1} - t_j} - \frac{x'_j - x'_{j-1}}{t_j - t_{j-1}}\right). \end{aligned}$$

Given the maximum acceleration constraint, $\frac{x'_{j+1}-x'_j}{t_{j+1}-t_j} - \frac{x'_j-x'_{j-1}}{t_j-t_{j-1}} \le a_{\max}(t_{j+1}-t_j)$), it has

$$\begin{aligned} x^{\max}_{k,j+1,a} - x^{\max}_{k,j,a} &\le a_{\max}((t_k - t_{j+1})^2 - (t_k - t_j)^2 + (t_k - t_j)(t_{j+1} - t_j)) \\ &= a_{\max}(t_{j+1} - t_j)(t_{j+1} - t_k) \le 0. \end{aligned}$$

As a result, $x^{\max}_{j,k,a} \ge x^{\max}_{j+1,k,a}$. Consequently, with the definitions in Formulas 14 and 15, i.e.,

$$\begin{aligned} x^{\min}_{k,j} &= \max\left(x^{\min}_{k,j,s}, x^{\min}_{k,j,a}\right), & x^{\max}_{k,j} &= \min\left(x^{\max}_{k,j,s}, x^{\max}_{k,j,a}\right), \\ x^{\min}_{k,j+1} &= \max\left(x^{\min}_{k,j+1,s}, x^{\min}_{k,j+1,a}\right), & x^{\max}_{k,j+1} &= \min\left(x^{\max}_{k,j+1,s}, x^{\max}_{k,j+1,a}\right), \end{aligned}$$

it is easy to find $x^{\min}_{k,j} \le x^{\min}_{k,j+1}$ and $x^{\max}_{k,j} \ge x^{\max}_{k,j+1}$. □

For instance, as illustrated in Figure 10, the candidate range of x'_k specified by x'_{k-2}, $[x^{\min}_{k,k-2}, x^{\max}_{k,k-2}]$, subsumes that by x'_{k-1}, $[x^{\min}_{k,k-1}, x^{\max}_{k,k-1}]$. Consequently, we can obtain a tight range of repairing for x'_k by x'_{k-1}, i.e.,

$$\left[x^{\min}_k, x^{\max}_k\right] = \left[x^{\min}_{k,k-1}, x^{\max}_{k,k-1}\right], \tag{16}$$

where $x^{\min}_k = x^{\min}_{k,k-1}$ and $x^{\max}_k = x^{\max}_{k,k-1}$ as defined in Formulas 14 and 15, respectively (with $j = k-1$).

The repair problem thus becomes to find the local optimum x'_k in the range of $[x^{\min}_k, x^{\max}_k]$.

3.1.2 Problem Definition of Local Optimum. We say that a data point x_k *locally satisfies* the speed constraints s, denoted by $x_k \vDash s$, if for any x_i in the window starting from x_k, i.e., $t_k < t_i \le t_k + w$, it has $s_{\min} \le \frac{x_i - x_k}{t_i - t_k} \le s_{\max}$, referring to Formula 1.

Similarly, x_k *locally satisfies* the acceleration constraints a, denoted by $x_k \vDash a$, if for any x_i in the window starting from x_k, i.e., $t_k < t_i \le t_k + w$, it has $a_{\min} \le \frac{\frac{x_i-x_k}{t_i-t_k} - \frac{x_k-x_{k-1}}{t_k-t_{k-1}}}{t_i-t_k} \le a_{\max}$, where $t_k - w \le t_{k-1} < t_k$, according to Formula 2.

Problem 2. *Given a data point x_k in a sequence x, speed constraints s, acceleration constraints a, and a candidate range $[x^{\min}_k, x^{\max}_k]$ for repairing x_k, the* local optimal repair *problem is to find a repair x' such that $x'_k \vDash s$, $x'_k \vDash a$, $x'_k \in [x^{\min}_k, x^{\max}_k]$ and $\Delta(x, x')$ is minimized.*

Similar to the global optimum, we write the local optimal repair problem as in Figure 5, where $x'_i, k \le i \le n$, are the variables in problem solving, and x'_{k-1} is a (previously repaired) value having $0 < t_k - t_{k-1} \le w$.

The local optimal repair in Formula 17 modifies only the data points i with $t_k \le t_i \le t_k + w$ in the window of the current x_k, i.e., much fewer variables. The constraints (in the window) are not sacrificed.

Example 3.2 (Local Optimum). Consider again the sequence $x = \{0, 0.5, 2, 12, 10, 12\}$ in Figure 2, the speed constraints $s_{\max} = 5$, $s_{\min} = -5$ and the acceleration constraints $a_{\max} = 1$, $a_{\min} = -1$ with window size $w = 2$.

Let $k = 4$ be the currently considered data point. Referring to Formulas 18 to 21, the constraint predicates declared w.r.t. $s_{\max} = 5$, $s_{\min} = -5$, $a_{\max} = 1$, $a_{\min} = -1$ are

$$\frac{x'_4 - x'_3}{5-3} \le 5, \qquad \frac{\frac{x'_4-x'_3}{5-3} - \frac{x'_3-x'_2}{3-2}}{5-3} \le 1, \qquad \frac{x'_4 - x'_3}{5-3} \ge -5, \qquad \frac{\frac{x'_4-x'_3}{5-3} - \frac{x'_3-x'_2}{3-2}}{5-3} \ge -1.$$

The local optimal solution with the minimum distance is $x'_4 = 9$ (the red solid point at time 5).

$$\min \quad \sum_{i=1}^{n} |x_i - x'_i| \tag{17}$$

$$\text{s.t.} \quad \frac{x'_i - x'_k}{t_i - t_k} \le s_{\max}, \qquad t_k < t_i \le t_k + w, 1 \le i \le n \tag{18}$$

$$\frac{x'_i - x'_k}{t_i - t_k} \ge s_{\min}, \qquad t_k < t_i \le t_k + w, 1 \le i \le n \tag{19}$$

$$\frac{\frac{x'_i - x'_k}{t_i - t_k} - \frac{x'_k - x'_{k-1}}{t_k - t_{k-1}}}{t_i - t_k} \le a_{\max}, \qquad t_k < t_i \le t_k + w, t_k - w \le t_{k-1} < t_k, 2 < i \le n \tag{20}$$

$$\frac{\frac{x'_i - x'_k}{t_i - t_k} - \frac{x'_k - x'_{k-1}}{t_k - t_{k-1}}}{t_i - t_k} \ge a_{\min}, \qquad t_k < t_i \le t_k + w, t_k - w \le t_{k-1} < t_k, 2 < i \le n \tag{21}$$

$$x_k^{\min} \le x'_k \le x_k^{\max} \tag{22}$$

Fig. 5. Local optimal repair.

3.2 Median-based Solution

Intuitively, a solution with the minimum distance (i.e., as close as possible to each point) probably lies in the *middle* of the data point candidates. We propose to efficiently search the local optimum in the scope of such middle data points, namely, the **median-based solution** (in Proposition 3.6). Following this median-based solution, we devise a linear time algorithm for computing the local optimal repair, instead of $O(n^{3.5}L)$ by LP.

Before presenting the median-based solution, let us first show that computing the local optimum w.r.t. x_k is indeed equivalent to determine an optimal repair x'_k, where the solution of other x'_i (in Formula 17) can be naturally derived.

3.2.1 Reformulating the Local Optimum Problem. We transform the local optimal repair problem in Formula 17 to a new form w.r.t. only one variable x'_k. The idea is to illustrate that there always exists an optimal solution x', whose x'_i can be derived from x'_k.

For any $t_k < t_i \le t_k + w, 1 \le i \le n$, let $y_{i,k,s}^{\min}$ and $y_{i,k,s}^{\max}$ denote the minimum and maximum values of possible x_i given x'_k referring to the speed constraints,

$$y_{i,k,s}^{\min} = x'_k + s_{\min}(t_i - t_k), \tag{23}$$

$$y_{i,k,s}^{\max} = x'_k + s_{\max}(t_i - t_k). \tag{24}$$

And similarly, $y_{i,k,a}^{\min}$ and $y_{i,k,a}^{\max}$ are the minimum and maximum values of possible x_i given x'_k referring to the acceleration constraints,

$$y_{i,k,a}^{\min} = \left(a_{\min}(t_i - t_k) + \frac{x'_k - x'_{k-1}}{t_k - t_{k-1}}\right)(t_i - t_k) + x'_k, \tag{25}$$

$$y_{i,k,a}^{\max} = \left(a_{\max}(t_i - t_k) + \frac{x'_k - x'_{k-1}}{t_k - t_{k-1}}\right)(t_i - t_k) + x'_k, \tag{26}$$

where $t_k - w \le t_{k-1} < t_k$. Consequently,

$$y_{i,k}^{\max} = \min\left(y_{i,k,s}^{\max}, y_{i,k,a}^{\max}\right), \tag{27}$$

$$y_{i,k}^{\min} = \max\left(y_{i,k,s}^{\min}, y_{i,k,a}^{\min}\right), \tag{28}$$

denote the maximum and minimum values of possible x_i given x'_k referring to the speed and acceleration constraints.

Proposition 3.3. *Let x^* be a local optimal solution w.r.t. x_k. With an unlimited candidate range, the following x' is also local optimal, with $x'_k = x^*_k$, and*

$$x'_i = \begin{cases} y^{\max}_{i,k}, & \text{if } x_i > y^{\max}_{i,k} \\ y^{\min}_{i,k}, & \text{if } x_i < y^{\min}_{i,k} \\ x_i, & \text{otherwise} \end{cases} \tag{29}$$

where $t_k < t_i \le t_k + w$, $1 \le i \le n$.

Proof. We prove the correctness in two aspects.

First, $x'_k \vDash s$ and $x'_k \vDash a$ are satisfied. For any x_i, $t_k < t_i \le t_k + w$, according to Formulas 23 to 26, we can find that

$$\frac{y^{\min}_{i,k,s} - x'_k}{t_i - t_k} = s_{\min}, \quad \frac{\frac{y^{\min}_{i,k,a} - x'_k}{t_i - t_k} - \frac{x'_k - x'_{k-1}}{t_k - t_{k-1}}}{t_i - t_k} = a_{\min}, \quad \frac{y^{\max}_{i,k,s} - x'_k}{t_i - t_k} = s_{\max}, \quad \frac{\frac{y^{\max}_{i,k,a} - x'_k}{t_i - t_k} - \frac{x'_k - x'_{k-1}}{t_k - t_{k-1}}}{t_i - t_k} = a_{\max}.$$

In addition, Formulas 28 and 27 lead to $y^{\min}_{i,k} \ge y^{\min}_{i,k,s}$, $y^{\min}_{i,k} \ge y^{\min}_{i,k,a}$ and $y^{\max}_{i,k} \le y^{\max}_{i,k,s}$, $y^{\max}_{i,k} \le y^{\max}_{i,k,s}$. It follows

$$\frac{y^{\min}_{i,k} - x'_k}{t_i - t_k} \ge s_{\min}, \quad \frac{\frac{y^{\min}_{i,k} - x'_k}{t_i - t_k} - \frac{x'_k - x'_{k-1}}{t_k - t_{k-1}}}{t_i - t_k} \ge a_{\min}, \quad \frac{y^{\max}_{i,k} - x'_k}{t_i - t_k} \le s_{\max}, \quad \frac{\frac{y^{\max}_{i,k} - x'_k}{t_i - t_k} - \frac{x'_k - x'_{k-1}}{t_k - t_{k-1}}}{t_i - t_k} \le a_{\max}.$$

Given $s_{\min} \le \frac{x'_k - x'_{k-1}}{t_k - t_{k-1}}$ and $a_{\max} \ge 0$, we have $y^{\max}_{i,k,a} - y^{\min}_{i,k,s} = (a_{\max}(t_i - t_k) + \frac{x'_k - x'_{k-1}}{t_k - t_{k-1}} - s_{\min})(t_i - t_k) \ge 0$, i.e., $y^{\min}_{i,k,s} \le y^{\max}_{i,k,a}$. Similarly, with $\frac{x'_k - x'_{k-1}}{t_k - t_{k-1}} \le s_{\max}$ and $a_{\min} \le 0$, we can also find that $y^{\min}_{i,k,a} \le y^{\max}_{i,k,s}$, since $y^{\min}_{i,k,a} - y^{\max}_{k,i,s} = (a_{\min}(t_i - t_k) + \frac{x'_k - x'_{k-1}}{t_k - t_{k-1}} - s_{\max})(t_i - t_k) \le 0$. It leads to $y^{\min}_{k,i} \le y^{\max}_{i,k}$ referring to Formulas 28 and 27, having

$$s_{\min} \le \frac{y^{\min}_{i,k} - x'_k}{t_i - t_k} \le \frac{y^{\max}_{i,k} - x'_k}{t_i - t_k} \le s_{\max}, \quad a_{\min} \le \frac{\frac{y^{\min}_{i,k} - x'_k}{t_i - t_k} - \frac{x'_k - x'_{k-1}}{t_k - t_{k-1}}}{t_i - t_k} \le \frac{\frac{y^{\max}_{i,k} - x'_k}{t_i - t_k} - \frac{x'_k - x'_{k-1}}{t_k - t_{k-1}}}{t_i - t_k} \le a_{\max}.$$

That is, the assignment of x'_i in Formula 29 satisfies the speed and acceleration constraints.

Second, $\Delta(x, x')$ is minimized. To illustrate the minimum repair distance, we consider three parts of all data points, as indicated in Formula 29.

(1) For any x_i in the window of $t_k < t_i \le t_k + w$ with $x_i > y^{\max}_{i,k}$, e.g., as shown in Figure 6, the assignment is $x'_i = y^{\max}_{i,k}$ according to Formula 29, with distance $|x_i - x'_i| = x_i - y^{\max}_{i,k}$. Indeed, referring to the derivations in the proof of the first aspect, the speed and acceleration constraints require that possible repairs for x_i must be in the range $[y^{\min}_{i,k}, y^{\max}_{i,k}]$. For any other repair x''_i, $y^{\min}_{i,k} \le x''_i \le y^{\max}_{i,k}$, it is easy to see $|x_i - x''_i| \ge |x_i - x'_i|$. That is, x'_i in Formula 29 is minimized.

(2) Similarly, for $x_i < y^{\min}_{i,k}$, we can show that $x'_i = y^{\min}_{i,k}$ in Formula 29 has minimum distance.

(3) For x_i having $y^{\min}_{i,k} \le x_i \le y^{\max}_{i,k}$, the unchanged assignment $x'_i = x_i$ already has the minimum distance 0.

To sum up, since each assignment of x'_i in Formula 29 is minimized w.r.t. the fixed $x'_k = x^*_k$, we have $\Delta(x, x') \le \Delta(x, x^*)$. Given the local optimum x^*, it is sufficient to conclude that x' is also local optimal with the minimum distance $\Delta(x, x')$. □

Formula 29 constructs an optimal solution x' upon x^*_k, where either no change or border change w.r.t. $s_{\max}$, $s_{\min}$, $a_{\max}$, $a_{\min}$ needs to be made. By border changes, we mean $\frac{x'_i - x'_k}{t_i - t_k} = s_{\max}$, $\frac{x'_i - x'_k}{t_i - t_k} = s_{\min}$,

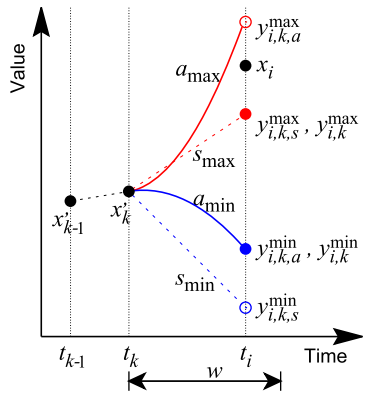


Fig. 6. Build solution from x'_k, where red solid point denotes $y^{\max}_{i,k}$ and blue solid point means $y^{\min}_{i,k}$.

$\frac{\frac{x'_i-x'_k}{t_i-t_k}-\frac{x'_k-x'_{k-1}}{t_k-t_{k-1}}}{t_i-t_k} = a_{\min}$ or $\frac{\frac{x'_i-x'_k}{t_i-t_k}-\frac{x'_k-x'_{k-1}}{t_k-t_{k-1}}}{t_i-t_k} = a_{\max}$. Intuitively, as illustrated in Figure 6, all the values in the range of $[y^{\min}_{i,k}, y^{\max}_{i,k}]$ are valid repair candidates for x'_i. If the speed or acceleration exceeds $y^{\max}_{i,k}$ specified by $s_{\max}$ or $a_{\max}$ in Formula 27, then a repair on the "border" drawn by $y^{\max}_{i,k}$ is obviously the closest to x_i, i.e., with the minimum repair distance. We denote

$$g(x_i, x'_k) = |x_i - x'_i| = \begin{cases} x_i - y^{\max}_{i,k}, & \text{if } x_i > y^{\max}_{i,k} \\ y^{\min}_{i,k} - x_i, & \text{if } x_i < y^{\min}_{i,k} \\ 0, & \text{otherwise} \end{cases}$$

where $t_k < t_i \le t_k + w, 1 \le i \le n$, $y^{\min}_{i,k}$ and $y^{\max}_{i,k}$ are as defined in Formulas 28 and 27 in Proposition 3.3. The local optimal repair problem in Formula 17 can be rewritten as

$$\min_{x'_k} \sum_{i=1}^{n} g\left(x_i, x'_k\right), \tag{30}$$

where x'_k is the only variable in problem solving.

3.2.2 Finite Set of Candidates. According to Proposition 3.3, the local optimal repair problem is equivalent to finding a x'_k that minimizes Formula 30. To this end, we first capture a finite set of candidates for x'_k, where the optimal solution can always be found. We define candidate sets of x_k,

$$X^{\max}_k = \{z^{\max}_{k,i} \mid t_k < t_i \le t_k + w, 1 \le i \le n\}, \tag{31}$$

$$X^{\min}_k = \{z^{\min}_{k,i} \mid t_k < t_i \le t_k + w, 1 \le i \le n\}, \tag{32}$$

where $z^{\max}_{k,i} = \max(z^{\max}_{k,i,s}, z^{\max}_{k,i,a})$, $z^{\min}_{k,i} = \min(z^{\min}_{k,i,s}, z^{\min}_{k,i,a})$,

$$z^{\max}_{k,i,s} = x_i - s_{\max}(t_i - t_k), \tag{33}$$

$$z^{\min}_{k,i,s} = x_i - s_{\min}(t_i - t_k), \tag{34}$$

are the candidates of repairing x_k suggested by x_i with speed constraints, and

$$z^{\max}_{k,i,a} = \frac{x'_{k-1}(t_i - t_k) - (a_{\max}(t_i - t_k)^2 - x_i)(t_k - t_{k-1})}{t_i - t_{k-1}}, \tag{35}$$

$$z^{\min}_{k,i,a} = \frac{x'_{k-1}(t_i - t_k) - (a_{\min}(t_i - t_k)^2 - x_i)(t_k - t_{k-1})}{t_i - t_{k-1}}, \tag{36}$$

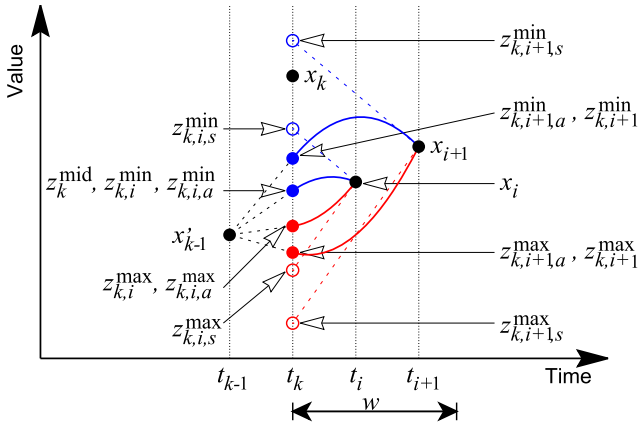


Fig. 7. Capture candidates for x'_k, where blue solid points denote $z^{\min}_{k,i}, z^{\min}_{k,i+1}$, and red solid points are $z^{\max}_{k,i}, z^{\max}_{k,i+1}$.

are the candidates suggested by x_i with acceleration constraints w.r.t. the previously repaired x'_{k-1}, having $0 < t_k - t_{k-1} \leq w$.

Intuitively, as shown in Figure 7, each candidate in $X^{\max}_k$ (red solid points) corresponds to a possible x'_k such that x_i serves as a border repair w.r.t. x'_k (as presented in Figure 6). Referring to the aforesaid discussion on minimum distances of border repairs, it is not surprising to have the following conclusion:

Lemma 3.4. *With speed or acceleration constraints only, and an unlimited candidate range, we can always find a local optimal solution x^* w.r.t. x_k such that $x^*_k \in X^{\min}_k \cup X^{\max}_k \cup \{x_k\}$.*

Proof. Let $m = |\{i | t_k < t_i \leq t_k + w, 1 \leq i \leq n\}|$ be the number of data points in the window starting from k. It is easy to see at most $2m + 1$ candidates in $X^{\min}_k \cup X^{\max}_k \cup \{x_k\}$. We sort the candidates $c_1, \ldots, c_{2m+1}$ in $X^{\min}_k \cup X^{\max}_k \cup \{x_k\}$ for all the m points after x_k in the window starting from t_k, having $c_j \leq c_{j+1}, j = 1, \ldots, 2m$.

Consider a local optimal solution x' built by Formula 29 in Proposition 3.3. If $x'_k \notin X^{\min}_k \cup X^{\max}_k \cup \{x_k\}$, then we construct in the following another repair x'' such that $\Delta(x'', x) \leq \Delta(x', x)$ and $x''_k \in X^{\min}_k \cup X^{\max}_k \cup \{x_k\}$. The major construction steps are outlined as follows: First, we prove that it always has $c_1 \leq x'_k \leq c_{2m+1}$, with two cases $x'_k < c_1$ and $x'_k > c_{2m+1}$. Next, we prove that $c_j < x'_k < c_{j+1}$ is not optimal, i.e., the optimal repair must be $c_j, j = 1, \ldots, 2m+1$. In this part, there are three cases of repairing x_i, (1) $x'_i = x_i$ of unchanged x_i; (2) repair by the maximum constraints, i.e., (2a) $x'_i = y^{\max}_{i,k,s}$ for speed and (2b) $x'_i = y^{\max}_{i,k,a}$ for acceleration; (3) repair by the minimum constraints, i.e., (3a) $x'_i = y^{\min}_{i,k,s}$ for speed, and (3b) $x'_i = y^{\min}_{i,k,a}$ for acceleration. Since there may be several x_i in the window, we should count the number of x_i repaired in the above cases. For (2a) and (3a) on speed, we have three cases (a1), (a2), and (a3) in counting. Likewise, for (2b) and (3b) on acceleration, we count in three cases (b1), (b2), and (b3) as well.

First, we show that it always has $c_1 \leq x'_k \leq c_{2m+1}$. For any $x'_k < c_1$, if only speed constraints s are given, then all the x_i in the window from t_k are modified to $y^{\max}_{i,k,s}$, referring to Formula 24, with distance $x_i - y^{\max}_{i,k,s}$ as the red dot line shown in Figure 8. Likewise, if only acceleration constraints a are given, then we can modify all the x_i in the window from t_k by $y^{\max}_{i,k,a}$, referring to Formula 26, with distance $x_i - y^{\max}_{i,k,a}$, as the red solid line shown in Figure 8. Following the notations of $y^{\max}_{i,k,s}$

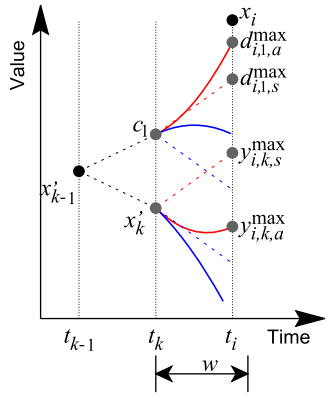


Fig. 8. Impossible case of x'_k smaller than the minimum candidate c_1, $x'_k < c_1 \le x_k$.

and $y^{\max}_{i,k,a}$ in Formulas 24 and 26, let

$$d^{\max}_{i,1,s} = c_1 + s_{\max}(t_i - t_k), \qquad d^{\max}_{i,1,a} = \left(a_{\max}(t_i - t_k) + \frac{c_1 - x'_{k-1}}{t_k - t_{k-1}}\right)(t_i - t_k) + c_1$$

be the candidates for repairing x_i suggested by c_1 in t_k, referring to Formulas 24 and 26. We always have $x_i \ge b^{\max}_{i,1,s}, x_i \ge b^{\max}_{i,1,a}$; otherwise, x_i will introduce some candidates $< c_1$ referring to Formula 31. Given $x'_k < c_1 \le x_k$, it is easy to see another solution x'' with $x''_k = c_1$ whose distances are lower than x'. It contradicts the local optimum of x'. Similar contradiction can also be observed for $x'_k > c_{2m+1}$.

Next, assume that $c_j < x'_k < c_{j+1}$ for some $j \in [1, 2m]$. Let

$$d^{\min}_{i,j,s} = c_j + s_{\min}(t_i - t_k), \tag{37}$$

$$d^{\max}_{i,j,s} = c_j + s_{\max}(t_i - t_k), \tag{38}$$

$$d^{\min}_{i,j,a} = \left(a_{\min}(t_i - t_k) + \frac{c_j - x'_{k-1}}{t_k - t_{k-1}}\right)(t_i - t_k) + c_j, \tag{39}$$

$$d^{\max}_{i,j,a} = \left(a_{\max}(t_i - t_k) + \frac{c_j - x'_{k-1}}{t_k - t_{k-1}}\right)(t_i - t_k) + c_j \tag{40}$$

be the candidates for repairing x_i suggested by c_j in t_k, referring to Formulas 23–26. And similarly,

$$d^{\min}_{i,j+1,s} = c_{j+1} + s_{\min}(t_i - t_k),$$

$$d^{\max}_{i,j+1,s} = c_{j+1} + s_{\max}(t_i - t_k),$$

$$d^{\min}_{j+1,i,a} = \left(a_{\min}(t_i - t_k) + \frac{c_{j+1} - x'_{k-1}}{t_k - t_{k-1}}\right)(t_i - t_k) + c_{j+1},$$

$$d^{\max}_{i,j+1,a} = \left(a_{\max}(t_i - t_k) + \frac{c_{j+1} - x'_{k-1}}{t_k - t_{k-1}}\right)(t_i - t_k) + c_{j+1}$$

denote the candidates for repairing x_i suggested by c_{j+1} in t_k. For repair x'_i, we consider three cases as follows:

(1) If the repair is unchanged $x'_i = x_i$, then it must have

$$x'_k + s_{\min}(t_i - t_k) \le c_{j+1} + s_{\min}(t_i - t_k) = d^{\min}_{i,j+1,s} \le x_i \le d^{\max}_{i,j,s} = c_j + s_{\max}(t_i - t_k) \le x'_k + s_{\max}(t_i - t_k)$$

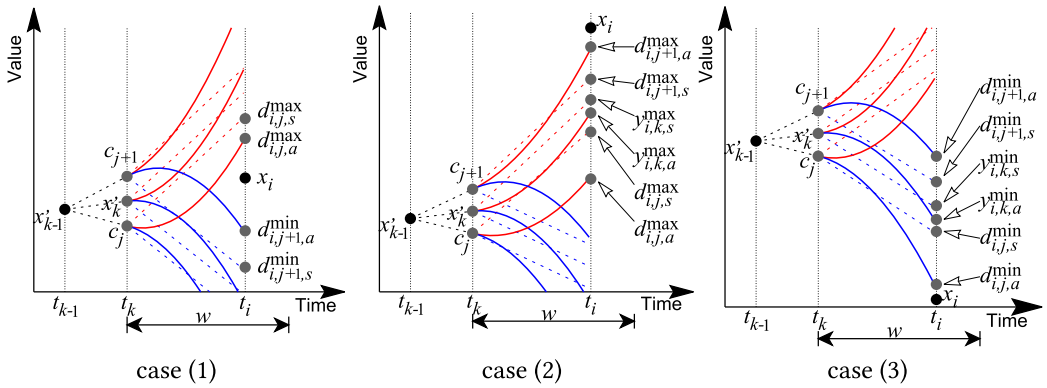


Fig. 9. Moving between candidates.

when given speed constraints s, or

$$\left(a_{\min}(t_i - t_k) + \frac{c_{j+1} - x'_{k-1}}{t_k - t_{k-1}}\right)(t_i - t_k) + c_{j+1} = d^{\min}_{i,j+1,a} \le x_i \le d^{\max}_{i,j,a}$$
$$= \left(a_{\max}(t_i - t_k) + \frac{c_j - x'_{k-1}}{t_k - t_{k-1}})(t_i - t_k\right) + c_j$$

for acceleration constraints a, as case (1) illustrated in Figure 9.

For the other cases, such as

$$d^{\max}_{i,j,s} = c_j + s_{\max}(t_i - t_k) \le x'_k + s_{\max}(t_i - t_k) < x_i \le c_{j+1} + s_{\max}(t_i - t_k),$$

x_i would introduce a candidate between c_j and c_{j+1} according to the definitions of $X^{\min}_k$ and $X^{\max}_k$ in Formulas 31 and 32. Consequently, we can construct another repair with $x''_k = c_j$ or c_{j+1} such that $x''_i = x_i$ is still unchanged on data point i and the distance is the same, $|x''_i - x_i| = |x'_i - x_i| = 0$.

(2a) For the repair $x'_i = y^{\max}_{i,k,s}$ when given speed constraints s only, the distance is $|x_i - x'_i| = x_i - y^{\max}_{i,k,s}$. It always has $x_i \ge d^{\max}_{i,j+1,s}$, as case (2) illustrated in Figure 9. For other cases such as $d^{\max}_{i,j,s} < x_i < d^{\max}_{i,j+1,s}$, x_i would introduce a candidate between c_j and c_{j+1}. To construct another repair with $x''_k = c_j$ or c_{j+1}, we have $x''_i = d^{\max}_{i,j,s}$ or $x''_i = d^{\max}_{i,j+1,s}$, for data point i. The corresponding distance is $|x''_i - x_i| = |x'_i - x_i| - c_j + x'_k$ or $|x''_i - x_i| = |x'_i - x_i| - c_{j+1} + x'_k$.

(2b) For the repair $x'_i = y^{\max}_{i,k,a}$ under acceleration constraints a only, with distance $|x_i - x'_i| = x_i - y^{\max}_{i,k,a}$, it always has $x_i \ge d^{\max}_{i,j+1,a}$. For other cases such as $d^{\max}_{i,j,a} < x_i < d^{\max}_{i,j+1,a}$, x_i would introduce a candidate between c_j and c_{j+1}. We construct another repair $x''_k = c_j$ or c_{j+1} having $x''_i = d^{\max}_{i,j,a}$ or $x''_i = d^{\max}_{i,j+1,a}$, for data point i, with distance $|x''_i - x_i| = |x'_i - x_i| + \frac{t_i - t_{k-1}}{t_k - t_{k-1}}(x'_k - c_j)$ or $|x''_i - x_i| = |x'_i - x_i| + \frac{t_i - t_{k-1}}{t_k - t_{k-1}}(x'_k - c_{j+1})$.

(3a) Similarly, for the repair $x'_i = y^{\min}_{i,k,s}$ with speed constraints s, it always has $x_i \le d^{\min}_{i,j+1,s}$. For other cases such as $d^{\min}_{i,j+1,s} < x_i < d^{\min}_{i,j,s}$, x_i would introduce a candidate between c_j and c_{j+1}. We construct another repair with $x''_k = c_j$ or c_{j+1} having $x''_i = d^{\min}_{i,j,s}$ or $x''_i = d^{\min}_{i,j+1,s}$ as case (3) illustrated in Figure 9. The repair distance on data point i is thus $|x''_i - x_i| = |x'_i - x_i| + c_j - x'_k$ or $|x''_i - x_i| = |x'_i - x_i| + c_{j+1} - x'_k$.

(3b) Again, for the repair $x'_i = y^{\min}_{i,k,a}$ under acceleration constraints a only, it always has $x_i \le d^{\min}_{i,j+1,a}$. For other cases such as $d^{\min}_{i,j+1,a} < x_i < d^{\min}_{i,j,a}$, x_i would introduce a candidate between c_j and c_{j+1}. To construct another repair with $x''_k = c_j$ or c_{j+1}, we can also have $x''_i = d^{\min}_{i,j,a}$ or $x''_i = d^{\min}_{i,j+1,a}$.

The corresponding repair distance on data point i is $|x_i'' - x_i| = |x_i' - x_i| + \frac{t_i - t_{k-1}}{t_k - t_{k-1}}(c_j - x_k')$ or $|x_i'' - x_i| = |x_i' - x_i| + \frac{t_i - t_{k-1}}{t_k - t_{k-1}}(c_{j+1} - x_k')$.

Now, we count the number of data points i of type (2a) and type (3a) repairs, when given speed constraints s only.

(a1) If the counts of type (2a) and type (3a) are equal, then we choose c_j or c_{j+1}, which is more close to x_k, as x_k''. When choosing c_j, the corresponding repair distance is thus

$$\Delta(x, x'') = \Delta(x, x') - (x_k' - c_j) < \Delta(x, x').$$

When choosing c_{j+1}, the corresponding repair distance is

$$\Delta(x, x'') = \Delta(x, x') - (c_{j+1} - x_k') < \Delta(x, x').$$

(a2) If the count of type (2a) is greater than that of type (3a), then we choose $x_k'' = c_{j+1}$. The corresponding repair distance is

$$\Delta(x, x'') \le \Delta(x, x') - c_{j+1} + x_k' + |c_{j+1} - x_k'| \le \Delta(x, x').$$

(a3) If the count of type (2a) is less than that of type (3a), then similarly, we choose $x_k'' = c_j$. The repair distance is

$$\Delta(x, x'') \le \Delta(x, x') + c_j - x_k' + |c_j - x_k'| \le \Delta(x, x').$$

Likewise, for acceleration constraints a, we count the number of data points i of type (2b) and type (3b) repairs.

(b1) If the count of type (2b) equals to that of type (3b), then we choose c_j or c_{j+1}, which is more close to x_k, as x_k''. When choosing c_j, the corresponding repair distance is still

$$\Delta(x, x'') = \Delta(x, x') - (x_k' - c_j) < \Delta(x, x').$$

When choosing c_{j+1}, the corresponding repair distance is again

$$\Delta(x, x'') = \Delta(x, x') - (c_{j+1} - x_k') < \Delta(x, x').$$

(b2) If the count of type (2b) is greater than that of type (3b), then we choose $x_k'' = c_{j+1}$. Its repair distance is

$$\Delta(x, x'') \le \Delta(x, x') + \frac{t_i - t_{k-1}}{t_k - t_{k-1}}(x_k' - c_{j+1}) + |c_{j+1} - x_k'| \le \Delta(x, x').$$

(b3) If the count of type (2b) is less than that of type (3b), then we choose $x_k'' = c_j$, with repair distance

$$\Delta(x, x'') \le \Delta(x, x') + \frac{t_i - t_{k-1}}{t_k - t_{k-1}}(c_j - x_k') + |c_j - x_k'| \le \Delta(x, x').$$

To sum up, we build a repair x'' with $x_k'' = c_j$ or c_{j+1}, and $\Delta(x, x'') \le \Delta(x, x')$. □

3.3 Streaming Computation

The integral cleaning algorithm iteratively determines the local optimal x_k', for $k \ge 1$. We assume that data points come in-order, i.e., $t_j < t_i$ for any $j < i$.

3.3.1 Optimal Solution in Candidate Range. From Lemma 3.4, we can find the local optimal solution w.r.t. x_k is in the set of $X_k^{\min} \cup X_k^{\max} \cup \{x_k\}$. Further, for any x_k', the construction of solution x' is indeed to "shrink" data points in violation to the border. Intuitively, a candidate in the middle of all data points x_i probably has less shrink distances.

Let x_k^{mid} denote the median of all candidates,

$$x_k^{\text{mid}} = \text{median}\left(X_k^{\max} \cup X_k^{\min} \cup \{x_k\}\right). \tag{41}$$

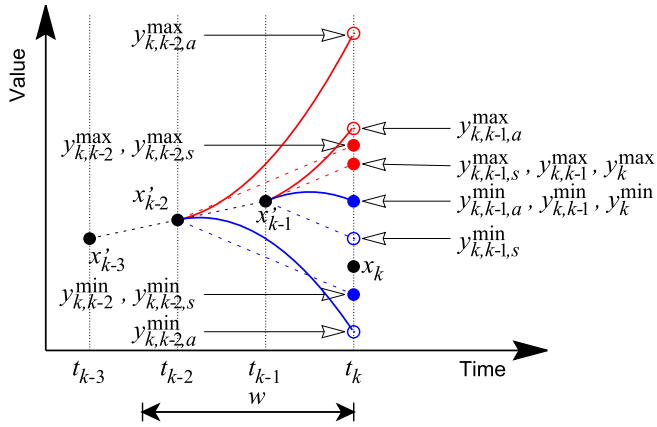


Fig. 10. Candidate range for x'_k, $[x^{\min}_k, x^{\max}_k]$ specified by x'_{k-1} under speed and acceleration constraints.

Since Formula 16 indicates a candidate range $[x^{\min}_k, x^{\max}_k]$ specified by x_{k-1} before x_k, if the suggested solution x^{mid}_k in Formula 41 drops into the range of $[x^{\min}_k, x^{\max}_k]$ in Formula 16, then the optimal solution is directly obtained, i.e., $x'_k = x^{\text{mid}}_k$. Otherwise, we need to re-calculate the repair w.r.t. the range $[x^{\min}_k, x^{\max}_k]$.

Fortunately, we have the following monotonicity of the function in Formula 30.

Proposition 3.5. *With speed or acceleration constraints only, for any* $u_1, u_2, v_1, v_2 \in X^{\min}_k \cup X^{\max}_k \cup \{x_k\}$ *such that* $u_1 \le u_2 \le x^{\text{mid}}_k \le v_1 \le v_2$, *we have*

$$\sum_{i=1}^{n} g(x_i, u_1) \ge \sum_{i=1}^{n} g(x_i, u_2) \ge \sum_{i=1}^{n} g(x_i, x^{\text{mid}}_k), \quad \sum_{i=1}^{n} g(x_i, x^{\text{mid}}_k) \le \sum_{i=1}^{n} g(x_i, v_1) \le \sum_{i=1}^{n} g(x_i, v_2).$$

Proof. Following the same line of proving Lemma 3.4, with speed or acceleration constraints, we sort all the candidates $c_1, \ldots, c_{2m+1}$ in $X^{\min}_k \cup X^{\max}_k \cup \{x_k\}$, i.e., $c_j \le c_{j+1}, j = 1, \ldots, 2m$.

Consider any $c_j \le c_{j+1}$. Let x' be a repair with $x'_k = c_j$ and x'' be another repair with $x''_k = c_{j+1}$. For speed constraints only, referring to the proof of Lemma 3.4, there are three cases for repairing a point i.

(a1) For $d^{\min}_{i,j+1,s} \le x_i \le d^{\max}_{i,j,s}$ as case (1) illustrated in Figure 9, the assignment of x_i is unchanged, with distance 0 for both $x'_i = x_i$ and $x''_i = x_i$.

(a2) For $x_i \ge d^{\max}_{i,j+1,s}$ in case (2) in Figure 9, we have $|x'_i - x_i| = x_i - d^{\max}_{i,j,s}$ and $|x''_i - x_i| = x_i - d^{\max}_{i,j+1,s}$.

(a3) For $x_i \le d^{\min}_{i,j,s}$ in case (3) Figure 9, we have $|x'_i - x_i| = d^{\min}_{i,j,s} - x_i$ and $|x''_i - x_i| = d^{\min}_{i,j+1,s} - x_i$.

In a similar way, for acceleration constraints only, there are 3 cases for repairing a point x_i.

(b1) For $d^{\min}_{i,j+1,a} \le x_i \le d^{\max}_{i,j,a}$, the assignment of x_i is unchanged, with distance 0 for both $x'_i = x_i$ and $x''_i = x_i$.

(b2) For $x_i \ge d^{\max}_{i,j+1,a}$, we have $|x'_i - x_i| = x_i - d^{\max}_{i,j,a}$ and $|x''_i - x_i| = x_i - d^{\max}_{i,j+1,a}$.

(b3) For $x_i \le d^{\min}_{i,j,a}$, we have $|x'_i - x_i| = d^{\min}_{i,j,a} - x_i$ and $|x''_i - x_i| = d^{\min}_{i,j+1,a} - x_i$.

Note that there should not exist any other x_i besides these aforesaid cases, according to the definition of $X^{\min}_k$ and $X^{\max}_k$.

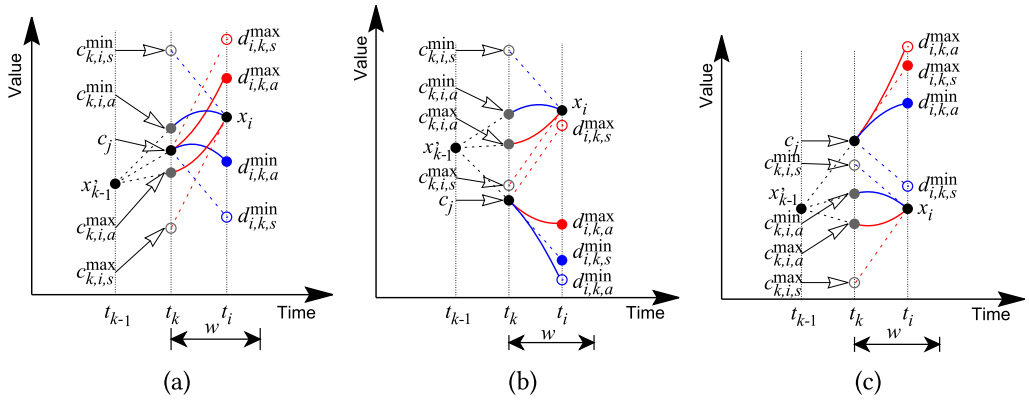


Fig. 11. Relationship of (a) x_i and c_j, (b) $x^{\geq}(c_j)$, and c_j, (c) $x^{\leq}(c_j)$ and c_j.

Referring to Formulas 23 to 26, we define

$$d^{\min}_{i,j,s} = c_j + s_{\min}(t_i - t_k), \tag{42}$$

$$d^{\max}_{i,j,s} = c_j + s_{\max}(t_i - t_k), \tag{43}$$

$$d^{\min}_{i,j,a} = \left(a_{\min}(t_i - t_k) + \frac{c_j - x'_{k-1}}{t_k - t_{k-1}}\right)(t_i - t_k) + c_j, \tag{44}$$

$$d^{\max}_{i,j,a} = \left(a_{\max}(t_i - t_k) + \frac{c_j - x'_{k-1}}{t_k - t_{k-1}}\right)(t_i - t_k) + c_j, \tag{45}$$

the candidates of x_i suggested by any candidate c_j of x_k. For speed constraints only, we define two sets below w.r.t. candidate c_j

$$x^{\geq}(c_j) = \{x_i \mid x_i \geq d^{\max}_{i,j,s}, t_k \leq t_i \leq t_k + w\}, \qquad x^{\leq}(c_j) = \{x_i \mid x_i \leq d^{\min}_{i,j,s}, t_k \leq t_i \leq t_k + w\}.$$

Similarly, for acceleration constraints only, let

$$x^{\geq}(c_j) = \{x_i \mid x_i \geq d^{\max}_{i,j,a}, t_k \leq t_i \leq t_k + w\}, \qquad x^{\leq}(c_j) = \{x_i \mid x_i \leq d^{\min}_{i,j,a}, t_k \leq t_i \leq t_k + w\}.$$

It is notable that x_k is also included in $x^{\geq}(c)$ or $x^{\leq}(c)$ with $t_k \leq t_i$, since x_k is a candidate as well.

Analogous to Formulas 33 and 34, we define

$$c^{\min}_{k,i,s} = x_i - s_{\min}(t_i - t_k), \qquad c^{\max}_{k,i,s} = x_i - s_{\max}(t_i - t_k),$$

the candidates of x_k suggested by x_i on speed constraints s, and

$$c^{\min}_{k,i,a} = \frac{x'_{k-1}(t_i - t_k) - (a_{\min}(t_i - t_k)^2 - x_i)(t_k - t_{k-1})}{t_i - t_{k-1}},$$

$$c^{\max}_{k,i,a} = \frac{x'_{k-1}(t_i - t_k) - (a_{\max}(t_i - t_k)^2 - x_i)(t_k - t_{k-1})}{t_i - t_{k-1}},$$

the candidates suggested by acceleration constraints a.

Referring to the definition of median in Formula 41, we consider the three cases for speed constraints only.

(1) If $d^{\min}_{i,j,s} \leq x_i \leq d^{\max}_{i,j,s}$, then the candidates of x_k generated by x_i w.r.t. speed constraints s are $c^{\max}_{k,i,s} \leq c_j$ and $c^{\min}_{k,i,s} \geq c_j$, as shown in Figure 11(a). They do not affect whether c_j is x^{mid}_k or not.

(2) If $x_i > d^{\max}_{i,j,s}$, which means $x_i \in x^{\geq}(c_j)$, then the candidates of x_k generated by x_i are greater than $< c_j$, i.e., $c^{\max}_{k,i,s} > c_j$ and $c^{\min}_{k,i,s} > c_j$, as shown in Figure 11(b).

(3) Similarly, if $x_i < d^{\min}_{i,j,s}$, i.e., $x_i \in x^{\leq}(c_j)$, then both candidates generated by x_i are lower than c_j, having $c^{\max}_{k,i,s} < c_j$ and $c^{\min}_{k,i,s} < c_j$, as shown in Figure 11(c).

Consequently, we can count x_i in types (2) and (3) w.r.t. c_j.

(a) When $c_j = x^{\text{mid}}_k$, the number of candidates greater than c_j is equal to the number of candidates less than c_j. It means that the aforesaid cases (2) and (3) generate the same number of candidates, i.e., $|x^{\geq}(c_j)| = |x^{\leq}(c_j)|$.

(b) When $c_j \leq x^{\text{mid}}_k$, there are more candidates greater than c_j. That is, the number of candidates generated from (2) is greater than that generated from (3), having $|x^{\geq}(c_j)| \geq |x^{\leq}(c_j)|$.

(c) When $c_j \geq x^{\text{mid}}_k$, i.e., there are more candidates less than c_j. The number of candidates generated from (2) is less than that generated from (3), having $|x^{\geq}(c_j)| \leq |x^{\leq}(c_j)|$.

Similar results could be derived when given acceleration constraints a only, which are also presented in Figure 11. As a result, we have

$$\begin{aligned} |x^{\geq}(c_j)| &= |x^{\leq}(c_j)|, && \text{for } c_j = x^{\text{mid}}_k, \\ |x^{\geq}(c_j)| &\geq |x^{\leq}(c_j)|, && \text{for } c_j \leq x^{\text{mid}}_k, \\ |x^{\geq}(c_j)| &\leq |x^{\leq}(c_j)|, && \text{for } c_j \geq x^{\text{mid}}_k. \end{aligned}$$

Moreover, for any $c_j \leq c_{j+1}$ under speed constraints, since $c_{j+1} + s_{\min}(t_i - t_k) \geq c_j + s_{\min}(t_i - t_k)$, we can find $|x^{\leq}(c_{j+1})| \geq |x^{\leq}(c_j)|$. And with $c_j + s_{\max}(t_i - t_k) \leq c_{j+1} + s_{\max}(t_i - t_k)$, it is easy to see $|x^{\geq}(c_{j+1})| \leq |x^{\geq}(c_j)|$. Similarly, for acceleration constraints,

$$\left(a_{\min}(t_i - t_k) + \frac{c_j - x'_{k-1}}{t_k - t_{k-1}}\right)(t_i - t_k) + c_j \leq \left(a_{\min}(t_i - t_k) + \frac{c_{j+1} - x'_{k-1}}{t_k - t_{k-1}}\right)(t_i - t_k) + c_{j+1},$$

$$\left(a_{\max}(t_i - t_k) + \frac{c_j - x'_{k-1}}{t_k - t_{k-1}}\right)(t_i - t_k) + c_j \leq \left(a_{\max}(t_i - t_k) + \frac{c_{j+1} - x'_{k-1}}{t_k - t_{k-1}}\right)(t_i - t_k) + c_{j+1}$$

imply $|x^{\leq}(c_{j+1})| \geq |x^{\leq}(c_j)|$ and $|x^{\geq}(c_{j+1})| \leq |x^{\geq}(c_j)|$.

Considering the aforesaid three repairing types for all x_i with $t_k \leq t_i \leq t_k + w$, the repair distances of x' and x'' have

$$\begin{aligned} \Delta(x, x'') - \Delta(x, x') &= \sum_{x_i \in x^{\geq}(c_{j+1})} c_j - c_{j+1} + \sum_{x_i \in x^{\leq}(c_j)} c_{j+1} - c_j \\ &= (|x^{\geq}(c_{j+1})| - |x^{\leq}(c_j)|) \cdot (c_j - c_{j+1}) \end{aligned}$$

For $c_j \leq c_{j+1} \leq x^{\text{mid}}_k$, we have $|x^{\geq}(c_{j+1})| \geq |x^{\leq}(c_{j+1})| \geq |x^{\leq}(c_j)|$. It follows $\Delta(x, x'') - \Delta(x, x') \leq 0$. Similarly, for $x^{\text{mid}}_k \leq c_j \leq c_{j+1}$, it has $|x^{\geq}(c_{j+1})| \leq |x^{\geq}(c_j)| \leq |x^{\leq}(c_j)|$, i.e., $\Delta(x, x'') - \Delta(x, x') \geq 0$. □

That is, under certain constraints, for any candidate $u < x^{\max}_k < x^{\text{mid}}_k$, it always has $\sum_{i=1}^{n} g(x_i, u) \geq \sum_{i=1}^{n} g(x_i, x^{\max}_k)$. In this case, $x^{\max}_k$ is the optimal solution in the range of $[x^{\min}_k, x^{\max}_k]$. Similar conclusion can also be made for $v > x^{\min}_k > x^{\text{mid}}_k$.

Proposition 3.6 (Median-based Solution). *With speed or acceleration constraints only, and an unlimited candidate range, a solution x' with x'_i determined by Formula 29 and $x'_k = x^{\text{mid}}_k$ is local optimal.*

Proof. According to Lemma 3.4, with speed or acceleration constraints only, there must exist an $x'_k \in X^{\min}_k \cup X^{\max}_k \cup \{x_k\}$ that can build a local optimum x' by Formula 29 in Proposition 3.3.

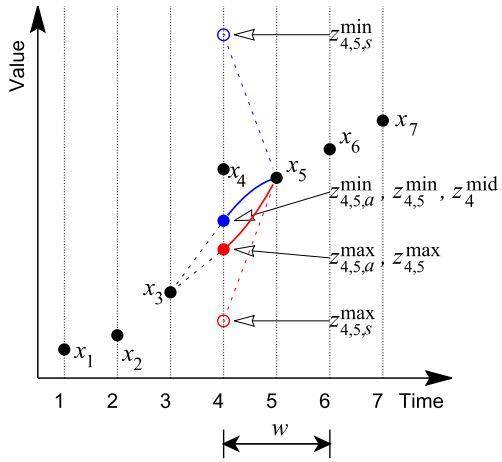


Fig. 12. Capture candidates for x'_4 under speed and acceleration constraints.

Referring to Proposition 3.5, with speed or acceleration constraints only, for any u, v such that $u \le x_k^{\text{mid}} \le v$, we have $\sum_{i=1}^{n} g(x_i, u) \ge \sum_{i=1}^{n} g(x_i, x_k^{\text{mid}}) \le \sum_{i=1}^{n} g(x_i, v)$. That is, x_k^{mid} always has the minimum repair distance among all the candidates in $x'_k \in X_k^{\min} \cup X_k^{\max} \cup \{x_k\}$. □

Example 3.7 (Candidates and Local Repair). Consider another sequence $x = \{0, 0.5, 2, 6.3, 6, 7, 8\}$, in Figure 12. Let $w = 2$. For data points 5 and 6, whose timestamps are within $t_4 + w$, w.r.t. the current $k = 4$. Each data point suggests two candidates w.r.t. $s_{\min} = -5$, $a_{\min} = -1$ and $s_{\max} = 5$, $a_{\max} = 1$ for $X_4^{\min}$ and $X_4^{\max}$, respectively.

For instance, $x_5 = 6$ contributes $z^{\max}_{4,5,s} = 6 + 5(4-5) = 1$ according to Formula 33, and $z^{\min}_{4,5,s} = 6-5(4-5) = 11$ according to Formula 34, given the speed constraints s. For acceleration constraints a, $x_3 = 2$ and $x_5 = 6$ lead to $z^{\min}_{4,5,a} = (-1(5-4)^2(4-3) - 6(4-3) - 2(5-4))/(3-5) = 4.5$ according to Formula 36, and $z^{\max}_{4,5,a} = (1(5-4)^2(4-3) - 6(4-3) - 2(5-4))/(3-5) = 3.5$ according to Formula 35. As a result, we have $z^{\max}_{4,5} = 3.5$ and $z^{\min}_{4,5} = 4.5$.

Similarly, $x_6 = 7$ contributes $z^{\max}_{4,6,s} = 2$, $z^{\min}_{4,6,s} = 12$ with speed constraints. By $x_3 = 2$ and $x_6 = 7$, we can also compute $z^{\max}_{4,6,a} = 2.33$, $z^{\min}_{4,6,a} = 5$ with acceleration constraints, likewise, $z^{\max}_{4,6} = 2.33$ and $z^{\min}_{4,6} = 5$.

It follows $X_4^{\min} = \{4.5, 5\}$, $X_4^{\max} = \{2.33, 3.5\}$, and $x_4^{\text{mid}} = 4.5$ according to Formula 41 with $x_4 = 6.3$. Referring to Proposition 3.6, by $x'_4 = x_4^{\text{mid}}$ and Formula 41, we build a solution $x'_4 = 4.5, x'_5 = 6$ and $x'_6 = 7$.

Consequently, according to Propositions 3.5 and 3.6, the local repair is directly computed by

$$x'_k = \begin{cases} x_k^{\max}, & \text{if } x_k^{\max} < x_k^{\text{mid}} \\ x_k^{\min}, & \text{if } x_k^{\min} > x_k^{\text{mid}} \\ x_k^{\text{mid}}, & \text{otherwise.} \end{cases} \tag{46}$$

Algorithm 1 presents the integral repair of a sequence x under the speed and acceleration constraints s and a. For each data point k in the sequence, $k = 1, 2, \ldots, n$, Line 3 computes the candidate range with Formula 16. By considering all the succeeding data points i in the window of k, Lines 4 to 17 calculate the candidates. With all the candidates captured, we can get x_k^{mid} by Formula 41. Finally, x'_k is obtained following the computation in Formula 46.

ALGORITHM 1: Local(x, s, a)

Input: an ordered sequence x, speed constraints s and acceleration constraints a
Output: a repair x' of x

1 **for** $k \leftarrow 1$ **to** n **do**
2 $\quad X_k^{\min} \leftarrow \emptyset$; $X_k^{\max} \leftarrow \emptyset$;
3 $\quad$ Compute $x_k^{\min}$ and $x_k^{\max}$ with Formula 16;
4 $\quad$ **for** $i \leftarrow k+1$ **to** n **do** // compute candidates
5 $\quad\quad$ **if** $t_i > t_k + w$ **then**
6 $\quad\quad\quad$ break;
7 $\quad\quad$ **end**
8 $\quad\quad$ **if** $0 < t_i - t_k \leq w, 0 < t_k - t_{k-1} \leq w$ **then**
9 $\quad\quad\quad$ Compute $z_{k,i,a}^{\min}, z_{k,i,a}^{\max}$ with Formulas 35–36;
10 $\quad\quad$ **end**
11 $\quad\quad$ **else**
12 $\quad\quad\quad$ $z_{k,i,a}^{\min} = \infty, z_{k,i,a}^{\max} = -\infty$;
13 $\quad\quad$ **end**
14 $\quad\quad$ Compute $z_{k,i,s}^{\min}, z_{k,i,s}^{\max}$ with Formulas 33–34;
15 $\quad\quad$ $X_k^{\min} \leftarrow X_k^{\min} \cup \{\min(z_{k,i,s}^{\min}, z_{k,i,a}^{\min})\}$;
16 $\quad\quad$ $X_k^{\max} \leftarrow X_k^{\max} \cup \{\max(z_{k,i,s}^{\max}, z_{k,i,a}^{\max})\}$;
17 $\quad$ **end**
18 $\quad$ Compute x_k^{mid} with Formula 41;
19 $\quad$ Compute x'_k with Formula 46;
20 **end**
21 **return** x'

Referring to Proposition 3.5, Algorithm 1 returns the local optimum repair when given speed or acceleration constraints only. It is easy to see that the number of distinct data points in a window is at most w. The median in the window can be trivially found in $O(w)$, i.e., the average complexity of quickselect [16]. Considering all the n data points in the sequence, Algorithm 1 runs in $O(nw)$ time. For a fixed w, it is a linear time, constant space algorithm.

Proposition 3.8 (Correctness of Algorithm 1). *The repair x' returned by Algorithm 1 always satisfies both speed constraints s and acceleration constraints a, and is local optimal when given speed or acceleration constraints only.*

Proof. To show the correctness of Algorithm 1, we need to prove $x'_k \vDash s$, $x'_k \vDash a$, $x'_k \in [x_k^{\min}, x_k^{\max}]$ and minimized $\Delta(x, x')$, as specified in Problem 2.

First, we prove $x'_k \in [x_k^{\min}, x_k^{\max}]$. Referring to Formula 46 in Line 18, the solution x'_k must be in the candidate range $[x_k^{\min}, x_k^{\max}]$. According to the definition of $[x_k^{\min}, x_k^{\max}]$ in Formula 16 and Proposition 3.1, it is sufficient to show that x'_k satisfies both speed and acceleration constraints with the previously repaired $x'_{k-1}, x'_{k-2}, \ldots$.

Next, we show $x'_k \vDash s$ and $x'_k \vDash a$. Referring to the computation of x'_i, $t_k < t_i \leq t_k + w$, in Formula 29 in Proposition 3.3, it is sufficient to show that x'_k satisfies both speed and acceleration constraints with $x'_{k+1}, x'_{k+2}, \ldots$ after x'_k in a window.

Finally, we prove that $\Delta(x, x')$ is minimized in the special case of specifying speed constraints s or acceleration constraints a only. For the last case in Formula 46, Proposition 3.6 states that x_k^{mid} is local optimal. For the first two cases w.r.t. the candidate range $[x_k^{\min}, x_k^{\max}]$ in Formula 46, referring

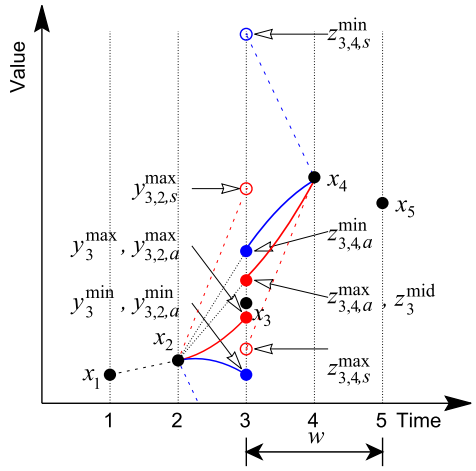


Fig. 13. Capture candidates for x'_3 under speed and acceleration constraints (x^{mid}_3 out of the candidate range $[y^{\min}_3, y^{\max}_3]$).

to the monotonicity in Proposition 3.5, the bounds of the candidate range lead to the minimum repair cost. □

In practice, to minimize the changes, we may heuristically skip the repairing on those points x_k that satisfy the speed and acceleration constraints with its neighbors, i.e., $x^{\min}_k \le x_k \le x^{\max}_k$ and $x^{\min}_{k+1} \le x_{k+1} \le x^{\max}_{k+1}$.

Example 3.9 (x^{mid}_k Out of Candidate Range). Consider a new sequence $x = \{0, 0.5, 2.5, 6.9, 6\}$ in Figure 13. Data points 4 and 5 are within $t_3 + w$ w.r.t. the current $k = 3$ and $w = 2$. According to Formulas 10 to 15, we computed $x^{\min}_{3,2,a} = 0$, $x^{\max}_{3,2,a} = 2$ and $x^{\min}_{3,2,s} = -4.5$, $x^{\max}_{3,2,s} = 6.5$, and the candidate range $[x^{\min}_3, x^{\max}_3] = [0, 2]$.

Following the same line of Example 3.7, we compute $X^{\min}_3 = \{4, 4.2\}$ and $X^{\max}_3 = \{1.33, 3.2\}$ by points 4 and 5 that are in the window of point 3. It follows $z^{\text{mid}}_3 = 3.2$, which is out of the candidate range $[0, 2]$. According to Formula 46, the local repair on $k = 3$ is $x'_3 = 2$.

The integral repair moves on to the next $k = 4$ and terminates when reaching the end of the sequence. A repaired sequence $\{0, 0.5, 2, 4.5, 6\}$ is finally returned.

3.3.2 Dynamic Constraints. In this section, we propose to employ different constraints on speed and acceleration at different times. Specifically, in the example of GPS data of a smartphone, the speed and acceleration could be very different when the user is walking or not. A static constraint on speed or acceleration over the entire sequence cannot detect such differences of moving status. As introduced in Section 5.3, the constraints can be derived from the statistical distribution of speed and acceleration. In this sense, we can derive different constraints on speed and acceleration at different times (for various moving status). Let $[s^k_{\min}, s^k_{\max}]$ and $[a^k_{\min}, a^k_{\max}]$ be the speed and acceleration constraints at time t_k. They are derived online from the statistical distribution of the data before time t_k (optionally after time $t_k - w_d$ where w_d is the window size for online adjusting the constraints). The local repair by Algorithm 1 is then performed on x_k w.r.t. the online determined constraints $[s^k_{\min}, s^k_{\max}]$ and $[a^k_{\min}, a^k_{\max}]$.

4 LIMITATIONS

As illustrated in Proposition 3.8, the proposed Algorithm 1 guarantees to return the optimal solution when given only speed or acceleration constraints. By combining both the speed and acceleration constraints, the candidates suggested by two constraint types may interact with each other, as illustrated in Figure 7 in the proof of Lemma 3.4. The median of the suggested candidates cannot guarantee to be optimal. Therefore, in the general case of specifying both speed and acceleration constraints at the same time, our proposal returns a repair that always satisfies both constraints but may not be the optimal.

As a constraint-based method, the proposed method will not perform if the speed and acceleration constraints are incorrectly specified. Although we have techniques for determining proper constraints by observing the statistical distributions of speed and constraints, as presented in Section 5.3, the results are still sensitive to the constraint settings.

Our constraint-based proposal cannot detect the dirty points indeed satisfying the constraints, e.g., lying in the range of $[x_k^{\min}, x_k^{\max}]$. Moreover, the proposed method may also repair too much when dirty points arrive consecutively, e.g., days 27 and 28 are also modified in Figure 1 while they are not dirty points. In this sense, the repairing under the speed and acceleration constraints would be useful for addressing large spike errors, such as day 15 in Figure 1. For the use cases with small or consecutive errors, it will be a promising future study on extending the speed and acceleration constraints to support such cases.

5 EXPERIMENT

In the experimental evaluation, we employ four real datasets: OliveOil, Stock, Trace, and GPS. The evaluation compares not only the L1 error between the repair result and truth data, but also the classification accuracy over the data with/without repair.

5.1 Experimental Settings

All programs are implemented in Java. Experiments were performed on a server with 2.1 GHz CPU and 128 GB RAM.

5.1.1 Real Dataset Preparation. The Stock[5] dataset records the daily prices of a stock from 1984–09 to 2010–02, with 12,826 data points in total. Since the Stock data is originally clean, following the same line of precisely evaluating the repair effectiveness [3], errors are injected by randomly replacing the values of some data points. For example, an error rate 0.1 denotes that 10% data point values are replaced. For each replaced data point, it takes a random value between the minimum and maximum values in the dataset.

The OliveOil and Trace datasets are from the UCR Time Series Classification Archive.[6] Both datasets have two parts, training set and testing set, which will be used to evaluate the classification over the repaired results. To perform classification, the OliveOil dataset is segmented into 60 time series with the same length 570, and the Trace dataset splits into 200 time series with length 275. Similar to the Stock dataset, we assume that the datasets are originally clean and manually inject errors as aforesaid. Various repairing methods are then performed on each time series.

To evaluate over a real dataset with *true errors* (instead of synthetically injected errors), a real GPS dataset is collected by a person carrying a smartphone and walking around at campus. Since we know exactly the path of walking, a number of 394 dirty points are manually identified (among

[5] http://finance.yahoo.com/q/hp?s=AIP.L+Historical+Prices.
[6] https://www.cs.ucr.edu/~eamonn/time_series_data.

Table 2. GPS Data with Manually Labeled Ground Truth

Method	L1 error	Time cost
Global(Speed+Acceleration)	0.0183	220.3
Global(Speed)	0.0193	200.8
Local(Speed+Acceleration)	0.0187	2.8
Local(Speed)	0.0195	0.8
Holistic	0.2757	88.2
Sequential	0.3193	0.4
EWMA	2.9979	0.2
Median Filter	0.0195	0.1
Savitzky-Golay Filter	0.1132	0.2
Kalman Filter	0.637	0.5
Kernel Smoother	0.0463	1.3
HoloClean	0.0258	9.2
IMR	0.0199	0.15

a total of 2,409 points in the trajectory). True locations of dirty points are also manually labeled, as ground truth.

5.1.2 Evaluation Criteria. Let x_{truth} be the ground truth of clean sequence, and x_{repair} be the repaired sequence. To evaluate the closeness of repair to the truth, we use the L1 error between the ground truth x_{truth} and the repaired sequence x_{repair}. The lower the L1 error between the repair and truth value is, the closer (more accurate) the repair is to the ground truth.

For the classification over datasets, we use the class labels provided in OliveOil and Trace from the UCR Time Series Classification Archive. KNN [37] and XGBoost [7] are employed as the classifiers, with k-fold Cross Validation [34] over the originally Clean data, the Dirty data with errors injected, and the repaired data by various approaches. We use the classification accuracy [30] as follows: $accuracy = \frac{\text{number of correctly classified objects}}{\text{total number of objects}}$.

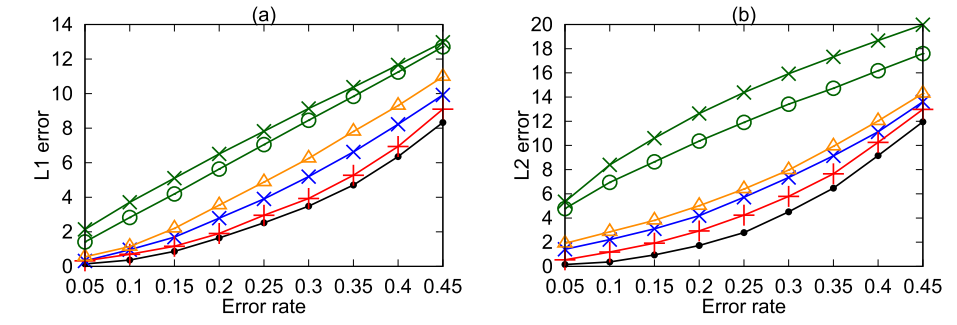
5.2 Comparison to Existing Approaches

In this experiment, we compare our proposed methods Global (Speed+Acceleration), Global (Speed) and Local (Speed+Acceleration), Local (Speed) repairs to the existing approaches, (1) smoother and filter-based EWMA [14], Median Filter [33], Kalman Filter [27], Savitzky-Golay Filter [1], Kernel Smoother [9], IMR [36], and (2) constraint-based Holistic repair [8], HoloClean [28], repair with Sequential Dependency [15].[7]

To tune the parameter of window sizes for each of the smoothing and filtering methods, we conduct a grid search and report the best results. For instance, the tuned window sizes are 5 for Median Filter, 7 for Kernel Smoother, and 5 for Savitzky-Golay Filter.

For our proposed methods, the speed constraints are $s_{\max} = -s_{\min} = 0.5$ for Stock, $s_{\max} = -s_{\min} = 0.6$ for OliveOil, $s_{\max} = -s_{\min} = 7$ for GPS, $s_{\max} = -s_{\min} = 0.3$ for Trace. For acceleration constraints, we use $a_{\max} = -a_{\min} = 0.4$ for Stock, $a_{\max} = -a_{\min} = 0.1$ for OliveOil, $a_{\max} = -a_{\min} = 6$ for GPS, $a_{\max} = -a_{\min} = 0.1$ for Trace. All these constraints are pre-defined by observing the speed and acceleration statistical distributions of the datasets in Figures 22 and 24, respectively. In short, we use the rule of three standard deviations [26] to choose the constraints.

[7]See a detailed introduction of the compared methods in Section 6.



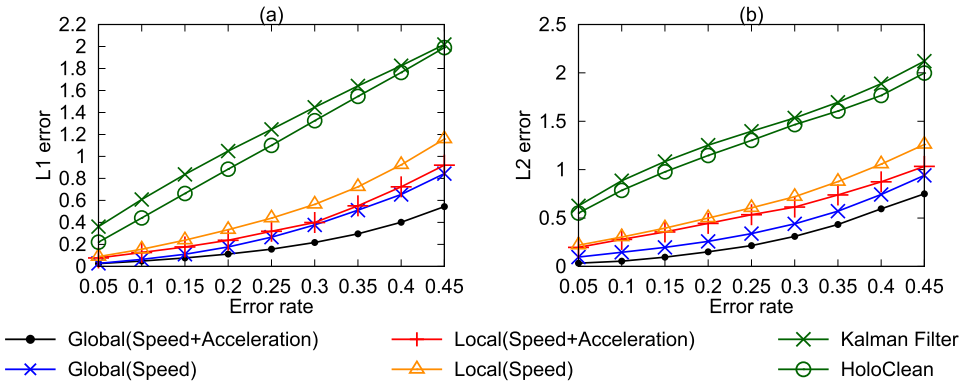
(a) L1 error	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
Global(Speed+Acc)	0.144	0.367	0.871	1.646	2.524	3.487	4.71	6.352	8.328	10.51
Global(Speed)	0.318	0.967	1.691	2.79	3.901	5.198	6.624	8.224	9.924	11.67
Local(Speed+Acc)	0.321	0.7	1.167	1.905	2.959	3.927	5.27	6.94	9.103	11.23
Local(Speed)	0.556	1.146	2.2	3.545	4.879	6.254	7.805	9.3	10.98	12.59
Holistic	0.732	1.698	2.766	3.929	5.4	7.189	8.864	10.26	11.78	12.98
Sequential	1.403	2.815	4.259	5.645	7.063	8.444	9.893	11.27	12.67	14.1
EWMA	2.052	3.074	4.257	5.589	6.913	8.254	9.566	10.92	12.31	13.58
Median Filter	0.69	1.857	3.014	4.296	5.654	6.913	8.371	9.811	11.52	12.95
Savitzky–Golay Filter	1.875	3.569	5.103	6.635	8.066	9.434	10.74	12.05	13.36	14.55
Kalman Filter	2.14	3.714	5.107	6.508	7.823	9.138	10.38	11.66	12.97	14.19
Kernel Smoother	1.517	2.923	4.281	5.716	7.123	8.526	9.906	11.3	12.75	14.08
HoloClean	1.417	2.829	4.193	5.634	7.047	8.456	9.842	11.25	12.71	14.04
IMR	1.492	3.138	4.708	6.125	7.399	8.713	9.828	10.95	11.66	12.25

Fig. 14. Varying error rates, over Stock data with size 12k.

Details for choosing the constraints are presented below (in Section 5.3). Since there are too many approaches in comparison, we only plot the important baselines in Figures 14, 15, 16, and so on, while giving the detailed results in tables. That is, in addition to our proposed Global(Speed+Acc), Local(Speed+Acc), Global(Speed), Local(Speed), we highlight the typical baselines Kalman Filter in the category of smoothing-based methods in Section 6.1 and HoloClean in the category of constraint-based methods in Section 6.2.

5.2.1 Repair Performance. Figures 14, 15, and 16 consider various error rates (denoting the amount of injected errors) with data sizes (the number of data points/the length of the sequence) 12k, 34k, and 55k, in Stock, OliveOil, and Trace, respectively. Figures 17, 18, and 19 study the scalability by varying data sizes (with error rates 0.1, 0.05, and 0.05, respectively). Table 2 presents the results over the GPS dataset with manually labeled ground truth. Note that the errors in the GPS data are real (394 out of 2,409 points). Therefore, we do not have the experiment on varying the rate of synthetically injected errors, e.g., Figure 14 over the Stock data.

For method of EWMA, the L1 error of this smoother method is high, compared with our proposed Global or Local (e.g., in Table 2). The reason is that smoother modifies almost all the data points (mostly are indeed correct data), while the true errors cannot be fully addressed, as illustrated in Figure 1. The corresponding repair L1 error of smoother is thus significantly higher than our proposal in Figures 14(a), 15(a), and 16(a).



(a) L1 error	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
Global(Speed+Acc)	0.025	0.049	0.077	0.112	0.157	0.218	0.296	0.401	0.543	0.728
Global(Speed)	0.028	0.063	0.11	0.176	0.266	0.377	0.511	0.654	0.844	1.048
Local(Speed+Acc)	0.077	0.126	0.177	0.238	0.319	0.4	0.551	0.723	0.921	1.073
Local(Speed)	0.089	0.155	0.239	0.333	0.438	0.564	0.724	0.924	1.158	1.498
Holistic	0.223	0.355	0.503	0.663	0.839	1.017	1.191	1.371	1.574	1.754
Sequential	0.406	0.441	0.662	0.884	1.101	1.325	1.546	1.763	1.99	2.208
EWMA	0.523	0.509	0.772	1.035	1.302	1.592	1.893	2.215	2.564	2.951
Median Filter	0.11	0.178	0.273	0.391	0.515	0.652	0.818	1.039	1.292	1.601
Savitzky–Golay Filter	0.288	0.555	0.807	1.048	1.269	1.492	1.7	1.898	2.1	2.288
Kalman Filter	0.362	0.609	0.837	1.049	1.246	1.448	1.641	1.825	2.018	2.199
Kernel Smoother	0.216	0.421	0.627	0.835	1.037	1.25	1.461	1.668	1.886	2.095
HoloClean	0.221	0.441	0.662	0.884	1.101	1.325	1.546	1.763	1.99	2.208
IMR	0.223	0.45	0.68	0.912	1.143	1.405	1.708	1.992	2.246	2.435

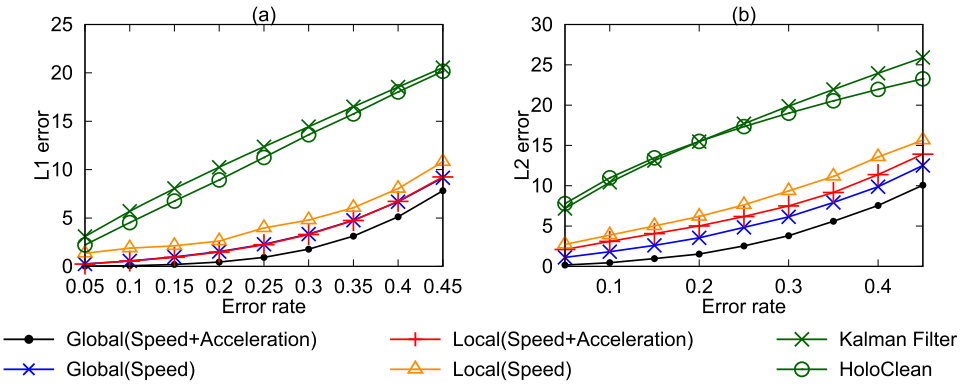
Fig. 15. Varying error rates, over OliveOil data with size 34k.

Among the statistical filtering and smoothing methods, Median Filter [33] performs better than others such as Kalman Filter [27], Savitzky-Golay Filter [1], and Kernel Smooth [9]. The reason is that errors often deviate more significantly from the truth than noises. Such errors in neighbor points as repair candidates affect less in finding the median. It also verifies the rationale of our median-based solution, from the candidates suggested by speed and acceleration constraints.

The Holistic repair, similar to our Global method, considers the data stream as a whole. The corresponding time cost is thus high. It is also notable that data points may still be involved in violations of the speed and acceleration constraints after repairing by Holistic. Thereby, the L1 error of Holistic is not as low as our speed/acceleration methods.

Sequential Dependencies (SDs) consider the constraints on value difference (e.g., ≤ 5) of two consecutive data points, ordered by timestamps. When the time interval of any two consecutive data points is the same, SDs denote the semantics similar to the speed constraints. However, SDs cannot express the semantics on acceleration, i.e., the difference on speeds. Consequently, Sequential is not as effective as our proposal, even worse than EWMA in Figures 17(a) and 19(a).

HoloClean [28] shows a result similar to the Holistic cleaning [8], as shown in Figures 14(a), 15(a), and 16(a) over various datasets. The result is not surprising, since both methods use the (extended) **denial constraints (DCs)**. While both HoloClean [28] and Holistic [8] are proposed for (approximately) cleaning general (tabular) data, our median-based algorithm specialized for repairing data sequences finds more accurate repairs.



(a) L1 error	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
Global(Speed+Acc)	0.022	0.074	0.195	0.443	0.927	1.777	3.12	5.126	7.82	11.14
Global(Speed)	0.251	0.568	0.981	1.526	2.291	3.345	4.766	6.686	9.164	12.2
Local(Speed+Acc)	0.245	0.55	0.945	1.478	2.235	3.294	4.739	6.715	9.25	12.3
Local(Speed)	1.36	1.885	2.129	2.621	3.973	4.785	6.062	8.034	10.81	12.98
Holistic	1.769	3.523	5.289	7.027	8.786	10.56	12.33	14.09	15.82	17.58
Sequential	2.228	4.42	6.616	8.804	11.06	13.22	16.1	19.04	22.68	25.09
EWMA	2.738	5.944	8.794	11.2	13.87	16.82	19.79	23.26	26.84	29.46
Median Filter	2.051	2.243	2.713	3.516	4.726	6.382	8.409	10.82	13.57	16.58
Savitzky–Golay Filter	2.882	5.531	8.04	10.38	12.66	14.86	16.99	19.07	21.09	23.1
Kalman Filter	3.092	5.693	8.064	10.24	12.35	14.44	16.5	18.54	20.55	22.58
Kernel Smoother	2.237	4.428	6.634	8.804	11	13.21	15.42	17.62	19.79	21.98
HoloClean	2.22	4.53	6.752	8.937	11.25	13.41	15.75	18.03	20.16	22.38
IMR	2.274	4.687	7.09	8.716	10.93	12.85	15.33	16.9	18.43	19.82

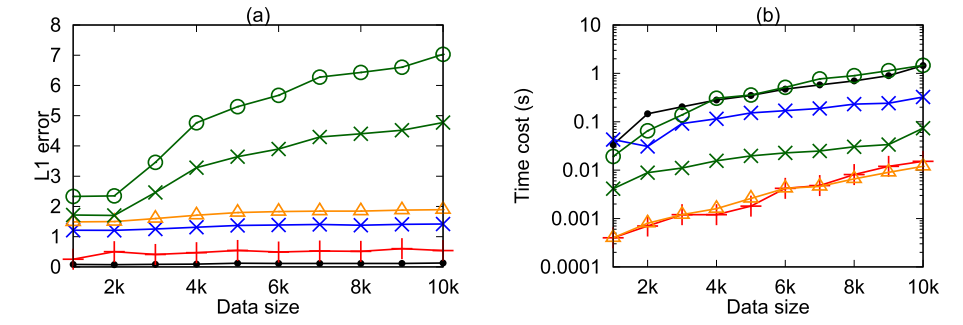
Fig. 16. Varying error rates, over Trace data with size 55k.

IMR [36] performs similarly as the weakly supervised HoloClean [28], as illustrated in Figures 14(a), 15(a), and 16(a). Following the settings in Reference [36], we label about 15% errors in the experiments. Since extensively labeling the errors with truths is unlikely in a streaming setting, the IMR repair is not as accurate as our proposal. Since IMR needs to conduct iterative repairing, its time costs are significantly higher.

The L1 error of Global (Speed+Acceleration) is lowest in all the experiments. Nevertheless, the L1 error of Local (Speed+Acceleration) are similar to Global methods, especially compared with the other baseline approaches in Figures 14(a), 15(a), and 16(a). The corresponding time costs of Local methods are significantly lower than the Global methods, comparable to the efficient EWMA in Figures 17, 18, and 19. The results demonstrate the time performance of Local methods without introducing much L1 error compared to Global in practice.

Note that for the local algorithm, if a previous point x_k is erroneously repaired, then it may propagate the error to the subsequent repair on x_{k+1}. Given an erroneous x_k, the repair on x_{k+1} would be unreliable even with more accurate constraints on both speed and acceleration. Therefore, in some extreme cases, it is possible that Local(Speed+Acceleration) may perform worse than Local(Speed).

In addition to the L1 errors, we also compare the techniques along their L2 errors in Figures 14, 15, and 16. Since there are too many lines in the figures, we plot the most important baselines, i.e., the typical baselines Kalman Filter in the category of smoothing-based methods in

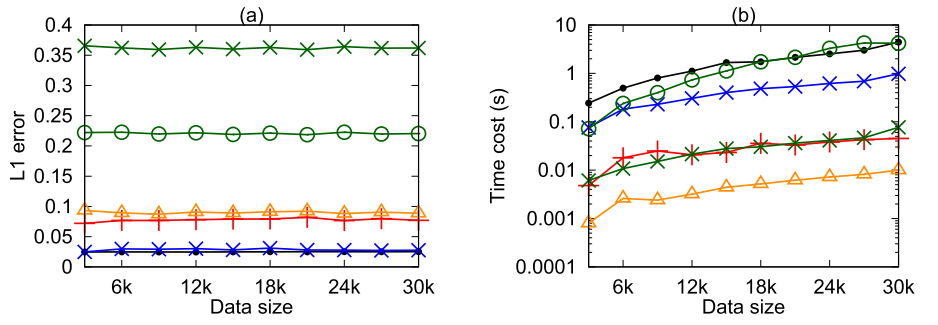


(a) L1 error	1282	2564	3847	5129	6412	7694	8976	10259	11541	12824
Global(Speed+Acc)	0.165	0.166	0.162	0.15	0.148	0.15	0.146	0.143	0.142	0.144
Global(Speed)	0.315	0.316	0.312	0.3	0.298	0.3	0.296	0.293	0.292	0.318
Local(Speed+Acc)	0.316	0.318	0.319	0.311	0.323	0.326	0.32	0.327	0.325	0.321
Local(Speed)	0.535	0.506	0.463	0.523	0.583	0.537	0.566	0.511	0.563	0.556
Holistic	0.756	0.756	0.73	0.748	0.738	0.737	0.748	0.744	0.741	0.732
Sequential	1.41	1.493	1.383	1.356	1.504	1.421	1.428	1.417	1.417	1.403
EWMA	1.955	2.109	2.003	2.078	2.15	1.983	2.099	2.052	1.966	2.052
Median Filter	0.63	0.632	0.649	0.659	0.669	0.673	0.677	0.677	0.683	0.69
Savitzky–Golay Filter	0.671	0.664	0.899	1.223	1.385	1.5	1.641	1.694	1.757	1.875
Kalman Filter	0.772	0.773	1.05	1.405	1.597	1.721	1.883	1.943	2.009	2.14
Kernel Smoother	0.538	0.541	0.729	0.99	1.124	1.217	1.329	1.372	1.427	1.517
HoloClean	0.497	0.494	0.67	0.917	1.041	1.129	1.239	1.279	1.33	1.417
IMR	0.527	0.504	0.686	0.985	1.102	1.174	1.325	1.294	1.368	1.492

(b) Time cost	1282	2564	3847	5129	6412	7694	8976	10259	11541	12824
Global(Speed+Acc)	0.0436	0.1464	0.2036	0.2816	0.3498	0.472	0.5782	0.699	0.8964	1.3428
Global(Speed)	0.0428	0.0308	0.0918	0.1154	0.1522	0.1694	0.1886	0.2314	0.2422	0.2596
Local(Speed+Acc)	0.0004	0.0009	0.0012	0.0012	0.0018	0.0022	0.0028	0.0031	0.0052	0.006
Local(Speed)	0.0004	0.0008	0.0012	0.0012	0.0016	0.002	0.0026	0.0029	0.0039	0.005
Holistic	0.025	0.0856	0.1834	0.4068	0.4732	0.6852	0.721	0.889	0.951	0.9884
Sequential	0.0004	0.0006	0.0012	0.0016	0.0016	0.0018	0.0029	0.0027	0.0038	0.0052
EWMA	0.0002	0.0004	0.0008	0.0008	0.0014	0.0016	0.002	0.0022	0.0026	0.0042
Median Filter	0.0002	0.0003	0.0007	0.0006	0.0007	0.0013	0.0015	0.0015	0.0021	0.0047
Savitzky–Golay Filter	0.0001	0.0003	0.0004	0.0005	0.0007	0.0007	0.0012	0.0011	0.0012	0.018
Kalman Filter	0.0041	0.0089	0.0111	0.0156	0.0197	0.0227	0.0248	0.0304	0.0338	0.0737
Kernel Smoother	0.0005	0.0009	0.0015	0.0019	0.0023	0.0031	0.0038	0.0041	0.0045	0.0213
HoloClean	0.0192	0.065	0.139	0.307	0.3565	0.5157	0.7687	0.895	1.135	0.747
IMR	0.1475	0.1859	0.4307	2.7977	4.6395	6.5739	15.24	8.355	12.766	31.84

Fig. 17. Scalability over Stock data with error rate 0.05.

Section 6.1 and HoloClean in the category of constraint-based methods in Section 6.2. While some larger differences will increase L2 to a greater extent than L1 [13], as shown in subfigures (b), two groups of techniques in L2 errors are still observed as those in L1 errors, i.e., the baselines and our more advanced proposals.

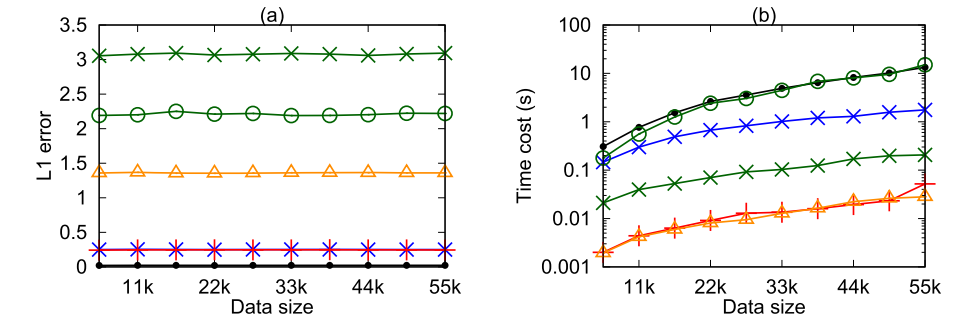


(a) L1 error	3420	6840	10260	13680	17100	20520	23940	27360	30780	34200
Global(Speed+Acc)	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025
Global(Speed)	0.029	0.03	0.029	0.03	0.028	0.031	0.028	0.028	0.027	0.028
Local(Speed+Acc)	0.072	0.077	0.077	0.078	0.079	0.079	0.082	0.077	0.08	0.077
Local(Speed)	0.093	0.089	0.087	0.091	0.089	0.091	0.092	0.088	0.091	0.089
Holistic	0.221	0.221	0.215	0.217	0.221	0.22	0.212	0.211	0.2	0.223
Sequential	0.422	0.399	0.402	0.401	0.405	0.406	0.403	0.402	0.406	0.406
EWMA	0.537	0.526	0.523	0.529	0.525	0.533	0.525	0.525	0.523	0.523
Median Filter	0.11	0.11	0.109	0.11	0.11	0.11	0.109	0.11	0.109	0.11
Savitzky–Golay Filter	0.29	0.291	0.286	0.29	0.286	0.289	0.285	0.291	0.287	0.288
Kalman Filter	0.366	0.362	0.359	0.363	0.36	0.363	0.359	0.364	0.362	0.362
Kernel Smoother	0.218	0.217	0.215	0.217	0.214	0.216	0.214	0.218	0.215	0.216
HoloClean	0.222	0.223	0.22	0.222	0.219	0.221	0.219	0.223	0.22	0.221
IMR	0.222	0.223	0.224	0.225	0.225	0.227	0.223	0.226	0.22	0.223

(b) Time cost (s)	3420	6840	10260	13680	17100	20520	23940	27360	30780	34200
Global(Speed+Acc)	0.2432	0.4968	0.8022	1.1114	1.6732	1.7326	2.1506	2.5298	3.0122	4.4246
Global(Speed)	0.0762	0.183	0.2286	0.3042	0.401	0.4828	0.5328	0.6204	0.6868	0.9802
Local(Speed+Acc)	0.0048	0.018	0.0248	0.0208	0.0232	0.0358	0.033	0.0382	0.0426	0.0452
Local(Speed)	0.0008	0.0026	0.0024	0.0032	0.0044	0.0052	0.0062	0.0072	0.0082	0.01
Holistic	0.0946	0.3176	0.5312	0.9738	1.498	2.3098	2.8602	4.3774	5.6586	5.575
Sequential	0.0008	0.001	0.0016	0.003	0.0022	0.003	0.0036	0.005	0.0058	0.0086
EWMA	0.0004	0.0008	0.001	0.0016	0.0032	0.0024	0.0028	0.0032	0.0056	0.006
Median Filter	0.0002	0.0003	0.0006	0.0014	0.001	0.0012	0.0015	0.002	0.0023	0.0035
Savitzky–Golay Filter	0.0009	0.0012	0.0015	0.0018	0.002	0.0022	0.0024	0.0026	0.0028	0.003
Kalman Filter	0.0061	0.0108	0.0151	0.0215	0.0279	0.0305	0.0363	0.0411	0.0465	0.0766
Kernel Smoother	0.0006	0.0012	0.0019	0.0025	0.0033	0.0041	0.0046	0.0056	0.0059	0.0213
HoloClean	0.0718	0.2392	0.4	0.7334	1.126	1.735	2.149	3.288	4.25	4.19
IMR	0.0273	0.0399	0.0537	0.0599	0.4912	0.772	2.8083	6.7459	9.5324	8.8469

Fig. 18. Scalability over OliveOil data with error rate 0.05.

5.2.2 Application Performance. In addition to evaluating directly the repair performance with synthetic and real-world errors, we further investigate the classification accuracy over the OliveOil data and Trace data without/with data cleaning. There are four classes in OliveOil dataset and four classes in Trace dataset. Since the UCR datasets are already split in training and testing sets, we directly use the already split training set and testing set in the datasets. In the training phase, the



(a) L1 error	5.5k	1.1k	1.65k	22k	27.5k	33k	38.5k	44k	49.5k	55k
Global(Speed+Acc)	0.02	0.022	0.022	0.021	0.022	0.023	0.023	0.022	0.022	0.022
Global(Speed)	0.251	0.253	0.252	0.249	0.252	0.252	0.253	0.251	0.251	0.251
Local(Speed+Acc)	0.243	0.246	0.247	0.248	0.245	0.245	0.249	0.245	0.245	0.245
Local(Speed)	1.358	1.368	1.356	1.353	1.359	1.363	1.363	1.362	1.36	1.36
Holistic	1.738	1.752	1.769	1.747	1.755	1.766	1.759	1.745	1.759	1.769
Sequential	2.235	2.238	2.229	2.232	2.262	2.237	2.243	2.233	2.239	2.228
EWMA	2.718	2.69	2.712	2.673	2.712	2.723	2.704	2.698	2.741	2.738
Median Filter	2.046	2.058	2.055	2.045	2.051	2.051	2.053	2.048	2.053	2.051
Savitzky–Golay Filter	2.84	2.858	2.88	2.854	2.862	2.877	2.866	2.848	2.867	2.882
Kalman Filter	3.054	3.078	3.094	3.066	3.077	3.088	3.079	3.059	3.08	3.092
Kernel Smoother	2.205	2.22	2.237	2.215	2.223	2.234	2.227	2.213	2.228	2.237
HoloClean	2.191	2.201	2.25	2.21	2.22	2.189	2.192	2.202	2.223	2.22
IMR	2.285	2.298	2.19	2.291	2.315	2.302	2.329	2.227	2.277	2.274

(b) Time cost	5.5k	1.1k	1.65k	22k	27.5k	33k	38.5k	44k	49.5k	55k
Global(Speed+Acc)	0.306	0.768	1.537	2.63	3.576	4.92	6.415	8.229	1.17	13.31
Global(Speed)	0.146	0.299	0.488	0.672	0.826	1.015	1.199	1.295	1.571	1.758
Local(Speed+Acc)	0.002	0.004	0.006	0.009	0.013	0.013	0.016	0.019	0.023	0.052
Local(Speed)	0.002	0.004	0.006	0.008	0.01	0.013	0.016	0.022	0.026	0.028
Holistic	0.237	0.743	1.667	3.223	4.041	5.941	9.121	1.76	12.7	19.58
Sequential	0.0008	0.0016	0.0027	0.0036	0.0047	0.006	0.0077	0.0118	0.014	0.0137
EWMA	0.0009	0.0013	0.0021	0.003	0.0044	0.0053	0.0065	0.011	0.0103	0.013
Median Filter	0.0007	0.0019	0.0025	0.0035	0.0048	0.0065	0.0082	0.0101	0.0123	0.013
Savitzky–Golay Filter	0.0005	0.0014	0.0024	0.0035	0.0043	0.0058	0.0083	0.0101	0.0117	0.0134
Kalman Filter	0.021	0.0396	0.0527	0.0701	0.0921	0.1035	0.1247	0.1707	0.1977	0.207
Kernel Smoother	0.0023	0.0045	0.007	0.0103	0.0122	0.0163	0.0198	0.0215	0.0261	0.032
HoloClean	0.1785	0.5585	1.253	2.421	3.035	4.462	6.849	8.082	9.542	15.07
IMR	0.001	0.005	0.0098	0.2209	1.232	7.786	25.98	38.58	45.5	50.58

Fig. 19. Scalability over Trace data with error rate 0.05.

classifiers KNN [37] and XGBoost [7] are learned over three types of data, (1) the Clean data of the original datasets, (2) the Dirty data with errors injected in the original datasets, and (3) the data with injected errors repaired by different approaches, as reported in Section 5.2.1. In the testing phase, these learned models are then evaluated over the clean testing data. It is not surprising that a method repairing the dirty data more accurately improves more the classifier training and thus the classification accuracy (i.e., closer to Clean). In this sense, the relationships among all the

methods in the new Figures 20 and 21 are generally similar to those with models trained over clean data and tested in clean/dirty/repaired data,

Figure 20 presents the classification results over the datasets injected with various rates of errors. First, it is not surprising that the more the errors are injected, the lower the classification accuracy will be. By repairing the errors, the classification is more or less improved compared to the Dirty data. The results verify the necessity of conducting (accurate) data cleaning. Indeed, the classification accuracy is generally proportional to the repair performance in Figures 15 and 16. That is, the Global (Speed+Acceleration) method showing the best L1 error in repairing has the highest classification accuracy (closest to Clean) as well.

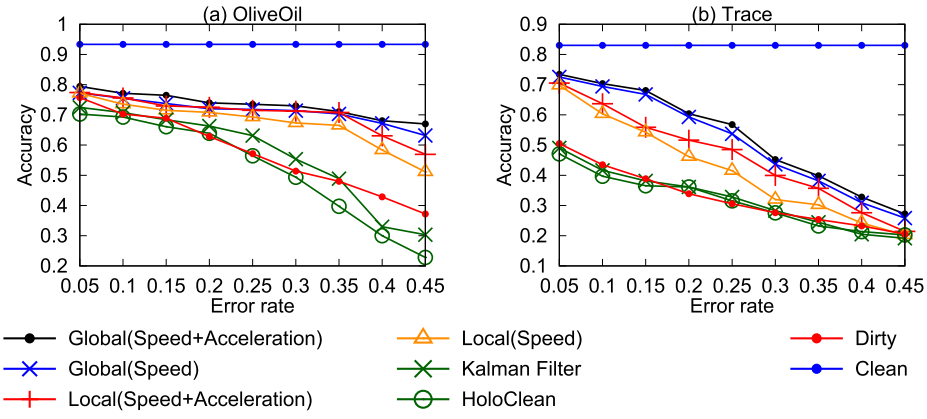
Again, we tune the parameter of EWMA by a grid search and report the best results. Specifically, for the tuned EWMA method, the rates of weighted descent are $\alpha = 0.072$ in the Trace dataset and $\alpha = 0.063$ in the OliveOil dataset. While the tuned EWMA method has results comparable to others smoothing/filtering methods such as Kernel Smooth, the corresponding classification accuracy is still much lower than our proposal. The reason is that, as illustrated in the example in Figure 1, the smoother methods may over-change almost all data points, most of which do not involve errors.

Figure 21 presents the classification accuracy with XGBoost. While XGBoost shows better performance than KNN in the original Clean data, the classification accuracy of XGBoost drops more significantly with the increase of the error rate. Nevertheless, the results of different approaches in Figure 21 are generally proportional to those with KNN in Figure 20. That is, the proposed Global (Speed+Acceleration) having the best repair performance in Figure 21 shows a classification accuracy closest to the Clean data.

5.3 Capturing Constraints

We note that in most scenarios, the speed and acceleration constraints are natural, e.g., the walking speed of a person, while some others could be derived. For dataset Stock, the speed constraints are naturally derived by the business semantics. The price limit in the market declares that the increase or decrease of daily price should not exceed $l \cdot r$ where l is the price of the last trading day and $r = 10\%$ is a percentage. Specifically, while the daily increase or decrease of the price should not exceed 10%, the speed constraint is declared using the absolute price value instead of percentage. In this sense, the constraints are different for different stocks if the dataset contains more than one stock. The GPS dataset is collected by a person carrying a smartphone and walking around at campus. We require seven meters per second as the maximum walking speed of the person. To repair two-dimensional data, we declare speed constraints separately for each dimension. The repairing is conducted separately as well, in the sequences of longitude and latitude, respectively. Such constraints on individual dimensions are stricter than the constraints defined w.r.t. two-dimensional distance. That is, the returned repair will always satisfy (be consistent with) the constraints on the velocity in the two-dimensional space.

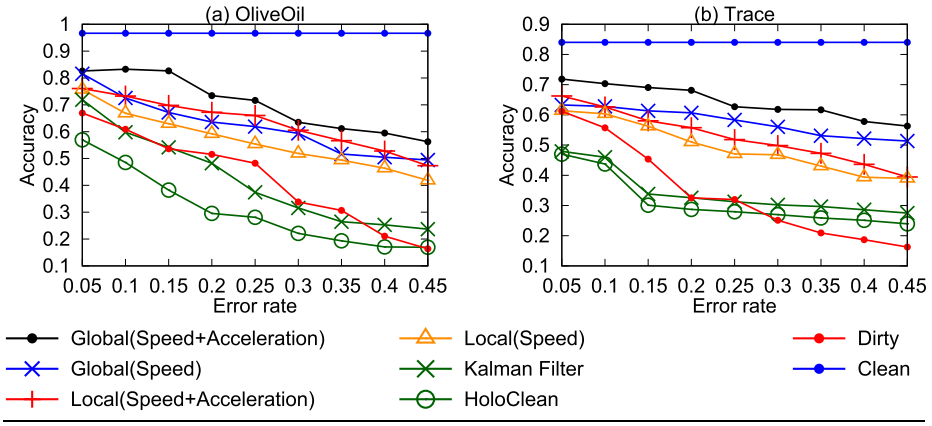
Nevertheless, for a particular domain where speed and acceleration knowledge is not available, the speed and acceleration constraints can be extracted from data. We consider the statistical distribution of speeds and accelerations by sampling data over Stock, OliveOil, GPS, and Trace in Figures 22 and 24. The constraints $s_{\min}$, $s_{\max}$, $a_{\min}$ and $a_{\max}$ are determined by observing the distributions of speeds and accelerations. Referring to statistics, confidence intervals are typically stated at the 95% confidence level [31]. In other words, 95% of the speeds are regarded as accurate (within $[s_{\min}, s_{\max}]$), and similarly for acceleration (within $[a_{\min}, a_{\max}]$). Moreover, the rule of three standard deviations [26] could also be applied to determine the aforesaid ranges on speeds and accelerations (with confidence level 99.73% for normal distribution). It suggests $s_{\max} = -s_{\min} = 0.5$ for Stock, $s_{\max} = -s_{\min} = 0.6$ for OliveOil, $s_{\max} = -s_{\min} = 7$ for GPS, $s_{\max} = -s_{\min} = 0.3$ for Trace, in Figure 23. Similarly, for acceleration, we have $a_{\max} = -a_{\min} = 0.4$ for Stock, $a_{\max} = -a_{\min} = 0.1$



(a) OliveOil	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
Global(Speed+Acc)	0.794	0.771	0.765	0.74	0.735	0.73	0.711	0.681	0.67	0.574
Global(Speed)	0.773	0.755	0.737	0.72	0.718	0.714	0.703	0.672	0.632	0.539
Local(Speed+Acc)	0.774	0.756	0.729	0.725	0.715	0.712	0.708	0.631	0.569	0.517
Local(Speed)	0.771	0.736	0.715	0.708	0.693	0.673	0.665	0.584	0.511	0.503
Holistic	0.744	0.73	0.692	0.653	0.566	0.507	0.471	0.458	0.415	0.371
Sequential	0.743	0.727	0.684	0.64	0.557	0.496	0.454	0.436	0.407	0.365
EWMA	0.725	0.697	0.662	0.604	0.524	0.49	0.428	0.31	0.26	0.196
Median Filter	0.755	0.731	0.728	0.704	0.682	0.656	0.601	0.505	0.392	0.363
Savitzky–Golay Filter	0.727	0.677	0.631	0.552	0.485	0.44	0.392	0.33	0.271	0.225
Kalman Filter	0.725	0.707	0.684	0.663	0.631	0.554	0.489	0.33	0.303	0.272
Kernel Smoother	0.714	0.716	0.682	0.673	0.638	0.562	0.532	0.361	0.342	0.293
HoloClean	0.703	0.693	0.661	0.639	0.565	0.494	0.397	0.3	0.228	0.203
IMR	0.714	0.7	0.676	0.629	0.553	0.506	0.405	0.327	0.268	0.214
Dirty	0.757	0.703	0.689	0.629	0.571	0.514	0.48	0.429	0.372	0.329
Clean	0.933	0.933	0.933	0.933	0.933	0.933	0.933	0.933	0.933	0.933

(b) Trace	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
Global(Speed+Acc)	0.734	0.703	0.681	0.604	0.568	0.452	0.398	0.328	0.272	0.239
Global(Speed)	0.726	0.694	0.668	0.593	0.537	0.436	0.381	0.309	0.258	0.216
Local(Speed+Acc)	0.705	0.637	0.559	0.516	0.485	0.399	0.357	0.276	0.214	0.21
Local(Speed)	0.698	0.603	0.543	0.462	0.415	0.32	0.302	0.242	0.203	0.193
Holistic	0.464	0.436	0.412	0.393	0.361	0.326	0.293	0.24	0.216	0.192
Sequential	0.446	0.424	0.395	0.373	0.351	0.322	0.261	0.219	0.22	0.194
EWMA	0.439	0.395	0.37	0.32	0.28	0.232	0.207	0.187	0.18	0.173
Median Filter	0.664	0.585	0.528	0.483	0.427	0.317	0.296	0.237	0.198	0.163
Savitzky–Golay Filter	0.479	0.416	0.372	0.36	0.326	0.276	0.237	0.209	0.189	0.17
Kalman Filter	0.49	0.416	0.382	0.361	0.328	0.283	0.244	0.205	0.192	0.178
Kernel Smoother	0.468	0.393	0.364	0.352	0.307	0.264	0.229	0.209	0.192	0.17
HoloClean	0.471	0.397	0.365	0.361	0.315	0.276	0.232	0.213	0.202	0.172
IMR	0.431	0.39	0.34	0.321	0.303	0.277	0.219	0.209	0.191	0.198
Dirty	0.505	0.435	0.388	0.339	0.326	0.32	0.304	0.272	0.205	0.194
Clean	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83	0.83

Fig. 20. Classification accuracy over datasets OliveOil and Trace for various error rates with kNN.



(a) OliveOil	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
Global(Speed+Acc)	0.826	0.833	0.826	0.734	0.716	0.635	0.612	0.595	0.563	0.528
Global(Speed)	0.816	0.725	0.671	0.636	0.619	0.594	0.517	0.505	0.494	0.473
Local(Speed+Acc)	0.761	0.733	0.698	0.672	0.66	0.605	0.566	0.528	0.473	0.39
Local(Speed)	0.757	0.669	0.63	0.592	0.554	0.519	0.494	0.464	0.419	0.361
Holistic	0.758	0.662	0.609	0.537	0.488	0.416	0.386	0.375	0.316	0.273
Sequential	0.722	0.639	0.582	0.564	0.517	0.432	0.392	0.391	0.361	0.326
EWMA	0.673	0.615	0.575	0.391	0.34	0.273	0.227	0.164	0.135	0.128
Median Filter	0.762	0.712	0.68	0.66	0.603	0.546	0.505	0.397	0.369	0.319
Savitzky–Golay Filter	0.697	0.531	0.508	0.382	0.35	0.329	0.252	0.238	0.217	0.202
Kalman Filter	0.719	0.598	0.541	0.482	0.374	0.316	0.265	0.253	0.237	0.216
Kernel Smoother	0.542	0.507	0.397	0.323	0.315	0.277	0.251	0.217	0.203	0.19
HoloClean	0.57	0.485	0.382	0.295	0.281	0.221	0.193	0.171	0.169	0.157
IMR	0.548	0.509	0.383	0.279	0.261	0.252	0.219	0.206	0.173	0.17
Dirty	0.669	0.609	0.536	0.515	0.483	0.338	0.307	0.211	0.164	0.109
Clean	0.967	0.967	0.967	0.967	0.967	0.967	0.967	0.967	0.967	0.967

(b) Trace	0.05	0.1	0.15	0.2	0.25	0.3	0.35	0.4	0.45	0.5
Global(Speed+Acc)	0.719	0.704	0.69	0.681	0.627	0.618	0.617	0.578	0.563	0.551
Global(Speed)	0.633	0.628	0.613	0.607	0.583	0.561	0.531	0.521	0.513	0.508
Local(Speed+Acc)	0.663	0.626	0.58	0.557	0.518	0.498	0.473	0.436	0.394	0.388
Local(Speed)	0.614	0.603	0.562	0.51	0.471	0.468	0.431	0.394	0.39	0.354
Holistic	0.661	0.62	0.517	0.501	0.505	0.438	0.385	0.356	0.319	0.302
Sequential	0.614	0.602	0.455	0.446	0.421	0.391	0.371	0.306	0.317	0.304
EWMA	0.521	0.516	0.427	0.396	0.376	0.326	0.291	0.279	0.222	0.206
Median Filter	0.561	0.558	0.538	0.517	0.51	0.507	0.496	0.467	0.432	0.383
Savitzky–Golay Filter	0.457	0.391	0.336	0.317	0.307	0.29	0.286	0.262	0.267	0.187
Kalman Filter	0.479	0.46	0.338	0.326	0.312	0.302	0.297	0.286	0.275	0.253
Kernel Smoother	0.45	0.398	0.299	0.284	0.275	0.264	0.257	0.24	0.227	0.207
HoloClean	0.47	0.437	0.301	0.287	0.279	0.27	0.259	0.251	0.239	0.219
IMR	0.453	0.431	0.297	0.281	0.272	0.269	0.26	0.253	0.239	0.213
Dirty	0.612	0.557	0.454	0.325	0.32	0.251	0.209	0.187	0.163	0.151
Clean	0.84	0.84	0.84	0.84	0.84	0.84	0.84	0.84	0.84	0.84

Fig. 21. Classification accuracy over datasets OliveOil and Trace for various error rates with XGBoost.

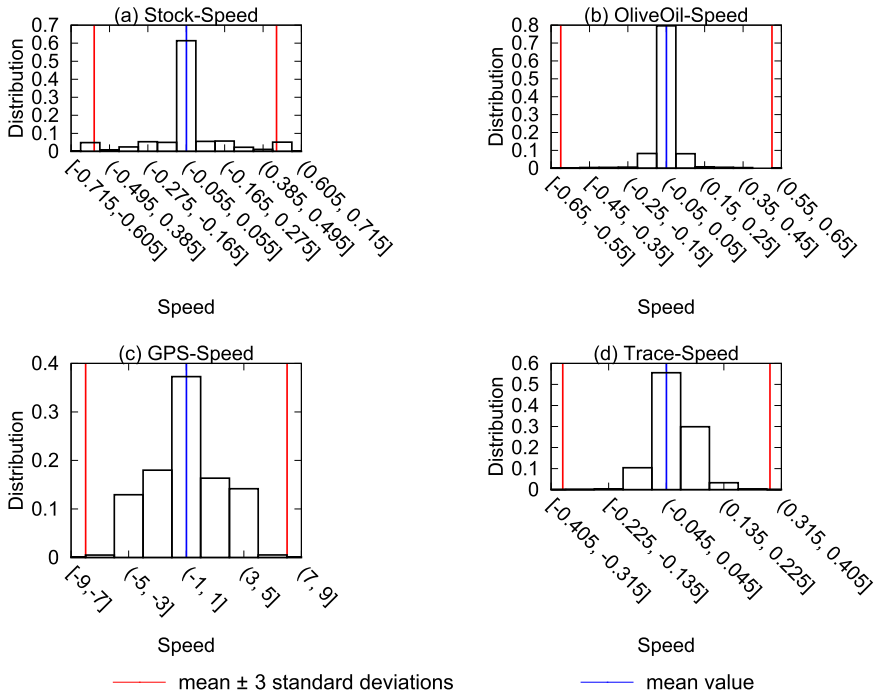


Fig. 22. Statistical distribution on speeds, (a) mean = 0.006, standard deviation = 0.167, (b) mean = -0.0006, standard deviation = 0.2, (c) mean = 0.003, standard deviation = 2.33, (d) mean = 0.004, standard deviation = 0.03.

for OliveOil, $a_{\max} = -a_{\min} = 6$ for GPS, $a_{\max} = -a_{\min} = 0.1$ for Trace, in Figure 25. As illustrated in Figures 23 and 25, given such speed and acceleration constraints, the L1 errors are much lower.

Generally, if the speed and acceleration constraints are set too loose, e.g., $s_{\max} = -s_{\min} = 10$ in Figure 23, or $a_{\max} = -a_{\min} = 10$ in Figure 25, then almost everything will pass the examination of speed and acceleration constraints without repairing, and thus the L1 error is high. However, if the speed and acceleration constraints are too tight, say, $s_{\max} = -s_{\min} = 0.1$ in Figure 23(a), $s_{\max} = -s_{\min} = 0.001$ in Figure 23(b) and (d), $s_{\max} = -s_{\min} = 3$ in Figure 23(c), or $a_{\max} = -a_{\min} = 0.001$ in Figures 25(a), (b), and (d), $a_{\max} = -a_{\min} = 0.3$ in Figure 25(c), then most values would be regarded as violations to such tight constraints. With over-repairing, the corresponding L1 error is high.

The experiments in Figures 23 and 25 consider various settings of constraints to illustrate the robustness. While the repairs are not reliable given incorrect constraints (too loose or too tight), the results are accurate under a wide range of speed and acceleration constraints. For instance, for the Stock dataset, a number of speed constraints $s_{\max} = -s_{\min}$ in the range of 0.4 to 0.9 achieve low repair error in Figure 23. Likewise, the acceleration constraints $a_{\max} = -a_{\min}$ between 0.1 and 1.0 lead to low repair error as well in Figure 25.

As aforesaid, we apply the rule of three standard deviations [26] to determine the constraints on speeds and accelerations (with confidence level 99.73% for normal distribution). For instance, given the mean 0 and the standard deviation 0.167 of Figure 22(a), we have the speed constraints $s_{\max} = -s_{\min} = 0.5$ for Stock. In this sense, we can avoid a grid search in Figures 23 and 25 to find the best constraints without under-repairing and over-repairing. Instead, with a confidence level 99.73% in the statistical distribution, we determine the constraints in Figures 22 and 24, which

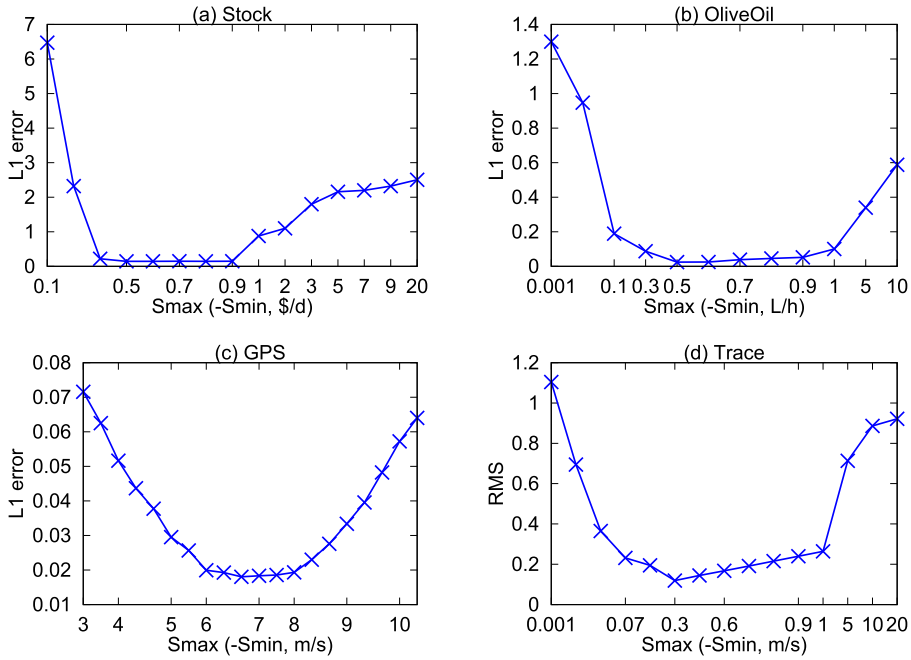


Fig. 23. Varying speed constraints in Local (Speed+Acceleration) over Stock with $a_{\max} = -a_{\min} = 0.4$, OliveOil with $a_{\max} = -a_{\min} = 0.1$, GPS with $a_{\max} = -a_{\min} = 6$, Trace with $a_{\max} = -a_{\min} = 0.1$.

would not be too loose (avoid under-repairing) or too tight (avoid over-repairing). Consequently, the aforesaid chosen $s_{\max} = -s_{\min} = 0.5$ for Stock shows a low repair error in Figure 23(a).

Figure 26 presents the results of dynamic constraints with OliveOil compared to static constraints in a streaming setting. As illustrated in Figure 26(c), both speed constraints $[s^k_{\min}, s^k_{\max}]$ and acceleration constraints $[a^k_{\min}, a^k_{\max}]$ change over time. It is not surprising that the repair performances of different algorithms in Figure 26(a) are more or less improved by the more accurate online adjusted constraints. The corresponding time costs in Figure 26(b) increase slightly for the extra cost of adjusting the constraints online.

Figure 27 presents the results of dynamic constraints over the Stock dataset, where the speed constraint in each day is dynamically determined by the price of the previous day, i.e., the daily increase or decrease of the price should not exceed 10%. Similar to the results with dynamic constraints over the OliveOil dataset in Figure 26, the results are also more or less improved by using dynamic constraints compared to those with static ones in the Stock data in Figure 27. The improvement is again not surprising, since the dynamic constraints capture more reasonable semantics in real world.

6 RELATED WORK

Noisy or dirty data streams have been highlighted. Techniques are developed to perform applications such as similarity matching queries directly over the noisy streams [23, 35]. However, if the accuracy of individual data points is concerned, e.g., the precision of RFID readings [17, 18], then repairing the individual data values is necessary. Matching learning techniques are employed to identify noises in data streams [38], but fail to repair the potentially dirty data.

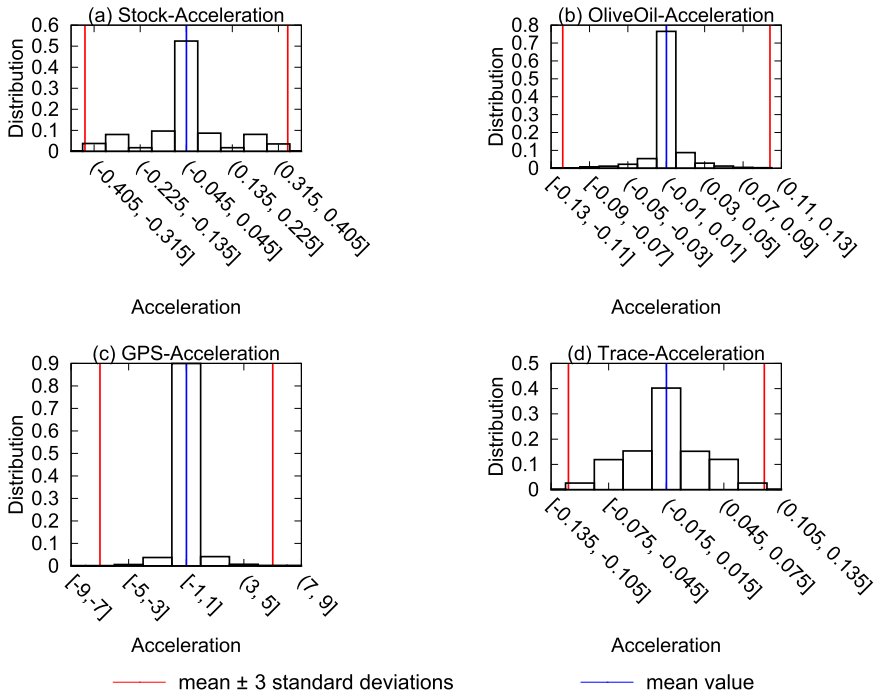


Fig. 24. Statistical distribution on accelerations, (a) mean = 0, standard deviation = 0.13, (b) mean = 0, standard deviation = 0.03, (c) mean = 0, standard deviation = 0.03, (d) mean = 0.0003, standard deviation = 2.

6.1 Smoother and Filter-based Data Cleaning

Smoother is often considered to reduce the affects of noises, e.g., for better visualization [29]. Moving average [6] is commonly used to smooth time series data and make forecasts. A **simple moving average (SMA)** is the unweighted mean of the last k data points. This average is used for forecasting the next value of the time series. Whereas in the simple moving average the past observations are weighted equally, a **weighted moving average (WMA)** multiplies factors to give different weights to data at different positions in the sample window, e.g., using the inverse value of time interval as the weight. Moreover, the **exponentially weighted moving average (EWMA)** [14] assigns exponentially decreasing weights over time.

Median Filter [33] is a kind of nonlinear signal processing technology that can effectively suppress noises based on the sorting statistics theory. The basic principle of Median Filter is to replace the value of a point in a sequence with the median value of its neighbor points, which can be used to eliminate isolated noise points. Instead of the values from neighbor points as the candidates, our median-based solution in Section 3.2 considers the candidates suggested by speed and acceleration constraints.

Savitzky-Golay Filter [1] is based on the average trend of High-quality **Normalized Difference Vegetation Index (NDVI)** time series to determine the appropriate filter parameters, using polynomial to achieve the ordinary least squares in sliding window. The most important feature of this filter is that it can keep the shape and width of the signal unchanged while filtering the noise. Using Savitzky-Golay method can improve the smoothness and reduce the noise interference.

Kalman Filter [27] uses linear system state equations to perform optimal estimation through observation data. It estimates the state of the dynamic system from a series of data with

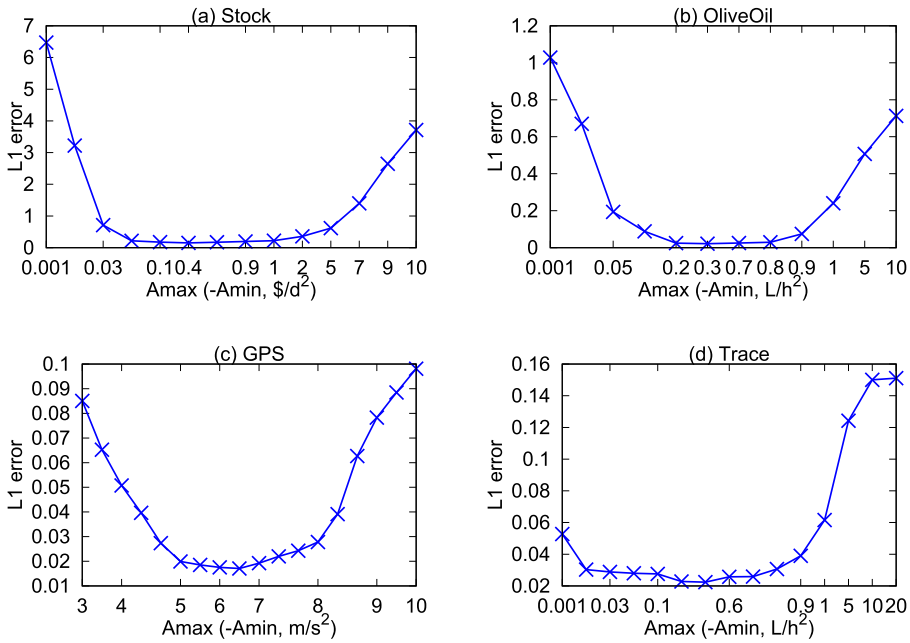


Fig. 25. Varying acceleration constraints in Local (Speed+Acceleration) over Stock with $s_{\max} = -s_{\min} = 0.5$, OliveOil with $s_{\max} = -s_{\min} = 0.6$, GPS with $s_{\max} = -s_{\min} = 7$, Trace with $s_{\max} = -s_{\min} = 0.3$.

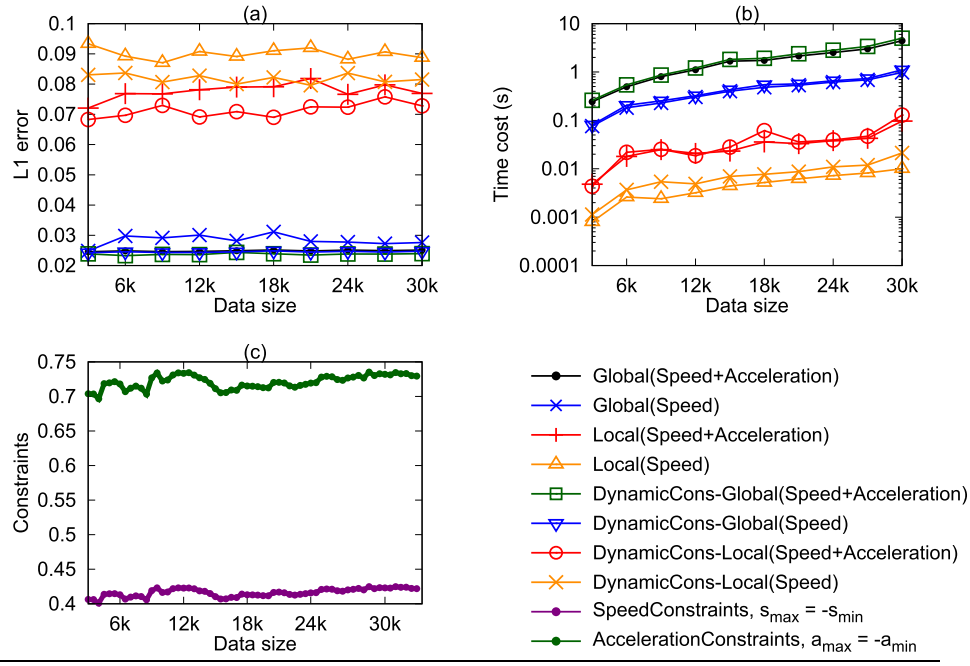
measurement noise when the measurement variance is known. The filter can be used to make estimation for data values, i.e., predicting according to the data points before the current one.

Kernel Smoother [9] is a statistical method used to estimate real-valued equations, which is actually a non-parametric regression. It refers to a statistical inference regression method that does not need to know the total distribution. Kernel Smoother is used as the weighted average of surrounding observation data. The weight is determined by the kernel, for example, the closer the data, the greater the weight. This simple method finds the structure of the data-set without applying parametric models.

Although the smoothing/filtering methods are very efficient, it is obvious to see that the smoother will modify a large number of data points. Therefore, as illustrated in the example of Figure 1, the major issue of smoother is the serious damage to the originally correct data points. One of our major contributions in this article is the employment of speed and acceleration constraints to supervise the more accurate cleaning. Following the minimum modification rule in constraint-based repairing, the original precise values are maximally preserved. The repair L1 error of our proposed method is much lower than those of smoother, as observed in our experimental evaluation.

Deshpande et al. [11] introduces a dynamic model to predict the current point according to the data points before it using Kalman Filter. The comparison to Kalman Filter is reported in Figures 14 to 21 and Table 2. Similarly, Considine et al. [10] approximate the network aggregation to handle dirty values. They generalize the well-known duplicate insensitive sketches to approximate counting processing. Instead of repairing data errors as in this study, the aggregation is approximated over the data with errors.

IMR [36] is an iterative minimum repair algorithm that combines the temporal features in anomaly detection with the minimum modification in data repair, i.e., performing the minimum repair



(a) L1 error	3420	6840	10260	13680	17100	20520	23940	27360	30780	34200
Global(Speed+Acc)	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025	0.025
Global(Speed)	0.025	0.03	0.029	0.03	0.028	0.031	0.028	0.028	0.027	0.028
Local(Speed+Acc)	0.072	0.077	0.077	0.078	0.079	0.079	0.082	0.077	0.08	0.077
Local(Speed)	0.093	0.089	0.087	0.091	0.089	0.091	0.092	0.088	0.091	0.089
Global(Speed+Acc)-D	0.024	0.023	0.024	0.024	0.024	0.024	0.023	0.024	0.024	0.024
Global(Speed)-D	0.024	0.025	0.024	0.024	0.025	0.025	0.024	0.025	0.024	0.025
Local(Speed+Acc)-D	0.068	0.07	0.073	0.069	0.071	0.069	0.072	0.072	0.076	0.073
Local(Speed)-D	0.083	0.084	0.081	0.083	0.08	0.082	0.08	0.084	0.081	0.082

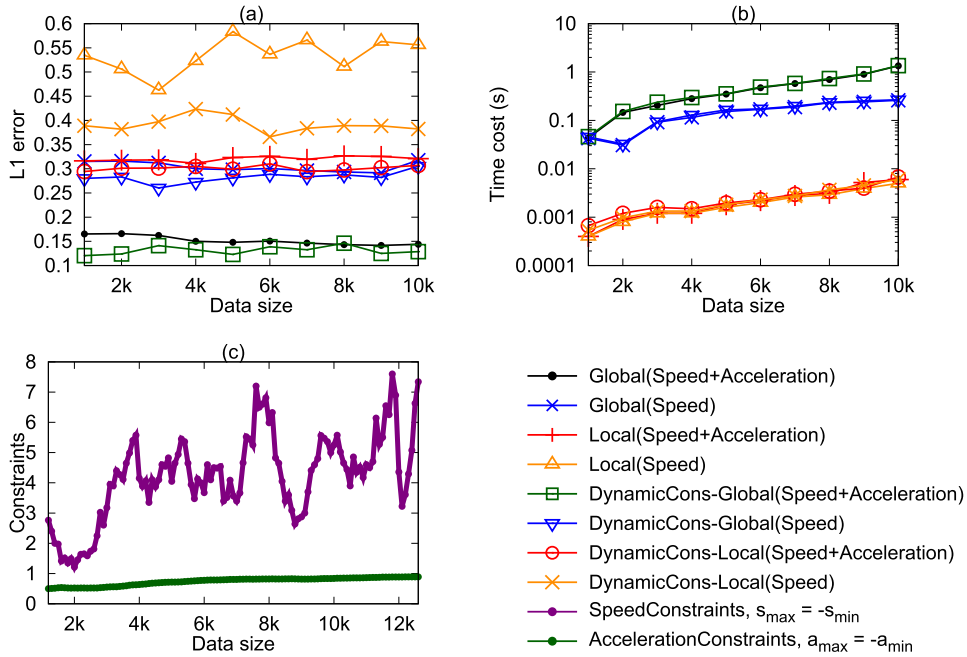
(b) Time cost	3420	6840	10260	13680	17100	20520	23940	27360	30780	34200
Global(Speed+Acc)	0.2432	0.4968	0.8022	1.111	1.673	1.733	2.151	2.53	3.012	4.4246
Global(Speed)	0.0762	0.183	0.2286	0.3042	0.401	0.4828	0.5328	0.6204	0.6868	0.9802
Local(Speed+Acc)	0.0048	0.018	0.0208	0.023	0.0232	0.0358	0.0373	0.0382	0.0426	0.0452
Local(Speed)	0.0008	0.0026	0.0027	0.0032	0.0044	0.0052	0.0062	0.0072	0.0082	0.01
Global(Speed+Acc)-D	0.2605	0.5419	0.8659	1.211	1.824	1.921	2.39	2.826	3.378	5.008
Global(Speed)-D	0.0805	0.2049	0.2538	0.3228	0.4291	0.5441	0.5686	0.6598	0.734	1.109
Local(Speed+Acc)-D	0.0053	0.0219	0.0252	0.0286	0.0281	0.0383	0.0398	0.0421	0.0472	0.0529
Local(Speed)-D	0.0011	0.0037	0.0054	0.0061	0.0069	0.0077	0.0087	0.0109	0.0119	0.0213

Fig. 26. Evaluation on dynamic constraints over OliveOil.

in each error prediction iteration. As a supervised method, we need to label the errors with the corresponding truths in a sequence. Online labeling is often unlikely in a streaming setting.

6.2 Constraint-based Data Repairing

Most data repairing techniques focus on the conventional integrity constraints, concerning equality relationships among tuples [4, 5]. The stream considered in our proposal consists of numerical



(a) L1 error	1282	2564	3847	5129	6412	7694	8976	10259	11541	12824
Global(Speed+Acc)	0.165	0.166	0.162	0.15	0.148	0.15	0.146	0.143	0.142	0.144
Global(Speed)	0.315	0.316	0.312	0.3	0.298	0.3	0.296	0.293	0.292	0.318
Local(Speed+Acc)	0.316	0.318	0.319	0.311	0.323	0.326	0.32	0.327	0.325	0.321
Local(Speed)	0.535	0.506	0.463	0.523	0.583	0.537	0.566	0.511	0.563	0.556
Global(Speed+Acc)-D	0.12	0.124	0.141	0.133	0.123	0.139	0.133	0.146	0.125	0.129
Global(Speed)-D	0.28	0.283	0.26	0.272	0.281	0.288	0.283	0.287	0.282	0.305
Local(Speed+Acc)-D	0.294	0.301	0.301	0.305	0.299	0.31	0.294	0.298	0.302	0.307
Local(Speed)-D	0.389	0.382	0.397	0.423	0.412	0.366	0.384	0.389	0.389	0.382

(b) Time cost	1282	2564	3847	5129	6412	7694	8976	10259	11541	12824
Global(Speed+Acc)	43.6	146.4	203.6	281.6	349.8	472	578.2	699	896.4	1343
Global(Speed)	42.8	30.8	91.8	115.4	152.2	169.4	188.6	231.4	242.2	259.6
Local(Speed+Acc)	0.4	0.9	1.2	1.2	1.8	2.2	2.8	3.1	5.2	6
Local(Speed)	0.4	0.8	1.2	1.2	1.6	2	2.6	2.9	3.9	5
Global(Speed+Acc)-D	46.27	152.9	237.3	297.3	353.9	489.1	581.3	735	912.3	1351
Global(Speed)-D	45.33	32.6	93.93	128.8	164.6	173.1	197.9	235.7	253.1	267.1
Local(Speed+Acc)-D	0.667	1.2	1.587	1.493	1.967	2.287	2.96	3.46	3.993	6.767
Local(Speed)-D	0.533	0.967	1.333	1.333	1.8	2.3	2.733	3.667	4.533	6.067

Fig. 27. Evaluation on dynamic constraints over Stock.

values, where the constraints in relational settings such as functional dependencies are not applicable. The temporal functional dependency [2] extending functional dependencies with temporal information is still not applicable to our study concerning numerical values. Moreover, since the constraints apply to any pair of tuples in a relation, the repairing problem (with the minimum modification) is often found to be NP-hard [21, 24]. In contrast, as stated in Section 2.1, the (speed and acceleration) constraints on data streams are usually valid to data points in a short period (window). The corresponding repairing could also be efficient (Corollary 2.3).

To the best of our knowledge, Holistic cleaning [8] is the only existing constraint-based technique that can support speed and acceleration constraints (expressed by extended denial constraints). It is worth noting the original **denial constraints (DCs)** [25] supports only six operators, $=, \neq, <, >, \leq, \geq$. To express speed and acceleration constraints in Formulas (1) and (2), respectively, we extend denial constraints, i.e., $\forall i, j, \neg(\frac{x_j - x_i}{t_j - t_i} \geq s_{\min} \wedge \frac{x_j - x_i}{t_j - t_i} \leq s_{\max} \wedge t_j - t_i > 0 \wedge t_j - t_i \leq w)$ as speed constraints and $\forall i, j, \neg(\frac{\frac{x_j - x_i}{t_j - t_i} - \frac{x_i - x_{i-1}}{t_i - t_{i-1}}}{t_j - t_i} \geq a_{\min} \wedge \frac{\frac{x_j - x_i}{t_j - t_i} - \frac{x_i - x_{i-1}}{t_i - t_{i-1}}}{t_j - t_i} \leq a_{\max} \wedge t_j - t_i > 0 \wedge t_j - t_i \leq w)$ as acceleration constraints. The implementation of the Holistic method is adapted to support the aforesaid extended denial constraints. Since the approach is proposed for repairing the general (tabular) data, it cannot support the online/integral cleaning over sliding windows in streaming data. In this sense, one of our contributions in this study is the local optimum method, which supports not only online cleaning but also out-of-order data arrival. Consequently, as illustrated in the experiments, our proposal can show up to two orders of magnitude improvement in time costs compared with Holistic cleaning.

HoloClean [28] is a semi-automatic data repair framework that relies on statistical learning and inference to unify a series of data repair methods for repairing errors in structured data. Based on the weak supervised paradigm, HoloClean uses a variety of signals, including user-defined heuristic rules (such as general data integrity constraints) and external dictionaries to repair the wrong data. Similar to the Holistic cleaning [8], we may use the (extended) **denial constraints (DCs)** to specify the constraints on speeds.

Moreover, **Sequential Dependency (SD)** [15] cannot express precisely the speed constraints. SDs concern the difference of two consecutive data points in a sequence. As discussed, data streams often deliver data points in various time intervals. Given different timestamp distances, the value difference of two consecutive points does not exactly denote the speed semantics. Owing to such imprecise constraint knowledge, as presented in the experiments, the L1 error of SD based repair could be much higher compared to our speed/acceleration constraint-based proposal. Our another contribution is the employment of the more accurate speed and acceleration rather than the simple value distance in repairing streaming data.

The preliminary conference version of this article [32] focuses on cleaning the dirty stream data under speed constraints. It considers the constraints on the speed of data changes, such as fuel consumption per hour, weekly temperature variation, or daily limit of stock prices. For example, the fuel consumption of a crane should not be negative and not exceed 40 liters per hour. The cleaning problem is thus to repair the data to meet the constraints of the minimum and maximum speeds. In this journal version, along with the speed constraints, we further consider the constraints on acceleration of value changes. For instance, consider the trajectory of a van. The speed constraints state that the GPS value change of two points should not exceed 100 km/h, while the acceleration constraints further require that the difference on speeds between two consecutive points in a second is no greater than 10 km/h. That is, the increase/decrease of speeds in a second is impossible to be greater than 10 km/h. The repairing needs to satisfy the constraints on the maximum and minimum speeds as well as the maximum and minimum accelerations. Since more constraints are utilized, the corresponding repair will be more accurate than considering the speed constraints solely, as illustrated in the experiments in Section 5.

Besides our studied speed and acceleration constraints, Fischer et al. [12] proposed a nice notation, Stream Schema, for representing structural and semantic constraints on data streams. The Stream Schema concerns general constraints with various semantics such as orderings between attribute values, while our study focuses only on the specific speed constraints over numeric values. As a promising future direction, it is interesting to extend the stream data cleaning w.r.t. the more general Stream Schema constraints.

7 CONCLUSIONS

In this study, we first indicate the inappropriateness of the smoothing-based stream data cleaning. It could not repair the dirty data such as large spikes, and even worse may seriously damage the originally accurate values. Following the same line of employing integrity constraints for relational data cleaning, in this article, we propose a repair method with acceleration and speed constraints. The repairing of imprecise data is guided by the innovative constraints on speed and acceleration. The speed and acceleration constraint semantics could be easily captured, such as daily price limit in financial markets, or the maximum walking speed and acceleration of a person. With speed and acceleration constraints, our method supports online streaming cleaning in linear time. In particular, the *median-based solution* can fast identify the local optimum under certain constraints, following the intuition that a solution with the minimum distance (i.e., as close as possible to each point) probably lies in the middle of the data points. Experiments on real datasets demonstrate that our method with speed and acceleration constraints can show significantly lower L1 error than the smoothing-based approach and up to two orders of magnitude improvement in time performance compared to the state-of-the-art data cleaning methods.

REFERENCES

[1] A. Savitzky A. and M. J. E. Golay. 1964. Smoothing and differentiation of data by simplified least-squares procedures. *Analyt. Chem.* 8, 36 (1964), 1627–1639. DOI: http://dx.doi.org/10.1021/ac60214a047

[2] Ziawasch Abedjan, Cuneyt Gurcan Akcora, Mourad Ouzzani, Paolo Papotti, and Michael Stonebraker. 2015. Temporal rules discovery for web data cleaning. *PVLDB* 9, 4 (2015), 336–347. DOI: https://doi.org/10.14778/2856318.2856328

[3] Patricia C. Arocena, Boris Glavic, Giansalvatore Mecca, Renée J. Miller, Paolo Papotti, and Donatello Santoro. 2015. Messing up with BART: Error generation for evaluating data-cleaning algorithms. *PVLDB* 9, 2 (2015), 36–47. DOI: https://doi.org/10.14778/2850578.2850579

[4] George Beskales, Ihab F. Ilyas, and Lukasz Golab. 2010. Sampling the repairs of functional dependency violations under hard constraints. *PVLDB* 3, 1 (2010), 197–207. DOI: https://doi.org/10.14778/1920841.1920870

[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 *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 143–154. DOI: https://doi.org/10.1145/1066157.1066175

[6] David R. Brillinger. 2001. *Time Series - Data Analysis and Theory. (Classics in Applied Mathematics*, Vol. 36.) SIAM.

[7] Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 785–794. DOI: https://doi.org/10.1145/2939672.2939785

[8] Xu Chu, Ihab F. Ilyas, and Paolo Papotti. 2013. Holistic data cleaning: Putting violations into context. In *Proceedings of the 29th IEEE International Conference on Data Engineering*. 458–469. DOI: https://doi.org/10.1109/ICDE.2013.6544847

[9] Moo K. Chung. 2020. Gaussian kernel smoothing. *CoRR* abs/2007.09539 (2020).

[10] Jeffrey Considine, Feifei Li, George Kollios, and John W. Byers. 2004. Approximate aggregation techniques for sensor databases. In *Proceedings of the 20th International Conference on Data Engineering*. 449–460. DOI: https://doi.org/10.1109/ICDE.2004.1320018

[11] Amol Deshpande, Carlos Guestrin, Samuel Madden, Joseph M. Hellerstein, and Wei Hong. 2004. Model-driven data acquisition in sensor networks. In *(e)Proceedings of the 30th International Conference on Very Large Data Bases*. 588–599. DOI: https://doi.org/10.1016/B978-012088469-8.50053-X

[12] Peter M. Fischer, Kyumars Sheykh Esmaili, and Renée J. Miller. 2010. Stream schema: Providing and exploiting static metadata for data stream processing. In *Proceedings of the 13th International Conference on Extending Database Technology*. 207–218. DOI: https://doi.org/10.1145/1739041.1739068

[13] David Freedman. 1991. *Statistics* (2nd ed.). Norton.

[14] Roland Fried and Ann Cathrice George. 2011. Exponential and holt-winters smoothing. In *International Encyclopedia of Statistical Science*. 488–490. DOI: https://doi.org/10.1007/978-3-642-04898-2_244

[15] Lukasz Golab, Howard J. Karloff, Flip Korn, Avishek Saha, and Divesh Srivastava. 2009. Sequential dependencies. *PVLDB* 2, 1 (2009), 574–585. DOI: https://doi.org/10.14778/1687627.1687693

[16] C. A. R. Hoare. 1962. Quicksort. *Comput. J.* 5, 1 (1962), 10–15. DOI: https://doi.org/10.1093/comjnl/5.1.10

[17] Shawn R. Jeffery, Gustavo Alonso, Michael J. Franklin, Wei Hong, and Jennifer Widom. 2006. Declarative support for sensor data cleaning. In *Proceedings of the 4th International Conference on Pervasive Computing*. 83–100. DOI: https://doi.org/10.1007/11748625_6

[18] Shawn R. Jeffery, Gustavo Alonso, Michael J. Franklin, Wei Hong, and Jennifer Widom. 2006. A pipelined framework for online cleaning of sensor data streams. In *Proceedings of the 22nd International Conference on Data Engineering*. 140. DOI: https://doi.org/10.1109/ICDE.2006.8

[19] Shawn R. Jeffery, Minos N. Garofalakis, and Michael J. Franklin. 2006. Adaptive cleaning for RFID data streams. In *Proceedings of the 32nd International Conference on Very Large Data Bases*. 163–174. Retrieved from http://dl.acm.org/citation.cfm?id=1164143.

[20] Narendra Karmarkar. 1984. A new polynomial-time algorithm for linear programming. In *Proceedings of the 16th ACM Symposium on Theory of Computing*. 302–311. DOI: https://doi.org/10.1145/800057.808695

[21] Solmaz Kolahi and Laks V. S. Lakshmanan. 2009. On approximating optimum repairs for functional dependency violations. In *Proceedings of the 12th International Conference on Database Theory*. 53–62. DOI: https://doi.org/10.1145/1514894.1514901

[22] Xian Li, Xin Luna Dong, Kenneth Lyons, Weiyi Meng, and Divesh Srivastava. 2012. Truth finding on the deep web: Is the problem solved? *PVLDB* 6, 2 (2012), 97–108. DOI: https://doi.org/10.14778/2535568.2448943

[23] Zheng Li, Tingjian Ge, and Cindy X. Chen. 2013. ϵ-Matching: Event processing over noisy sequences in real time. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 601–612. DOI: https://doi.org/10.1145/2463676.2463715

[24] Ester Livshits, Benny Kimelfeld, and Sudeepa Roy. 2018. Computing optimal repairs for functional dependencies. In *Proceedings of the 37th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*. 225–237. DOI: https://doi.org/10.1145/3196959.3196980

[25] Andrei Lopatenko and Loreto Bravo. 2007. Efficient approximation algorithms for repairing inconsistent databases. In *Proceedings of the 23rd International Conference on Data Engineering*. 216–225. DOI: https://doi.org/10.1109/ICDE.2007.367867

[26] Sekander Hayat Khan M. 2011. Standard deviation. In *International Encyclopedia of Statistical Science*. 1378–1379. DOI: https://doi.org/10.1007/978-3-642-04898-2_535

[27] A. K. Mahalanabis. 1986. Introduction to random signal analysis and Kalman filtering: Robert G. Brown. *Autom.* 22, 3 (1986), 387–388. DOI: https://doi.org/10.1016/0005-1098(86)90041-5

[28] Theodoros Rekatsinas, Xu Chu, Ihab F. Ilyas, and Christopher Ré. 2017. HoloClean: Holistic data repairs with probabilistic inference. *Proc. VLDB Endow.* 10, 11 (2017), 1190–1201. DOI: https://doi.org/10.14778/3137628.3137631

[29] Kexin Rong and Peter Bailis. 2017. ASAP: Prioritizing attention via time series smoothing. *PVLDB* 10, 11 (2017), 1358–1369. DOI: https://doi.org/10.14778/3137628.3137645

[30] Claude Sammut and Geoffrey I. Webb (Eds.). 2017. *Encyclopedia of Machine Learning and Data Mining*. Springer. DOI: https://doi.org/10.1007/978-1-4899-7687-1

[31] Michael Smithson. 2011. Confidence interval. In *International Encyclopedia of Statistical Science*. 283–284. DOI: https://doi.org/10.1007/978-3-642-04898-2_183

[32] Shaoxu Song, Aoqian Zhang, Jianmin Wang, and Philip S. Yu. 2015. SCREEN: Stream data cleaning under speed constraints. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*. 827–841. DOI: https://doi.org/10.1145/2723372.2723730

[33] John W. Tukey. 1977. *Exploratory Data Analysis*. Addison-Wesley. Retrieved from https://www.worldcat.org/oclc/03058187.

[34] Tzu-Tsung Wong and Nai-Yu Yang. 2017. Dependency analysis of accuracy estimates in k-fold cross validation. *IEEE Trans. Knowl. Data Eng.* 29, 11 (2017), 2417–2427. DOI: https://doi.org/10.1109/TKDE.2017.2740926

[35] Wush Chi-Hsuan Wu, Mi-Yen Yeh, and Jian Pei. 2012. Random error reduction in similarity search on time series: A statistical approach. In *Proceedings of the IEEE 28th International Conference on Data Engineering*. 858–869. DOI: https://doi.org/10.1109/ICDE.2012.83

[36] Aoqian Zhang, Shaoxu Song, Jianmin Wang, and Philip S. Yu. 2017. Time series data cleaning: From anomaly detection to anomaly repairing. *Proc. VLDB Endow.* 10, 10 (2017), 1046–1057. DOI: https://doi.org/10.14778/3115404.3115410

[37] Shichao Zhang, Xuelong Li, Ming Zong, Xiaofeng Zhu, and Debo Cheng. 2017. Learning k for kNN classification. *ACM Trans. Intell. Syst. Technol.* 8, 3 (2017), 43:1–43:19. DOI: https://doi.org/10.1145/2990508

[38] Xingquan Zhu, Peng Zhang, Xindong Wu, Dan He, Chengqi Zhang, and Yong Shi. 2008. Cleansing noisy data streams. In *Proceedings of the 8th IEEE International Conference on Data Mining*. 1139–1144. DOI: https://doi.org/10.1109/ICDM.2008.45

Received February 2020; revised April 2021; accepted May 2021