# Sequential Data Cleaning: A Statistical Approach

Aoqian Zhang Shaoxu Song Jianmin Wang
Tsinghua National Laboratory for Information Science and Technology
KLiss, MoE; School of Software, Tsinghua University, China
{zaq13, sxsong, jimwang}@tsinghua.edu.cn

## ABSTRACT

Errors are prevalent in data sequences, such as GPS trajectories or sensor readings. Existing methods on cleaning sequential data employ a constraint on value changing speeds and perform constraint-based repairing. While such speed constraints are effective in identifying large spike errors, the small errors that do not significantly deviate from the truth and indeed satisfy the speed constraints can hardly be identified and repaired. To handle such small errors, in this paper, we propose a statistical based cleaning method. Rather than declaring a broad constraint of max/min speeds, we model the probability distribution of speed changes. The repairing problem is thus to maximize the likelihood of the sequence w.r.t. the probability of speed changes. We formalize the likelihood-based cleaning problem, show its NP-hardness, devise exact algorithms, and propose several approximate/heuristic methods to trade off effectiveness for efficiency. Experiments on real data sets (in various applications) demonstrate the superiority of our proposal.

## 1. INTRODUCTION

Data sequences are often found with dirty or imprecise values, such as GPS trajectories or sensor reading sequences [8]. According to the survey [10], even the data of stock prices could be dirty. 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 below.)

To clean dirty data, constraint-based repairing is often employed [1]. Existing study [14] on sequential data cleaning considers the constraints on speeds of value changes, namely speed constraints. For example, the speed constraints on fuel meter values state that the fuel consumption of a crane should not be negative and not exceed 40 liters per hour. Constraint-based cleaning identifies the violations to such speed constraints and (minimally) modify the values so that the repaired results satisfy the specified speed constraints.

While speed constraints can successfully identify large spike errors (see examples below), the constraint-based cleaning repairs the dirty point to a value w.r.t. maximum/minimum speeds. As indicated in [14], the repair strategy of adjusting the value to the maximum/minimum allowed is made referring to the minimum change principle in data repairing [1]. The rationale is that people or systems try to minimize mistakes in practice. This minimum change principle is widely considered in repairing relational data [1, 3]. As the first application of the minimum change principle to sequential data [14], it unfortunately leads to the aforesaid repair strategy of choosing the maximum/minimum allowable values.

An alternative approach is to consider the average of the previous values as a repair, a.k.a. smoothing methods [2, 6]. For example, the simple moving average (SMA) [2] smooths time series data by computing the unweighted mean of the last $k$ points. Instead of weighting equally, the exponentially weighted moving average (EWMA) [6] assigns exponentially decreasing weights over time. As indicated in [14], the problem of the smoothing methods is over-repairing, i.e., almost all the data points are modified, most of which are indeed correct originally and do not need repair.

Moreover, the speed constraints fail to identify the small errors that do not significantly deviate from the original values, and indeed satisfy the speed constraints. The small errors are particularly important in some applications. For instance, a deviation of 1m in GPS readings is prevalent and small relative to 10m large spikes. Such a small error (1m), however, is critical in car localization for automatic driving. Moreover, aggregating a large number of small errors, data mining results could be seriously misled, e.g., unable to form meaningful clusters over imprecise GPS readings with many small errors [13]. Our results in Section 6.3 also show that repairing small errors could improve the accuracy of prediction application.

Instead of considering the aforesaid max/min speeds, in this paper, we propose a novel statistical-based cleaning by introducing the likelihoods w.r.t. various speeds. Let us first illustrate a motivation example below.

**Example 1.** *Figure 1 presents a sequence of stock prices. Two dirty values appear at time point 1547 and 1569, respectively, in the observed sequence (in black, the corresponding true values are presented in blue).*

*Existing speed constraint-based cleaning (SCREEN) [14] employs the maximum and minimum speed constraints, denoting the largest rates of allowed (stock price) value increase and decrease, respectively.*

*1) A violation to the speed constraints is detected at time 1547, whose value decrease exceeds the minimum speed constraint. The value of time point 1547 is thus repaired to a*

*SIGMOD'16, June 26-July 01, 2016, San Francisco, CA, USA*
© 2016 ACM. ISBN 978-1-4503-3531-7/16/06…$15.00
DOI: http://dx.doi.org/10.1145/2882903.2915233

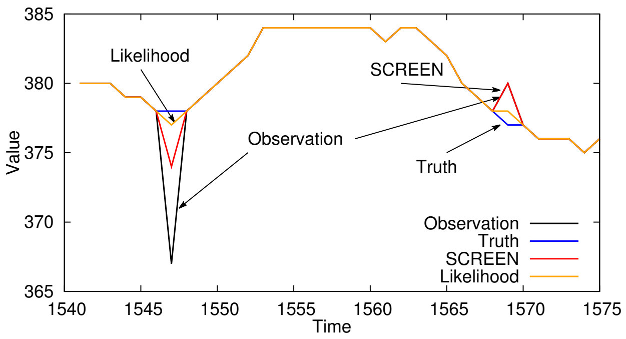

**Figure 1: Speed constraint-based cleaning (SCREEN) can identify the large spike error but repair it w.r.t. the max/min speed constraints. The small error that does not significantly deviate the original values cannot be identified.**

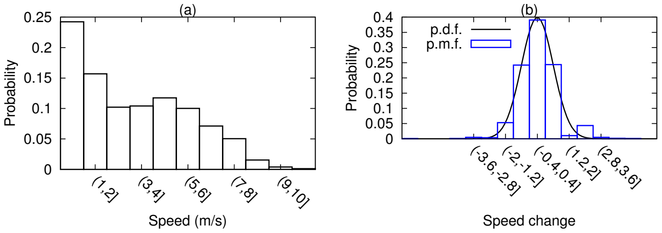

**Figure 2: Probability distributions of (a) speeds and (b) speed changes, on a real GPS dataset presented in Section 6**

*new value w.r.t. this minimum speed constraint. Since the max/min speeds may not be reached all the time in practice, the repairs w.r.t. the max/min speed constraints could be away from the truth.*

*2) Moreover, the small error at time point 1569 indeed satisfies the max/min speed constraints, and thus is not detected as violations and not repaired.*

*By considering likelihoods w.r.t. various speeds (see detailed definition and discussion below), our proposed method computes a most likely repair , instead of a repair restricted to the max/min speed constraints.*

Defining the likelihood of a repair w.r.t. various speeds is non-trivial. It is not rational by simply measuring the probabilities of speeds. The reason is that max/min speeds may also appear with considerably high probability. For example, stock prices increase in daily limit (max speed) in a period, or a car drives in its max speed in highway. As shown in Figure 2(a), the speed probabilities are almost equal, e.g., from 3 to 7. The probabilities of speeds may not be able to distinguish the likelihoods of various repairs.

Intuitively, while speeds capture the *change of values* in consecutive data points, we may also consider the relationships of speeds between data points in a sequence, i.e., modeling the *change of speeds*. The rationale behind is that the change of speeds in consecutive data points (roughly interpreted as acceleration) should not be significant. As shown in Figure 2(b), 90% speed changes are within [-1.2, 1.2].

Enlightened by the aforesaid discipline of non-significant speed changes, we employ the probability distribution of speed changes, and calculate the likelihood of a sequence w.r.t. the speed changes (see example in Figure 4 in Example 2 below). The cleaning problem is thus to find a repaired sequence with larger speed change likelihood.

### *Contributions*

Our major contributions in this paper are summarized as:

1) We formalize the problem of repairing sequential data with the maximum likelihood (Problem 1), show its NP-hardness (Theorem 1), and introduce a pseudo-polynomial time algorithm for computing the optimal solution (Proposition 2). Efficient pruning is also devised.

2) We devise a quadratic-time constant-factor approximation algorithm (Proposition 5), again together with efficient pruning. To further accelerate the computing, a linear-time heuristic is proposed (Proposition 4).

3) We approximate the discrete probability distribution of speed changes by a continuous probability distribution, to support even faster computing. With a proper continuous probability distribution, we show that the maximum likelihood repairing problem is indeed transformed to a quadratic programming problem (Proposition 8). Efficient solvers and simple fast greedy heuristics are directly applicable.

4) We report an extensive experimental evaluation on three real datasets in different scenarios: i) To evaluate the performance over various errors, the first dataset STOCK is originally clean and injected with various errors. ii) To evaluate real errors, the second dataset collects GPS trajectories with errors naturally embedded, and the corresponding truth manually labelled. iii) To apply the methods in practice, the third experiment performs on a real dataset ENGINE with both unknown errors and unknown truth. The performance is evaluated on an application over the data with and without repairing. The results demonstrate that our proposal achieves better performance in both repair accuracy and application accuracy.

The remainder of this paper is organized as follows. We first introduce the preliminaries and problem statement in Section 2. The exact and approximate solutions are then presented in Sections 3 and 4, respectively. Section 5 develops the repairing over the continuous probability distribution. The experimental evaluation is reported in Section 6. Finally, we discuss the related studies in Section 7, and conclude the paper in Section 8. Table 3 in the Appendix lists the notations frequently used in this paper.

## 2. PROBLEM STATEMENT

### 2.1 Preliminaries

Consider a sequence $x = x[1], x[2], \ldots$, where each $x[i]$ is the value of the $i$-th data point from a finite domain. For brevity, we write $x[i]$ as $x_i$, and $x_{i\ldots j}$ denoting the subsequence $x_i, x_{i+1}, \ldots x_j$ of $x$.

Each $x_i$ is associated with a timestamp $t_i$, and an error range $\theta_i$. The error range, e.g., specified by engineering tolerance, denotes that the true value $x'_i$ of $i$-th data point may be in the range of $[x_i - \theta_i, x_i + \theta_i]$, denoted by $x'_i \in [x_i \pm \theta_i]$. While some data sequences may have individual $\theta_i$ for each data point $i$, e.g., indicated as "accuracy" in GPS readings, others may specify a single $\theta_{\max}$ denoting the maximum error range for all the data points in the sequence, such as in sensor readings.

Referring to [14], the speed is defined on the *change of* *value*, e.g., $v_{i-1,i} = \frac{x_i - x_{i-1}}{t_i - t_{i-1}}$ from data point $i-1$ to $i$. Let

$$u_i = v_{i,i+1} - v_{i-1,i} = \frac{x_{i+1} - x_i}{t_{i+1} - t_i} - \frac{x_i - x_{i-1}}{t_i - t_{i-1}} \tag{1}$$

denote the *change of speed* before and after the $i$-th point.

The likelihood $L(x)$ of a sequence $x$ w.r.t. speed changes is

$$L(x) = \sum_{i=2}^{n-1} L(u_i) = \sum_{i=2}^{n-1} \log P(u_i), \tag{2}$$

where $P(u_i)$ denotes the probability of speed change $u_i$, and $L(u_i)$ denotes the corresponding (log) likelihood. The empirical probability distribution $P$ on speed changes can be estimated simply by statistics on the sequence.

**Example 2** (Probability distribution and likelihood computation)**.** *Consider a sequence* $x = \{11, 12, 15, 14, 15, 15, 17\}$, *with timestamps* $t = \{1, 2, 3, 4, 5, 6, 7\}$. *Figure 3(a) illustrates the data points (in black) and Figure 4 shows the corresponding probability distribution of speed changes.*

*The probability of speed change on the 3rd point* ($x_3$) *is*

$$P(u_3) = P(\frac{14-15}{4-3} - \frac{15-12}{3-2}) = P(-4) = 0.1,$$

*with likelihood* $L(u_3) = \log(0.1) = -1.2$. *By similarly computing the likelihoods on other data points, we have the likelihood of the sequence* $x$, *i.e.,* $L(x) = \log(0.25) + \log(0.1) + \log(0.25) + \log(0.2) + \log(0.25) = -8.1$.

*Indeed, the 3rd data point is dirty with error value* $x_3 = 15$. *The corresponding truth value should be* $x'_3 = 13$ *instead. With this truth value, the likelihood is* $L(x') = 3 * \log(0.3) + \log(0.2) + \log(0.25) = -6.6$, *which is higher than* $L(x)$ *of the aforesaid* $x$ *with dirty value.*

Referring to formula (2), the likelihood $L(x)$ is computed by the summation over the probabilities of speed changes $P(u_i)$. Consider the probability distribution of speed changes in Figure 4 for Example 2. The maximum likelihood will be reached when the probability of speed change $P(u_i)$ is maximized for each $u_i$, i.e., having speed change $u_i \in (-1, 1]$ with $P(u_i) = 0.3$. Consequently, the maximum likelihood is $L(x^*) = 5 * \log(0.3) \approx -6.0$.

## 2.2 Repair Problem

Following the intuition (presented in Figure 2 in the introduction) that speeds should not change significantly before and after a data point, we propose to find a repaired sequence $x'$ of $x$ such that the likelihood w.r.t. speed changes increases. On the other hand, referring to the minimum change principle in data repairing [1], a repaired sequence $x'$ close to $x$ is preferred. Therefore, as in [14], we also consider the repair cost from $x$ to $x'$

$$\Delta(x, x') = \sum_{i=1}^{n} |x'_i - x_i|.$$

Following the same line of maximal likelihood repairing over relational data [15], we describe the problem of likelihood-based repairing over sequential data as follows. (See Section 7 for a discussion on the difference between likelihood-based relational and sequential data repairing.)

**Problem 1.** *Given a finite sequence* $x$ *of* $n$ *data points and a repair cost budget* $\delta$, *the* maximum likelihood repair *problem is to find a repair* $x'$ *such that* $\Delta(x, x') \le \delta$ *and the likelihood* $L(x')$ *is maximized.*

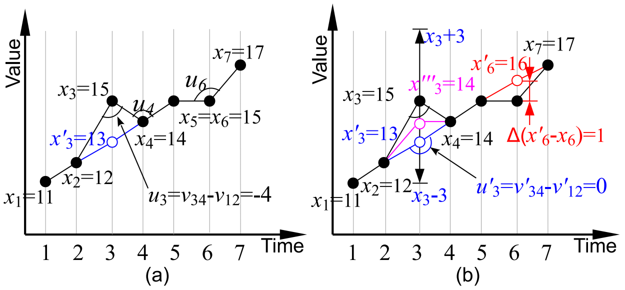

**Figure 3: Possible repairs of an example sequence**

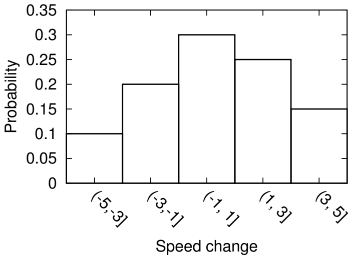

**Figure 4: Probability distribution of speed changes**

It is worth noting that the bound $\delta$ on repair cost is often necessary to avoid over-repaired results (see example below). As illustrated in Section 6, such a budget threshold could be practically determined by observing the likelihoods of returned repair results.

We formalize the repair problem as follows.

$$\begin{aligned} \max \quad & \sum_{i=2}^{n-1} \log P\left(\frac{x'_{i+1} - x'_i}{t_{i+1} - t_i} - \frac{x'_i - x'_{i-1}}{t_i - t_{i-1}}\right) && (3) \\ \text{s.t.} \quad & \sum_{i=1}^{n} |x'_i - x_i| \le \delta \\ & x'_i \in [x_i \pm \theta_i] && 1 \le i \le n \end{aligned}$$

**Example 3** (Repair and cost)**.** *Consider the sequence in Figure 1, a real segment from the STOCK*[1] *dataset, with probability distribution on speed changes in Figure 9(a). The error range is* $\theta_i = 12$ *for all the data points* $i$. *That is, a value in the range of* $[x_i - 12, x_i + 12]$, *simply denoted by* $[x_i \pm 12]$, *could be considered as a repair* $x'_i$ *of* $x_i$.

*Given a small repair cost budget, e.g.,* $\delta = 5$, *referring to the optimization problem in formula (3), a repair* $x'$ *will be returned, with four points changed at time 1553, 1561, 1569 and 1573, as shown in Figure 5. It has repair cost* $\Delta(x, x') = |x_{1553} - x'_{1553}| + |x_{1561} - x'_{1561}| + |x_{1569} - x'_{1569}| + |x_{1573} - x'_{1573}| = 5 \le \delta$, *with the maximized likelihood* $L(x') = -50.1$. *It is notable that the large spike at time 1547 could not be repaired under this small budget.*

*On the other hand, if the repair cost budget is too large, e.g.,* $\delta = 35$, *a repair* $x'$ *with a large number of modified points is returned. The corresponding repair cost is* $\Delta(x, x') = 33 \le \delta$, *with the maximized likelihood* $L(x') = -27.0$.

*Intuitively, we would consider a "proper" setting of repair cost budget, e.g.,* $\delta = 15$, *which is neither too small*

[1] http://finance.yahoo.com/q/hp?s=AIP.L+Historical+Prices

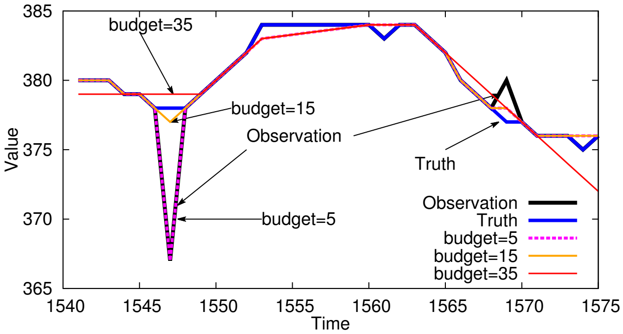

**Figure 5: Example repairs with various repair cost budgets $\delta$**

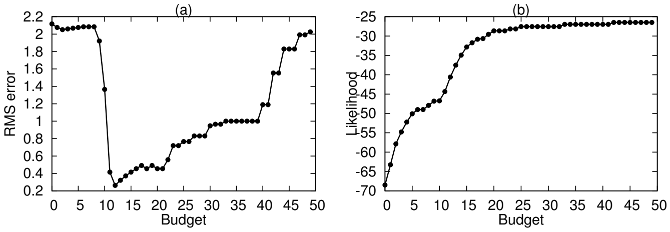

**Figure 6: Repair result (a) error and (b) likelihood under various repair cost budgets $\delta$**

*($\delta = 5$ with data points barely repaired and low returned likelihood) nor too large ($\delta = 35$ with data points over repaired but no significant likelihood improvement). Following this guideline, a good $\delta$ could be practically chosen by observing the repair likelihoods returned together with the repair results. For instance, the likelihood does not significantly increase by setting $\delta$ from 15 to 20 in Figure 6(b). That is, with $\delta$ in the range from 15 to 20, the data is neither insufficiently repaired (with a low likelihood) nor overrepaired (with too large repair cost budget but no much likelihood gain). Thereby, the corresponding repair result error (see Section 6.1 for formal definition) is low in Figure 6(a). (Similar results are also observed in other real datasets in Section 6, which verify again the guideline on selecting a proper repair cost budget $\delta$.)*

# 3. EXACT SOLUTION

In this section, we first prove the NP-hardness of the maximum likelihood repair problem, which motivates us to devise a pseudo-polynomial time algorithm based on dynamic programming in Section 3.2.

## 3.1 Hardness

Consider the sequence $x = \{11, 12, 15, 14, 15, 15, 17\}$ in Example 2. Suppose that the error range is $\theta_i = 3$ for all the data points $i$. That is, for each point $x_i$, there are 7 potential modifications, $x'_i = \{x_i - 3, \ldots, x_i + 3\}$. A large number of $7^7$ combinations could be considered as possible repairs. In particular, the repairing of $x'_i$ is affected by the choices of $x'_{i-1}$ and $x'_{i+1}$ w.r.t. the speed change probability (in Figure 4). Intuitively, we can build a reduction from the 0/1 knapsack problem, by modeling the item values as the speed change probabilities, and thus show the hardness of our repairing problem.

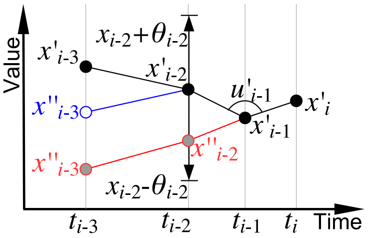

**Figure 7: Recurrence equation**

**Theorem 1.** *Given a sequence $x$ with error range $\theta$, repair cost budget $\delta$, and likelihood threshold $\ell$, the problem is NP-complete to determine whether exists a repair $x'$ of $x$ such that $\Delta(x, x') \le \delta$ and $L(x') \ge \ell$.*

## 3.2 Exact Algorithm

Referring to the aforesaid hardness on the maximum likelihood repair problem, we introduce a pseudo-polynomial time algorithm based on dynamic programming. Let us first illustrate the recurrence equation, upon which the dynamic programming algorithm naturally conducts. The correctness is also analyzed in the proof of Proposition 2.

### *3.2.1 Recurrence Equation*

Intuitively, referring to formulas (1) and (2), the speed change and the corresponding likelihood on a point $i$ are determined together with the preceding point $i-1$ and the successive point $i+1$. As illustrated in Figure 7, by considering one additional data point, say $i$, in the recurrence of dynamic programming, a new likelihood $L(u'_{i-1})$ defined on data point $i-1$ is introduced. Thereby, to find an optimal solution, we need to consider possible $x'_{i-2}, x'_{i-1}, x'_i$ values in each recurrence.

Let $x'_{1\ldots i}$ be a repair of the subsequence $x_{1\ldots i}$ with the maximum likelihood $L(x'_{1\ldots i})$, whose cost is $\Delta(x'_{1\ldots i}, x_{1\ldots i}) = c_i$, and the last two values of $x'_{1\ldots i}$ are $x'_{i-1}, x'_i$, respectively. We denote this maximum likelihood $L(x'_{1\ldots i})$ by $D(i, c_i, x'_{i-1}, x'_i)$.

The recurrence computation is as follows

$$\begin{aligned} & D(i, c_i, x'_{i-1}, x'_i) \qquad (4) \\ = & \max_{x'_{i-2} \in [x_{i-2} \pm \theta_{i-2}]} D(i-1, c_{i-1}, x'_{i-2}, x'_{i-1}) + L(u'_{i-1}) \end{aligned}$$

where $c_{i-1} = c_i - \Delta(x'_i, x_i)$, and $u'_{i-1} = \frac{x'_i - x'_{i-1}}{t_i - t_{i-1}} - \frac{x'_{i-1} - x'_{i-2}}{t_{i-1} - t_{i-2}}$.

Initially, for $i = 2$, we have

$$D(2, c_2, x'_1, x'_2) = 0, \forall x'_1 \in [x_1 \pm \theta_1], \forall x'_2 \in [x_2 \pm \theta_2].$$

For each $i \in \{2, \ldots, n\}, c_i \in \{0, \ldots, \delta\}, x'_{i-1} \in [x_{i-1} \pm \theta_{i-1}], x'_i \in [x_i \pm \theta_i]$, we compute $D(i, c_i, x'_{i-1}, x'_i)$ according to the aforesaid recurrence. Algorithm 1 presents the procedure of this recurrence computation.

Let $D(n, c_n, x'_{n-1}, x'_n)$ be the maximum likelihood, among all $c_n, x'_{n-1}, x'_n$. By retracing all $D(i, c_i, x'_{i-1}, x'_i)$ leading to this maximum likelihood, an optimal solution $x'$ is obtained.

**Example 4** (Recurrence computing)**.** *Consider again the sequence $x = \{11, 12, 15, 14, 15, 15, 17\}$ in Example 2, with probability distribution on speed changes in Figure 4. Suppose that the repair cost budget is $\delta = 3$.*

*To perform the recurrence on $D(i, c_i, x'_{i-1}, x'_i)$ in formula (4), we need to maintain a 4-dimension structure.* Table 1

```
Algorithm 1: DP($x, \theta, \delta$)
   Data: data sequence $x$, error range $\theta$, repair budget $\delta$
   Result: the maximum likelihood of the optimal repair
 1 initialize $D(2, c_2, x'_1, x'_2) \leftarrow 0$ for each $c_2, x'_1, x'_2$;
 2 for $i \leftarrow 3$ to $n$ do
 3     for $c_i \leftarrow 0$ to $\delta$ do
 4         foreach $x'_i \in [x_i \pm \theta_i]$ do
 5             $c_{i-1} \leftarrow c_i - \Delta(x'_i, x_i)$;
 6             foreach $x'_{i-1} \in [x_{i-1} \pm \theta_{i-1}]$ do
 7                 $D(i, c_i, x'_{i-1}, x'_i) \leftarrow -\infty$;
 8                 foreach $x'_{i-2} \in [x_{i-2} \pm \theta_{i-2}]$ do
 9                     if $D(i-1, c_{i-1}, x'_{i-2}, x'_{i-1}) \neq -\infty$ then
10                         $l \leftarrow D(i-1, c_{i-1}, x'_{i-2}, x'_{i-1}) + L(u'_{i-1})$;
11                         if $l > D(i, c_i, x'_{i-1}, x'_i)$ then
12                             $D(i, c_i, x'_{i-1}, x'_i) \leftarrow l$;
13                 if $D(i, c, x'_{i-1}, x'_i) + L^u(x_{i-1 \dots n}) \leq L^w(x)$ then
14                     $D(i, c, x'_{i-1}, x'_i) \leftarrow -\infty$;
15 return $\max_{c_n, x'_{n-1}, x'_n} D(n, c_n, x'_{n-1}, x'_n)$
```

*illustrates two dimensions on $i$ and $c$, and omits the other two dimensions of $x'_i, x'_{i-1}$ for simplicity. Each cell, e.g., for $i = 3, c_i = 2$, presents one of the $D(3, 2, x'_2, x'_3)$.*

*The recurrence performs from $i = 3$ to $i = n = 7$, with $c_i \in [0, 3]$ (as $\delta = 3$). The cell of $i = 7, c_i = 3$ obtains a $D(7, 3, x'_6, x'_7) = -6.0$. with the maximum likelihood. By retracing all $D(i, c_i, x'_{i-1}, x'_i)$ leading to this maximum likelihood, an optimal solution $x' = \{11, 12, 13, 14, 15, 16, 17\}$ is obtained. It is exactly the solution $x'$ under $\delta = 3$ in Example 3, with likelihood $L(x') = -6.0$.*

Intuitively, to show the correctness of Algorithm 1, it is sufficient to illustrate that the recurrence equation in formula (4) in dynamic programming always calculates the maximum likelihood in each step.

**Proposition 2.** *Algorithm 1 computes the optimal solution in $O(n\theta^3_{\max}\delta)$ time with $O(n\theta^2_{\max}\delta)$ space.*

### 3.2.2 *Pruning with Likelihood Bounds*

Intuitively, $D(i, c_i, x'_{i-1}, x'_i)$ in each recurrence obtain the optimal results on the subsequence $x_{1 \dots i}$. If we can obtain an upper bound of likelihood for the remaining subsequence $x_{i+1 \dots n}$, efficient pruning on subsequent recurrence involving this $D(i, c_i, x'_{i-1}, x'_i)$ enables.

Given a subsequence $x_{i \dots n}$, we can quickly compute an *upper* bound of maximum likelihood for possible repair $x'_{i \dots n}$

$$L^u(x_{i \dots n}) = (n - i - 1) \cdot \log p_{\max} \geq L(x'_{i \dots n}) = \sum_{j=i+1}^{n-1} L(x'_j),$$

where $p_{\max}$ is the maximum probability of a speed change.

Let $x''$ be a currently known repair with $\Delta(x, x'') \leq \delta$, e.g., efficiently computed by the simple greedy method below (introduced in Section 5.2), or simply $x'' = x$. We use the likelihood $L(x'')$ as the *lower* bound $L^w(x)$ of the maximum likelihood of the optimal solution $x^*$

$$L^w(x) = L(x'') \leq L(x^*).$$

**Table 1: Example of recurrence**

| $i \backslash c_i$ | 0 | 1 | 2 | 3 | $L^u_i$ |
|---|---|---|---|---|---|
| 1 | | | | | |
| 2 | $x'_1$=11,<br>$x'_2 = 12$ | | | | $L^u_2$=-6.0 |
| 3 | $x''_2$=12,<br>$x''_3$=15,<br>$D$=-1.4 | | $x'_2$=12,<br>$x'_3$=13,<br>$D$=-1.2 | | $L^u_3$=-4.8 |
| 4 | $x''_3$=15,<br>$x''_4$=14,<br>$D$=-3.7 | | $x'_3$=13,<br>$x'_4$=14,<br>$D$=-2.4 | | $L^u_4$=-3.6 |
| 5 | | $x''_4$=14,<br>$x''_5$=16,<br>$D$=-6.0 | $x'_4$=14,<br>$x'_5$=15,<br>$D$=-3.6 | | $L^u_5$=-2.4 |
| 6 | | | | $x'_5$=15,<br>$x'_6$=16,<br>$D$=-4.8 | $L^u_6$=-1.2 |
| 7 | | | | $x'_6$=16,<br>$x'_7$=17,<br>$D$=-6.0 | |

**Proposition 3.** *The recurrence on $D(i, c_i, x'_{i-1}, x'_i)$ could be pruned, if*

$$D(i, c_i, x'_{i-1}, x'_i) + L^u(x_{i-1 \dots n}) \leq L^w(x).$$

As Line 14 shown in Algorithm 1, we set $D(i, c_i, x'_{i-1}, x'_i) \leftarrow -\infty$ to stop the subsequent recurrence on $D(i, c_i, x'_{i-1}, x'_i)$.

**Example 5** (Pruning with likelihood)**.** *Let us still consider the sequence $x$ in Example 4. According to the probability distribution in Figure 4, we have $p_{\max} = \log(0.3) = -1.2$. For each level $i$, an upper bound of*

$$L^u(x_{i-1 \dots n}) = -1.2 * (7 - i)$$

*is computed, denoted by the column $L^u_i$ in Table 1. For instance, for $i = 2$, we have $L^u_2 = L^u(x_{1 \dots 7}) = -6.0$. It denotes that any repair on the (sub)sequence $x_{1 \dots 7}$ will not have likelihood greater than $-6.0$.*

*Moreover, we use the likelihood of the input sequence $x$ as the lower bound $L^w(x) = L(x) = -8.1$, which is calculated in Example 2.*

*Consider the cell of $i = 5, c_i = 1$ (in red in Table 1) with $D(5, 1, 14, 16) = -6.0$. According to $L^u_5 = L^u(x_{4 \dots 7}) = -2.4$, we have $D(5, 1, 14, 16) + L^u(x_{4 \dots 7}) = -6.0 - 2.4 = -8.4 < L^w(x) = -8.1$. It indicates that, given the current repair $x''_{1 \dots 4} = \{11, 12, 15, 14\}$ on the processed subsequence $x_{1 \dots 4}$, no matter how the remaining $x_{5 \dots 7}$ is repaired, the generated result $x''$ will always have likelihood $L(x'')$ lower than the lower bound, and thus cannot be the optimal solution. Any subsequent recurrence on this $D(5, 1, 14, 16)$ could be pruned.*

## 4. APPROXIMATE SOLUTION

As introduced at the beginning of Section 3, Algorithm 1 is pseudo-polynomial, whose complexity is determined by $\delta$ the budget of repair cost. Depending on the granularity of data values, there may be a huge number of possible repair costs to enumerate within $\delta$. For instance, in bad cases, the considered repair cost budget $\delta$ could be as high as $\delta = 4500$ in Figure 14 on STOCK data, or a $\delta = 3500$ in Figure 17 for ENGINE data.

To support efficient computing, our first method (in Section 4.1) is to map the space of possible repair cost values in $[0, \delta]$ to a constant space. While this simple approximation makes Algorithm 1 runs in linear time, no guarantee on approximation performance is obtained.

To trade off the complexity for approximation performance guarantee, we devise a quadratic-time constant-factor approximation algorithm (in Section 4.2). The intuition is, rather than approximating repair cost, we approximate the likelihood instead.

## 4.1 Linear Time Heuristics

We map the space of granted repair cost from $[0, \delta]$ to a constant space $[0, d]$, where $d$ is an integer constant, by introducing an approximate repair cost function. Let

$$H = \frac{\delta}{d}.$$

We define

$$\Delta'(x_i', x_i) = \lceil \frac{|x_i' - x_i|}{H} \rceil. \tag{5}$$

With this approximate repair cost function, instead of all $c_i \in \{0, \dots, \delta\}$, we calculate only $D(i, c_i', x_{i-1}', x_i')$ for

$$c_i' \in \{H \cdot \lceil \frac{c_i}{H} \rceil \leq \delta \mid c_i \in \{0, \dots, \delta\}\}, \tag{6}$$

with a total number $d + 1$ of possible cost values.

Moreover, instead of considering all candidates $x_i' \in [x_i \pm \theta_i]$, we only need to consider candidates in

$$x_i' \in \{x_i \pm (H \cdot j) \mid j \in [-\lfloor \frac{\theta_i}{H} \rfloor, \lfloor \frac{\theta_i}{H} \rfloor]\} \tag{7}$$

with a total number $2 \cdot \lfloor \frac{\theta_i}{H} \rfloor + 1$ of candidates. The reason is that all the candidates $x_i' \in [H \cdot j, H \cdot j + H)$ share the same approximate repair cost $j$, and thus are not considered.

**Example 6** (Linear time computation). *Consider the sequence $x$ in Example 2, with error range $\theta_i = 3$ for all data points $i$. Given a repair cost budget $\delta = 4$ and $d = 2$, we have $H = \frac{\delta}{d} = 2$.*

*According to formula (6), we calculate $D(i, c_i', x_{i-1}', x_i')$ only for the repair cost values $c_i' \in \{0, 2, 4\}$ instead of all possible repair costs $c_i \in \{0, 1, 2, 3, 4\}$.*

*In addition, the number of candidates is also reduced. For instance, for $x_3 = 15$ with $\theta_3 = 3$, the considered candidates in the exact algorithm are $x_3' \in \{12, 13, 14, 15, 16, 17, 18\}$. In the approximate computing, according to formula (7), only the candidates $x_3' \in \{13, 15, 17\}$ are considered instead.*

**Proposition 4.** *Algorithm 1 with approximate repair cost $\Delta'$ in formula (5) runs in $O(nd^4)$ time with $O(nd^3)$ space, where $d$ is a fixed constant.*

While this simple approximation performs well in practice (as shown in Section 6 of experiments), unfortunately, we did not obtain a theoretical bound of approximation ratio compared to the optimal solution.

## 4.2 Constant-Factor Approximation

Intuitively, rather than approximating repair cost, by directly approximating the likelihood, it might be more practical to keep the approximate likelihood bounded compared to the exact one. We present below a quadratic-time (for fixed error range $\theta$), constant-factor approximation algorithm, by approximating the likelihood. Before introducing the approximation, let us first revise the recurrence defined on likelihood (instead of defined on repair cost in Section 3.2).

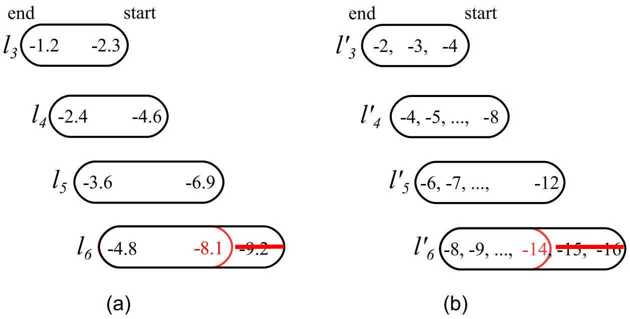

**Figure 8: Space of likelihood values considered in recurrence**

### 4.2.1 Recurrence Equation

Referring to the same intuition for the recurrence equation in formula (4), we still need to consider possible $x_{i-2}', x_{i-1}', x_i'$ values, to address the likelihood incrementation in each recurrence.

Let $x_{1\dots i}'$ be a repair of the subsequence $x_{1\dots i}$, whose likelihood is $L(x_{1\dots i}') = l_i$, the last two values of $x_{1\dots i}'$ are $x_{i-1}', x_i'$, respectively, and the cost $\Delta(x_{1\dots i}', x_{1\dots i})$ minimized. We denote this minimized cost $\Delta(x_{1\dots i}', x_{1\dots i})$ by $C(i, l, x_{i-1}', x_i')$.

The recurrence computation is as follows

$$\begin{aligned} & C(i, l_i, x_{i-1}', x_i') \qquad (8) \\ = & \min_{x_{i-2}' \in [x_{i-2} \pm \theta_{i-2}]} C(i-1, l_{i-1}, x_{i-2}', x_{i-1}') + \Delta(x_i', x_i) \end{aligned}$$

where $l_{i-1} = l_i - L(u_{i-1}')$, and $u_{i-1}' = \frac{x_i' - x_{i-1}'}{t_i - t_{i-1}} - \frac{x_{i-1}' - x_{i-2}'}{t_{i-1} - t_{i-2}}$.

Initially, for $i = 2$, we have

$$C(2, 0, x_1', x_2') = \Delta(x_1', x_1) + \Delta(x_2', x_2),$$

$\forall x_1' \in [x_1 \pm \theta_1], \forall x_2' \in [x_2 \pm \theta_2]$.

For each $i$, we consider $l_i$ in the range from $l_i^{start}$ to $l_i^{end}$, where

$$\begin{aligned} l_i^{start} &= (i-2) \cdot \log p_{\min} \\ l_i^{end} &= (i-2) \cdot \log p_{\max} \end{aligned}$$

(See the specific $l_i$ values considered in the range soon in Section 4.2.2.)

**Example 7.** *Consider again the sequence $x$ in Example 2. Referring to the probability distribution of speed changes in Figure 4, we have $\log p_{\max} = \log(0.3) = -1.2, \log p_{\min} = \log(0.1) = -2.3$.*

*Figure 8(a) illustrates the range $[l_i^{start}, l_i^{end}]$ of possible likelihoods that need to be considered for each $i$. For instance, we have $l_3^{start} = -2.3, l_i^{end} = -1.2$.*

### 4.2.2 Approximation Algorithm

Intuitively, the likelihood approximation is performed by mapping the range of likelihood, from $l_i^{start}$ to $l_i^{end}$, to a space presented in formula (10) below with the total number of likelihood values bounded (see the bound in the proof of Proposition 5).

Consider

$$K = -\varepsilon \cdot \log p_{\max},$$

where $\varepsilon > 0$ is an error parameter in approximation.

For each point $x_i$, we define approximate likelihood

$$L'(u'_i) = \lfloor \frac{L(u'_i)}{K} \rfloor. \tag{9}$$

In recurrence, for each $i$, we only need to consider the following approximate likelihood values

$$l'_i \in \{\lfloor \frac{l_i^{start}}{K} \rfloor, \lfloor \frac{l_i^{start}}{K} \rfloor + 1, \ldots, \lfloor \frac{l_i^{end}}{K} \rfloor\}, \tag{10}$$

with a total number $(i-2)\lfloor \frac{1}{\varepsilon}(\frac{\log p_{\min}}{\log p_{\max}} - 1)\rfloor$ of considered approximate likelihood values (according to formula (14) the proof of Proposition 5).

By further considering each $x'_{i-1} \in [x_{i-1} \pm \theta_{i-1}], x'_i \in [x_i \pm \theta_i]$, we compute $C(i, l'_i, x'_{i-1}, x'_i)$ according to the aforesaid recurrence. Algorithm 2 presents the procedure of this recurrence computation with approximate likelihood.

```
Algorithm 2: DPC($x, \theta, \delta, \varepsilon$)
  Data: data sequence $x$, error range $\theta$, repair budget $\delta$,
        approximation factor $\varepsilon$
  Result: the maximum likelihood of the optimal repair
  $K \leftarrow -\varepsilon \cdot \log p_{\max}$;
  initialize $C(2, 0, x'_1, x'_2) \leftarrow \Delta(x'_1, x_1) + \Delta(x'_2, x_2)$ for each
    $x'_1, x'_2$;
  for $i \leftarrow 3$ to $n$ do
      for $l'_i \leftarrow \lfloor \frac{l_i^{start}}{K} \rfloor$ to $\lfloor \frac{l_i^{end}}{K} \rfloor$ do
          foreach $x'_i \in [x_i \pm \theta_i]$ do
              $l'_{i-1} \leftarrow l'_i - L'(u'{}_i)$;
              foreach $x'_{i-1} \in [x_{i-1} \pm \theta_{i-1}]$ do
                  $C(i, l'_i, x'_{i-1}, x'_i) \leftarrow \infty$;
                  foreach $x'_{i-2} \in [x_{i-2} \pm \theta_{i-2}]$ do
                      if $C(i-1, l'_{i-1}, x'_{i-2}, x'_{i-1}) \neq \infty$ then
                          $c \leftarrow$
                          $C(i-1, l'_{i-1}, x'_{i-2}, x'_{i-1}) + \Delta(x'_i, x_i)$;
                          if $c \leq \delta$ and $c < C(i, l'_i, x'_{i-1}, x'_i)$
                          then
                              $C(i, l'_i, x'_{i-1}, x'_i) \leftarrow c$;
  return $\max_{l_n, x'_{n-1}, x'_n} C(n, l'_n, x'_{n-1}, x'_n)$
```

Let $l'_n$ be the maximum (approximate) likelihood with $C(n, l'_n, x'_{n-1}, x'_n) \leq \delta$, among all $l'_n, x'_{n-1}, x'_n$. By retracing all $C(i, l'_i, x'_{i-1}, x'_i)$ leading to this $C(n, l'_n, x'_{n-1}, x'_n)$, an approximate solution $x'$ is obtained.

**Example 8.** *Consider the ranges of likelihood in Figure 8(a) in Example 7. Given $\varepsilon = 0.5$, we have $K = -\varepsilon \cdot \log(p_{\max}) = -0.5 * -1.2 = 0.6$.*

*According to formula (10), for each level i, we compute a finite set of approximate likelihood values to consider in recurrence. For instance, as illustrated in Figure 8(b), we consider $l'_3 \in \{-4, -3, -2\}$ for $i = 3$, where $\lfloor \frac{l_3^{start}}{K} \rfloor = \lfloor \frac{-2.3}{0.6} \rfloor = -4$, $\lfloor \frac{l_3^{end}}{K} \rfloor = \lfloor \frac{-1.2}{0.6} \rfloor = -2$.*

**Proposition 5.** *Algorithm 2 with approximate likelihood $L'$ outputs a repair $x'$ with $L(x') \geq (1+\varepsilon) \cdot L(x^*)$, in $O(n^2\theta_{\max}^3)$ time with $O(n^2\theta_{\max}^2)$ space.*

### 4.2.3 Pruning

Similar to the pruning for the exact algorithm in Section 3.2.2, let us consider $L^u(x)$ and $L^w(x)$, the upper and lower bounds of the likelihood of the optimal solution.

**Proposition 6.** *The recurrence on $C(i, l_i, x'_{i-1}, x'_i)$ stops, if $l_i + L^u(x_{i-1\ldots n}) \leq L^w(x)$.*

Indeed, since $L^u(x_{i-1\ldots n}) < 0$, any $l_i \leq L^w(x)$ can be directly ignored. We can further set

$$l_i^{start} = \max\left((i-2) \cdot \log p_{\min}, L^w(x)\right). \tag{11}$$

For the approximate $l'_i$ considered in the implementation of recurrence, the pruning condition is as follows.

**Corollary 7.** *The recurrence on $C(i, l'_i, x'_{i-1}, x'_i)$ could be pruned, if*

$$K \cdot (l'_i + i - 2) + L^u(x_{i-1\ldots n}) \leq L^w(x).$$

**Example 9.** *Similar to Example 5, we use the likelihood of the input sequence x as the lower bound $L^w(x) = L(x) = -8.1$, which is calculated in Example 2.*

*According to formula (11), for $i = 6$, we have the new $l_6^{start} = \max(-9.2, -8.1) = -8.1$, where $-9.2$ is the original $l_6^{start}$, as shown in Figure 8(a).*

*Referring to $K = 0.6$ in Example 8, the new $\lfloor \frac{l_6^{start}}{K} \rfloor = \lfloor \frac{-8.1}{0.6} \rfloor = -14$ is sufficient, as also illustrated in Figure 8(b).*

# 5. FROM DISCRETE TO CONTINUOUS

Rather than approximating the computation, in this section, we approximate the discrete probability distribution $P$ by a continuous probability distribution. The intuition is that, with a proper continuous probability distribution, the maximum likelihood repairing problem could be transformed to a quadratic programming problem (according to Proposition 12). It enables more fast computing and the application of existing efficient solvers.

## 5.1 Transformation

First, let us illustrate the high similarity between the discrete probability distribution observed in real datasets and the continuous probability distribution.

**Example 10** (Probability distribution approximation)**.** *We study the probability distribution of speed changes over three real datasets (see Section 6 for details of these datasets), together with the corresponding probability density function of normal distribution $\mathcal{N}(\mu, \sigma)$ for approximation.*

*Figure 2(a) presents the distribution over GPS dataset, together with a normal distribution $\mathcal{N}(0, 1)$. For the STOCK dataset, we plot $\mathcal{N}(0, 1.2)$ in Figure 9(a). Figure 9(b) illustrates the distribution $\mathcal{N}(0, 0.8)$ over the ENGINE dataset.*

Referring to the aforesaid proximity, assume that the probability distribution $u_i$ follows the normal distribution $\mathcal{N}(0, \sigma^2)$, having the probability density function

$$P(u_i) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{u_i^2}{2\sigma^2}}. \tag{12}$$

**Proposition 8.** *Given the probability density function in formula (12), we have $L(x') \geq L(x'')$ if and only if*

$$\sum_{i=2}^{n-1} (u'_i)^2 \leq \sum_{i=2}^{n-1} (u''_i)^2,$$

*where $u'_i, u''_i$ are speed changes w.r.t. $x'_i, x''_i$, respectively.*

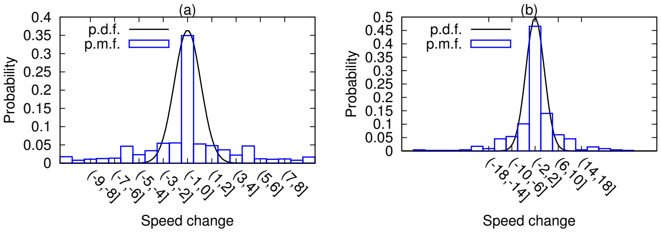

**Figure 9: Probability distribution approximation over (a) STOCK and (b) ENGINE datasets**

Therefore, we can rephrase Problem 1 as

$$\min \sum_{i=2}^{n-1}\left(\frac{x'_{i+1}-x'_i}{t_{i+1}-t_i}-\frac{x'_i-x'_{i-1}}{t_i-t_{i-1}}\right)^2 \tag{13}$$

$$\text{s.t.} \quad \sum_{i=1}^{n}|x'_i-x_i|\le\delta$$

$$x'_i\in[x_i\pm\theta_i] \qquad 1\le i\le n$$

Existing tools on quadratic programming, e.g., Gurobi[2], can be directly applied.

## 5.2 Simple Greedy

To compute very fast a repair (e.g., for pruning in Section 3.2.2), we can use a simple greedy heuristic for the transform problem in formula (13).

Let $x'$ be a current repair, initially $x' = x$. A natural greedy strategy is to select a $x'_i$ with the maximum $|u'_i|$, and pay one unit cost on such $x'_i$ towards the reduce of $|u'_i|$.

$$x'_i = \underset{x'_i\in[x'_i\pm 1]\cap[x_i\pm\theta_i]}{\arg\max}\left|\frac{x'_{i+1}-x'_i}{t_{i+1}-t_i}-\frac{x'_i-x'_{i-1}}{t_i-t_{i-1}}\right|.$$

The algorithm terminates after $\delta$ iterations or no $|u'_i|$ could be further reduced.

**Example 11** (Greedy computation). *Consider again the sequence $x = \{11, 12, 15, 14, 15, 15, 17\}$ in Example 2. The simple greedy algorithm will first select a data point, i.e., $x_3 = 15$, with the maximum $|u_3| = 4$. A unit cost repair will be performed on $x_3$, i.e., $x'_3 = 14$ such that $|u'_3| = 2$ is reduced, as illustrated in Figure 10(a). It is notable that such a repair $x'_3$ on data point 3 also affects the speed changes in data points 2 and 4, having $u'_2 = 1, u'_4 = 1$ in the repaired sequence.*

*In the next iteration, the greedy algorithm will choose the current largest $|u_i|$, i.e., still $u'_3$. It generates a new repair with $x''_3 = 13$, as shown in Figure 10(b).*

*The iteration carries on by further repairing $x'''_6 = 16$ in Figure 10(c). Since all the speed changes become 0 in Figure 10(d), i.e., no $|u'_i|$ could be further reduced, the algorithm terminates, and returns a repaired sequence $x''' = \{11, 12, 13, 14, 15, 16, 17\}$.*

*If a repair cost budget $\delta = 2$ is given, the greedy computation will stop after two iterations. The returned repair is $x'' = \{11, 12, 13, 14, 15, 15, 17\}$ as illustrated in Figure 10(b).*

**Proposition 9.** *The greedy algorithm runs in $O(\max(n, \delta))$ time.*

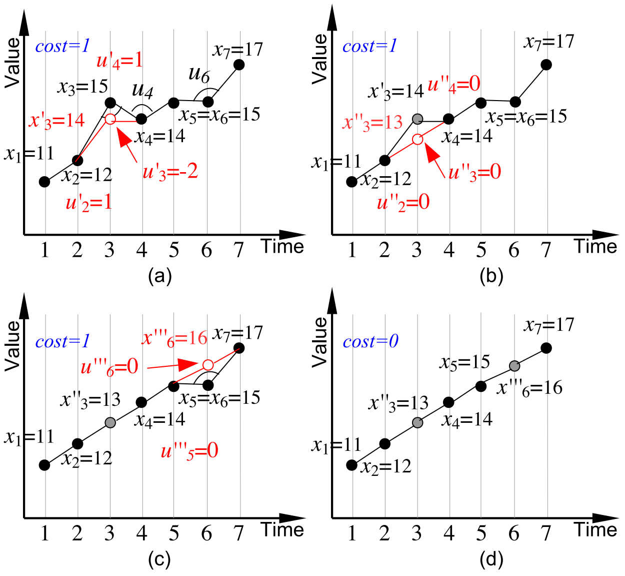

**Figure 10: Simple Greedy Example**

# 6. EXPERIMENT

In this section, we experimentally compare our proposed methods, (1) DP the exact algorithm in Section 3.2, (2) DPC the quadratic-time constant-factor approximation algorithm in Section 4.2, (3) DPL the linear time heuristic algorithm in Section 4.1, (4) QP the quadratic programming solver in Section 5.1, (5) SG the simple greedy algorithm in Section 5.2, with (6) SCREEN the state-of-the-art approach [14]. We omit reporting the other methods, such as the smoothing-based EWMA [6], or the constraint-based [7], owing to the clearly worse results (which are also observed in [14]).

The experiment runs on three real datasets and one synthetic datasets. (1) The STOCK dataset is originally clean, and errors are injected by randomly replacing the values of some data points. Thereby, the original clean data serves as the ground truth. (2) For the GPS data, collected by carrying a smartphone and walking around at campus, we manually mark the trajectory during data collection in a map and use the trajectory as the ground truth. (3) For the ENGINE data, we do not have the ground truth of sensor reading sequences. However, we have another observation (switching-count), which could be predicted from sensor readings. Therefore, instead of directly evaluating w.r.t. the ground truth of sensor readings (which are not available), we evaluate the prediction of switching-count by sensor readings with/without repairing. The switching-count observation serves as the ground truth of prediction. (4) For the SYNTHETIC data, generated by ourselves, we naturally have the ground truth.

The probability distributions of speed changes are estimated over the employed datasets. For each data point in a sequence, we calculate its speed change value before and after the point, according to formula (1). By counting the appearance of speed change values in the sequence, we estimate the probabilities of the speed changes. Figures 9(a), 2(b), 9(b) and 18 report the estimated probability distributions over the STOCK, GPS, ENGINE and SYNTHETIC datasets, respectively.

[2] http://www.gurobi.com

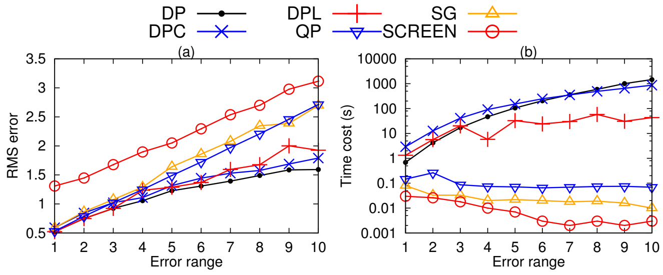

**Figure 11: Varying error range $\theta$, over STOCK with error number 600 and data size 1282**

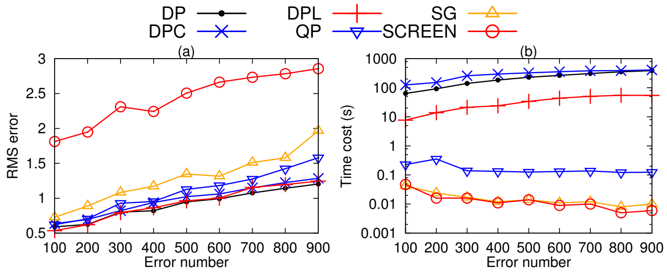

**Figure 12: Varying error numbers, over STOCK with error range $\theta = 5$ and data size 2564**

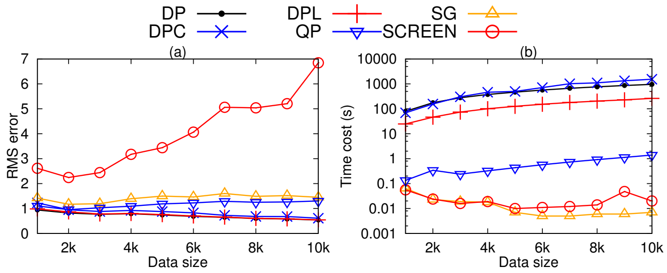

**Figure 13: Scalability, over STOCK with error range $\theta = 5$ and error size 400**

## 6.1 STOCK with Various Injected Errors

The STOCK dataset records the daily prices of a stock from 1984-09 to 2010-02, with 12826 data points in total. Since the data is originally clean, following the same line of precisely evaluating the repair effectiveness [1], errors are injected by randomly replacing the values of some data points.

Let $x_{\text{truth}}$ be the ground truth of clean sequence, and $x_{\text{repair}}$ be the repaired sequence. The repair accuracy is measured by root-mean-square error (RMS) [8], evaluating how close the repaired sequence $x_{\text{repair}}$ is to the ground truth $x_{\text{truth}}$. The lower the RMS error is, the closer (more accurate) the repair is to the ground truth.

Besides the RMS performance on repairing accuracy, we also report the corresponding time cost and report the likelihood of repair results.

For the original clean STOCK dataset, we have error range $\theta = 0$ (i.e., no need to repair). To evaluate the repair performance, we manually inject errors in the dataset with error range $\theta > 0$. Figure 11 presents the results by varying the ranges of injected errors from $\theta = 1$ to 10. First, as shown in Figure 11(a), it is not surprising that the larger the error range is, the higher the RMS distance will be between the truth and repair results. Our exact approach DP achieves the lowest RMS measure, while its corresponding time cost is the highest in Figure 11(b). While both the approximate methods DPC (with $\varepsilon = 2$) and DPL (with $d = 1000$) also have a considerably low RMS, the time cost of the linear time heuristic DPL is significantly lower.

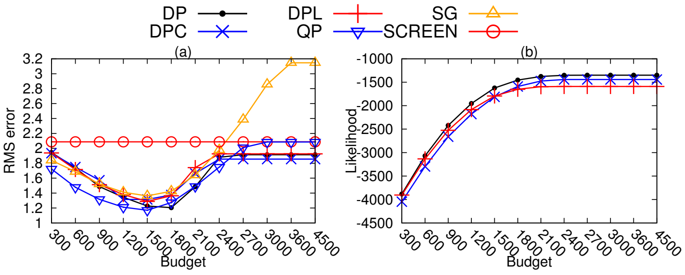

**Figure 14: Varying repair cost budget $\delta$, over STOCK with error range $\theta = 5$, error number 600 and data size 1282**

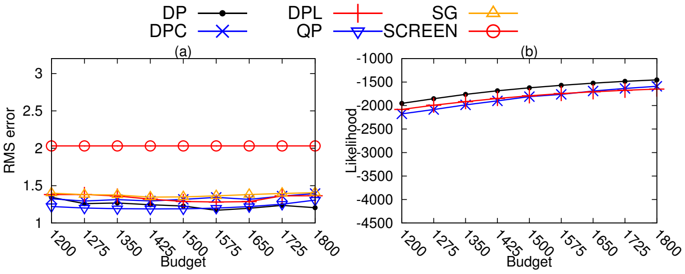

**Figure 15: Detailed results with $\delta$ in the range of 1200 to 1800 in Figure 14**

To evaluate the scalability, Figure 12 reports the results on various number of errors that are injected in the data, and Figure 13 presents the results over various data sizes. Generally, similar results are observed as in Figure 11. DP, DPC and DPL methods show better RMS performance, while their time costs are higher. The existing method SCREEN has worse RMS measure, but runs faster. QP and SG provide a trade-off between effectiveness and efficiency.

One interesting result is that the time cost of DP is lower than that of DPC, with small error range in Figure 11(b), or with small error number in Figure 12(b). The reason is that with a small error range, the corresponding repair budget $\delta$ needed is small as well (see results on various repair budgets below). The DP algorithm with $O(n\theta_{\max}^3\delta)$ complexity runs faster as well. For the same reason, in Figure 12(b), a larger number of errors lead to larger $\delta$, and thus DP shows higher time cost (closer to that of DPC).

Figure 14 reports the results by varying the repair cost budget $\delta$. As shown in Figure 14(a), if the repair cost budget is set too small, the dirty points cannot be fully repaired, with higher RMS measure. The corresponding likelihood of repair results in Figure 14(b) is low as well. On the other hand, if the repair cost budget $\delta$ sets too large, the sequence might be over-repaired. The RMS measure is high as well. Note that by further increasing the repair cost budget, the likelihood could not increase further in Figure 14(b). The reason is that the repaired sequence reaches the allowed re-

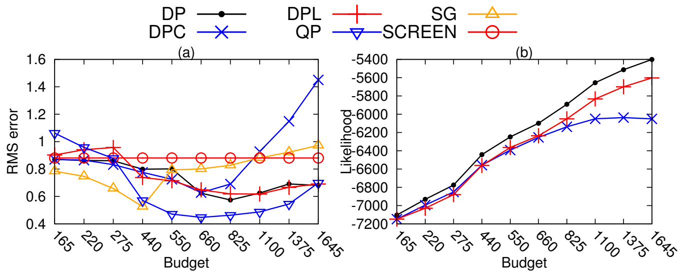

**Figure 16: Varying repair cost budget $\delta$ over GPS**

pair range $\theta$. The corresponding RMS measure in Figure 14(a) does not change either.

Nevertheless, Figure 14 provides a guideline of setting the repair cost budget $\delta$ in practice. That is, by increasing $\delta$, we observe the corresponding likelihood of results. When the likelihood does not significantly increase, e.g., with $\delta = 1500$ in Figure 14(b), the corresponding repair result is the best (with the lowest RMS). It denotes the case where the sequence is neither insufficiently repaired (with a low likelihood) nor over-repaired (with too large repair cost budget but no much likelihood gain).

It is worth noting that a range of $\delta$ could be considered, from 15 to 20 in Figure 6(b) in Example 3 as aforesaid, such that the repair error keeps low in Figure 6(a). We illustrate a very large range of $\delta$ in Figure 14 in order to verify the guideline of choosing a proper $\delta$. Once a proper range of $\delta$ is identified, e.g., from 1500 to 1800 where the likelihood no longer significantly increases, the repair results are stable. To demonstrate the robustness, Figure 15 for the results with $\delta$ in the range of 1200 to 1800. As shown, the result appears to be less sensitive in the chosen range of $\delta$ under the aforesaid guideline.

## 6.2 GPS with Naturally Embedded Errors

In order 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 150 dirty points are manually identified (among total 2358 clear points in the trajectory). True locations of dirty points are also manually labeled, as ground truth.

Since the errors are originally embedded, we don't have experiments on various errors settings in this dataset. There is only one parameter to tune, i.e., the repair cost budget $\delta$.

Figure 16 reports the RMS measure and likelihood of results, by varying the repair cost budget $\delta$. Similar to Figure 14, when the budget $\delta$ is small, dirty data might not be sufficiently repaired and thus the likelihoods of results are low. By granting more budget (larger $\delta$), while the data could be over-repaired (higher RMS error), the likelihood does not significantly increase further. The results verify again the aforesaid guideline of setting the repair cost budget $\delta$, by observing the likelihood of results.

## 6.3 ENGINE without Labeled Errors and Truth

To demonstrate the effectiveness in real applications, we employ the ENGINE dataset, where neither the dirty data nor the corresponding ground truth are labeled. The ENGINE dataset collects four sequences of a crane, produced by a heavy industry company, including DT0, engine-speed, pump-volume and switching-count, which monitor the working status of the device. The meaning of variables and their domains are presented in Table 2. The total number of data points in each sequence is 464. (Link to data will be publicly available after anonymous review.)

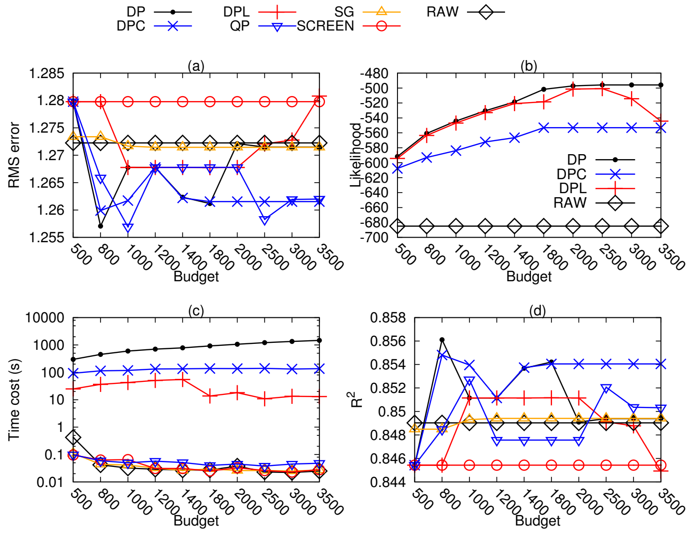

**Figure 17: Varying repair budget $\delta$ over ENGINE**

**Table 2: ENGINE variables**

| Variable | Description | Domain |
|---|---|---|
| DT0 | Current of a proportioner called DT0 | [200, 800] |
| engine-speed | Rotate speed of the engine | [800, 2000] |
| pump-volume | Swept volume of the pump | [0, 100] |
| switching-count | Times the crane pumping per minute | [3, 27] |

Owing to the sensor issues, the readings of engine-speed are often inaccurate, and thus need cleaning. Instead of directly measuring the accuracy of repairs (which is not possible since no error and truth are known in advance), we evaluate the application performance over the data with and without cleaning. Since switching-count is often missing in practice, the application is to predict switching-count according to the readings of DT0, engine-speed, pump-volume. To perform the prediction, we use the *LinearRegression Class* in WEKA[3], i.e., switching-count $= \alpha *$ pump-volume $+ \beta *$ DT0 $+ \gamma *$ engine-speed, where $\alpha, \beta, \gamma$ are parameters of regression.

To evaluate the application accuracy, two measures are employed, RMS reporting the closeness of the predicted values to the observed switching-count values (that are not missing) and $R^2$ the coefficient of determination [5]. A lower RMS error or a higher $R^2$ measure denotes better prediction accuracy.

Figure 17 illustrates the results by varying the repair cost budget $\delta$. RAW denotes the results of prediction application over the raw data without performing cleaning. As shown,

[3] http://www.cs.waikato.ac.nz/~ml/weka/

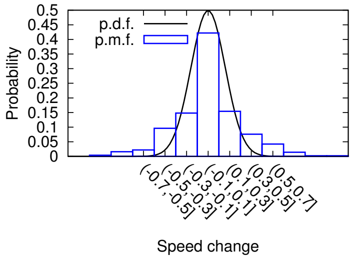

**Figure 18: Probability distribution of speed changes over SYNTHETIC data**

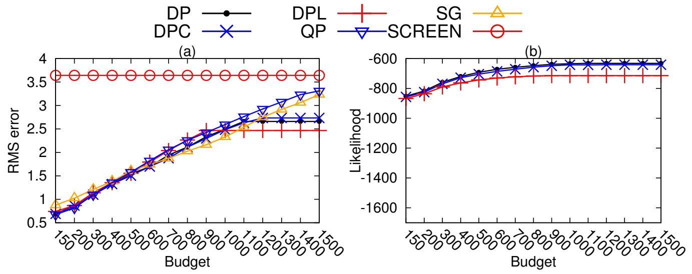

**Figure 19: Repair results on various repair cost budget $\delta$ over SYNTHETIC-clean**

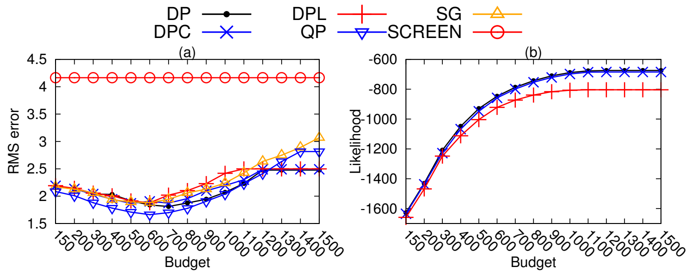

**Figure 20: Repair results on various repair cost budget $\delta$ over SYNTHETIC-injected**

by a proper setting of repair cost budget $\delta$, the RMS and $R^2$ of prediction application improves, compared to RAW without repairing.

The RMS and $R^2$ measures over the application are not as stable as the RMS of repair results, owing to the prediction model. Nevertheless, as illustrated in Figure 17(b), the likelihoods of repair results are stable, which is similar to the observation in other datasets. It is also observed that a result, neither insufficiently repaired (with a low likelihood) nor over-repaired (with too large repair cost budget but no much likelihood gain), leads to better (application) effectiveness.

As shown in Figure 17(a), the existing SCREEN repair leads to higher RMS error in prediction. It is not surprising given the higher RMS error of repairing by SCREEN, e.g., in Figure 14(a) over STOCK and Figure 16(a) over GPS. The other prediction measure $R^2$ of SCREEN in Figure 17(d) is lower, which verifies again the results.

## 6.4 SYNTHETIC Evaluating False Positives

We consider two synthetic datasets, SYNTHETIC-clean and SYNTHETIC-injected. The SYNTHETIC-clean dataset is generated by using speeds with changes sampled from the probabilistic model (Gaussian distribution with $\mu = 0$ and $\sigma = 0.8$) as the p.d.f. shown in Figure 18. By injecting errors into SYNTHETIC-clean data, following the same line of injection over STOCK dataset, with error range $\theta = 5$, we obtain the SYNTHETIC-injected version. Figure 18 shows the probabilistic distribution of speed changes over SYNTHETIC data.

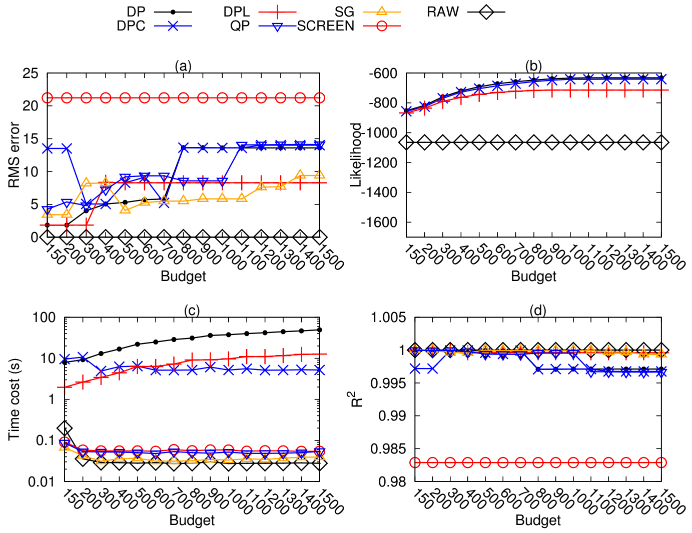

**Figure 21: Prediction results on various repair cost budget $\delta$ over SYNTHETIC-clean**

Figures 19 and 20 present repair results over SYNTHETIC-clean and SYNTHETIC-injected, respectively. The results verify again the guideline on setting the repair cost budget $\delta$ (see Example 3 for more details about the guideline). That is, as shown in Figure 20(b) over SYNTHETIC-injected, a budget $\delta$ from 500 to 700 will be preferred, where the likelihood stops significantly increase. The RMS error of repair results under such $\delta$ is lower in Figure 20(a). Similarly, for the SYNTHETIC-clean data in Figure 19(b), no significant increase of likelihood is observed. A small $\delta$ such as 150 is sufficient, i.e., the approach prefers to make little change over the clean data. This conservative result over SYNTHETIC-clean data also demonstrates that our proposed algorithm is *not* over aggressive.

Moreover, to perform the prediction experiments (as in Section 6.3), we generate similarly other two determinant sequences plus one dependent sequence. In the SYNTHETIC-clean version without error injection, the dependent sequence could be predicted from determinant sequences, i.e., the standard baseline that always predicts the mean from the training data.

Figures 21 and 22 present the prediction results over the SYNTHETIC-clean and SYNTHETIC-injected data, respectively. Again, referring to the guideline of setting repair cost budget $\delta$, a small $\delta = 150$ is preferred in Figure 21(b) on SYNTHETIC-clean. The corresponding prediction results are closest to the standard baseline, i.e., the prediction over the RAW clean data without repairing, in Figure 21(a). Recall that the standard baseline (on RAW over SYNTHETIC-clean) always predicts the dependent sequence with RMS error 0. Moreover, a budget $\delta$ around 600 will be chosen ac-

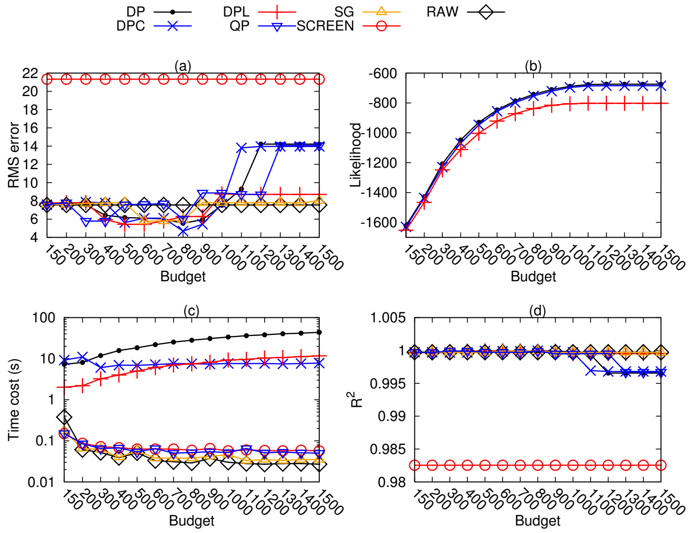

**Figure 22: Prediction results on various repair cost budget $\delta$ over SYNTHETIC-injected**

cording to Figure 22(b) on SYNTHETIC-injected. Again, the corresponding prediction RMS error is lower under such $\delta$ in Figure 22(a).

## Summary

We list the proposed methods in the decreasing order of time cost as follows: (1) The exact DP algorithm is preferred, in order to achieve better repair accuracy (as shown in Figure 11(a)), while its time cost is high given the pseudo-polynomial time complexity $O(n\theta_{\max}^3\delta)$ in Proposition 2. (2) The DPC approximation algorithm, with time complexity $O(n^2\theta_{\max}^3)$ not directly related to $\delta$ in Proposition 5, shows better time performance when given a larger repair cost budget $\delta$, e.g., $\delta = 3500$ in Figure 17(c). (3) The DPL approximation algorithm, with time complexity $O(nd^4)$ where $d$ is a constant in Proposition 4, achieves even lower time cost, but its repair error could be higher, as shown in Figure 11. (4) The QP algorithm can have significantly lower time cost without losing much repair accuracy, when the exact discrete probability distribution is similar to the approximate continuous probability distribution, e.g., when the error range is small in Figure 11. (5) The simple heuristic algorithm SG always achieves the lowest time cost among the proposed methods, while its repair accuracy could not be guaranteed.

# 7. RELATED WORK

### *Constraint-based Cleaning*

Besides speed constraints [14], sequential dependencies [7] also specify constraints on sequential data. Instead of concerning the speeds of value changes, sequential dependencies consider the range of value changes between two consecutive data points, while the distances on timestamps between data points are not involved. In this sense, sequential dependencies could be interpreted as a special class of speed constraints declared on time series with fixed time intervals.

With either constraint, the constraint-based cleaning identifies and repairs only the violations to the constraints, without indicating the most likely answers among all the valid repairs that satisfy the constraints. As reported in Section 6, by further revealing the likelihood of repairs, our proposal can obtain more accurate repair results than the constraint-based method.

### *Statistical-based Cleaning*

The likelihood-based repairing over relational data has been studied in [15]. For the relation to repair, several attributes are identified with dirty values, namely *flexible* attributes, which can be modified, while the other attributes contain correct values, called *reliable* attributes. Dependencies between reliable attributes and flexible attributes are modeled and changed with flexible attributes repaired. The repairing is thus to maximize the likelihood of data replacement, given the data distribution in the relation. The difference between relational data and our studied sequential data is obvious, and thus this likelihood-based repairing over relational data [15] is not directly applicable to sequential data.

A more complex relational dependency network is introduced in [11] to model the probabilistic relationships among attribute, such as cyclic relational dependencies. Instead of maximizing the likelihood as in [15] and our proposal, the repairing in [11] performs iteratively, and observe the change of distributions before and after a repair. The cleaning process terminates when the divergence of distributions is sufficiently small. Again, without a clear dependency relationship, the relational dependency network cannot be built for single data sequences, and thus this method [11] is not applicable.

To assess the quality of a repair, a statistical distortion method is proposed in [4]. While the statistical distortion directly observes the value distribution, our considered likelihood is defined on the changes of speed (value changes).

# 8. CONCLUSIONS

In this study, we study the cleaning of dirty vales in a sequence. First, we show that existing speed constraint-based approach either does not precisely repair large spike errors or simply ignore small errors. Rather than restricting the repair w.r.t. max/min speed constraints, we model the likelihood of a repair by observing its speed changes. Under the discipline that speeds should not change significantly in a time point, the likelihood-based repairing is thus to maximize the likelihood on speed changes, instead of minimizing the changes in the constraint-based repairing.

To efficiently compute the maximum likelihood solution, we propose 1) a pseudo-polynomial time exact algorithm, 2) a quadratic-time constant-factor approximation algorithm, 3) a linear time heuristic algorithm, 4) a quadratic programming transformation approximation, and 5) a simple greedy heuristic. Experiments on several real datasets demonstrate the better performance of our proposal, in both repairing and application accuracies, compared to the state-of-the-art constraint-based repairing.

## Acknowledgement

This work is supported in part by the Tsinghua University Initiative Scientific Research Program; Tsinghua National Laboratory Special Fund for Big Data Science and Technology; China NSFC under Grants 61572272, 61325008, 61370055 and 61202008.

# 9. REFERENCES

[1] P. Bohannon, M. Flaster, W. Fan, and R. Rastogi. A cost-based model and effective heuristic for repairing constraints by value modification. In *SIGMOD*, pages 143–154, 2005.

[2] D. R. Brillinger. *Time series: data analysis and theory*, volume 36. Siam, 2001.

[3] X. Chu, I. F. Ilyas, and P. Papotti. Holistic data cleaning: Putting violations into context. In *ICDE*, pages 458–469, 2013.

[4] T. Dasu and J. M. Loh. Statistical distortion: Consequences of data cleaning. *PVLDB*, 5(11):1674–1683, 2012.

[5] N. R. Draper and H. Smith. *Applied regression analysis (2. ed.)*. Wiley series in probability and mathematical statistics. Wiley, 1981.

[6] E. S. Gardner Jr. Exponential smoothing: The state of the art–part ii. *International Journal of Forecasting*, 22(4):637–666, 2006.

[7] L. Golab, H. J. Karloff, F. Korn, A. Saha, and D. Srivastava. Sequential dependencies. *PVLDB*, 2(1):574–585, 2009.

[8] S. R. Jeffery, M. N. Garofalakis, and M. J. Franklin. Adaptive cleaning for RFID data streams. In *VLDB*, pages 163–174, 2006.

[9] R. M. Karp. Reducibility among combinatorial problems. In *Proceedings of a symposium on the Complexity of Computer Computations, held at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York.*, pages 85–103, 1972.

[10] X. Li, X. L. Dong, K. Lyons, W. Meng, and D. Srivastava. Truth finding on the deep web: Is the problem solved? *PVLDB*, 6(2):97–108, 2012.

[11] C. Mayfield, J. Neville, and S. Prabhakar. ERACER: a database approach for statistical inference and data cleaning. In *SIGMOD*, pages 75–86, 2010.

[12] M. D. Smucker, J. Allan, and B. Carterette. A comparison of statistical significance tests for information retrieval evaluation. In *CIKM*, pages 623–632, 2007.

[13] S. Song, C. Li, and X. Zhang. Turn waste into wealth: On simultaneous clustering and cleaning over dirty data. In *SIGKDD*, pages 1115–1124, 2015.

[14] S. Song, A. Zhang, J. Wang, and P. S. Yu. SCREEN: stream data cleaning under speed constraints. In *SIGMOD*, pages 827–841, 2015.

[15] M. Yakout, L. Berti-Equille, and A. K. Elmagarmid. Don't be scared: use scalable automatic repairing with maximal likelihood and bounded changes. In *SIGMOD*, pages 553–564, 2013.

# APPENDIX

# A. PROOFS

### *Proof of Theorem 1*

*Proof.* The problem is clearly in NP. Given a repair $x'$, it can be verified in polynomial time whether each point repair $x'_i$ is in the valid range and $\Delta(x, x') \leq \delta$. Besides, the likelihood $L(x')$ can also be computed in polynomial time.

To prove the NP-hardness, we show a reduction from the 0/1 knapsack problem, which is one of Karp's 21 NP-complete problems [9]. Given a set of $n$ items numbered from 1 up to $n$, each with a weight $w_i$ and a value $v_i$, along with a maximum weight capacity, the problem is to maximize the sum of the values of the items in the knapsack so that the sum of the weights is less than or equal to the knapsack's capacity.

We create five data points with values $x_{i1}, \dots, x_{i5}$ for each item $i$, having

$$\begin{aligned} x_{i1} &= b_i, & \theta_{i1} &= 0, \\ x_{i2} &= b_i, & \theta_{i2} &= 0, \\ x_{i3} &= 2b_i, & \theta_{i3} &= w_i, \\ x_{i4} &= 4b_i, & \theta_{i4} &= 0, \\ x_{i5} &= 7b_i, & \theta_{i5} &= 0, \end{aligned}$$

where $b_i = 4 * (w_1 + \cdots + w_{i-1}) + 2w_i + i$.

The probabilities of speed changes in $P$ are defined as

$$P(u_i) = \begin{cases} 0 & \text{if } u_i = u_{i1} = 0, \\ v_i/3 & \text{if } u_i = u_{i2} = b_i + w_i, \\ v_i/3 & \text{if } u_i = u_{i3} = b_i - 2w_i \\ v_i/3 & \text{if } u_i = u_{i4} = b_i + w_i, \\ 0 & \text{if } u_i = u_{i5} = -7bi, \\ 0 & \text{otherwise} \end{cases}$$

for $i = 1, \dots, n$.

We can show that there is a subset of items with total weight $W$ and total value $V$, if and only if there is a repair $x'$ with $\Delta(x, x') = W$ and $L(x') = V$. □

### *Proof of Proposition 2*

*Proof.* First, we show the correctness of the recurrence equation in formula (4) in the dynamic programming. That is, $D(i, c_i, x'_{i-1}, x'_i)$ in formula (4) always calculates the maximum likelihood $L(x'_{1\dots i})$ whose cost is $\Delta(x'_{1\dots i}, x_{1\dots i}) = c_i$, and the last two values are $x'_{i-1}, x'_i$.

Suppose that there is another repair $x''$ whose cost is also $\Delta(x''_{1\dots i}, x_{1\dots i}) = c_i$, and the last two values are also $x''_{i-1} = x'_{i-1}, x''_i = x'_i$, and $L(x'') > L(x')$.

If $x''_{i-2} \neq x'_{i-2}$, referring to $L(x'') > L(x')$, we have

$$\begin{aligned} &D(i-1, c_{i-1}, x''_{i-2}, x''_{i-1}) + L(u''_{i-1}) \\ >&D(i-1, c_{i-1}, x'_{i-2}, x'_{i-1}) + L(u'_{i-1}). \end{aligned}$$

It contradicts the condition that formula (4) takes a $x'_{i-2}$ with the maximum $D(i-1, c_{i-1}, x'_{i-2}, x'_{i-1}) + L(u'_{i-1})$.

If $x''_{i-2} = x'_{i-2}$, we have $L(u''_{i-1}) = L(u'_{i-1})$. It follows

$$D(i-1, c_{i-1}, x''_{i-2}, x''_{i-1}) > D(i-1, c_{i-1}, x'_{i-2}, x'_{i-1}),$$

which contradicts the condition that $D(i-1, c_{i-1}, x'_{i-2}, x'_{i-1})$ is the maximum likelihood whose cost is $c_{i-1}$, and the last two values are $x'_{i-2}(= x''_{i-2}), x'_{i-1}(= x''_{i-1})$.

The correctness of the recurrence equation is proved.

Given the recurrence equation in formula (4), we need $O(n\theta^2\delta)$ space to maintain $D(i, c_i, x'_{i-1}, x'_i)$. By considering $2\theta + 1$ candidates for $x'_{i-2}$ in $[x_{i-2} \pm \theta_{i-2}]$, in each calculation of $D(i, c_i, x'_{i-1}, x'_i)$, the dynamic programming runs in $O(n\theta^3_{\max}\delta)$ time. □

Table 3: Notations

| Symbol | Description |
|---|---|
| $x$ | a sequence of $n$ data points |
| $x'$ | repair of sequence $x$ |
| $x[i]$ or $x_i$ | value of $i$-th data point in $x$ |
| $x_{i \dots j}$ | a subsequence of $x$ from $i$-th to $j$-th data points |
| $t_i$ | timestamp of $i$-th data point |
| $\theta$ | error range of each data point |
| $\delta$ | cost budget for repairing |
| $u_i$ | speed change before and after $i$-th data point |
| $P$ | probability distribution of speed changes |
| $p_{\max}, p_{\min}$ | max/min probabilities in distribution $P$ |
| $L(x)$ | The likelihood of a sequence $x$ w.r.t. speed changes (Equation 2) |
| $D(i, c_i, x'_{i-1}, x'_i)$ | The maximum likelihood of the subsequence $x_{1 \dots i}$, whose cost is $c_i$ and the last two values are $x'_{i-1}, x'_i$, respectively (Equation 4) |
| $C(i, l_i, x'_{i-1}, x'_i)$ | The minimized cost of subsequence $x_{1 \dots i}$, whose likelihood is $l_i$ and the last two values are $x'_{i-1}, x'_i$, respectively (Equation 8) |

## *Proof of Proposition 3*

*Proof.* We show that

$$\begin{aligned} L(x') &= L(x'_{1 \dots i}) + L(x'_{i-1 \dots n}) \\ &= D(i, c_i, x'_{i-1}, x'_i) + L(x'_{i-1 \dots n}) \\ &\leq D(i, c_i, x'_{i-1}, x'_i) + L^u(x_{i-1 \dots n}) \\ &\leq L^w(x) \leq L(x^*). \end{aligned}$$

That is, any solution $x'$ with this $D(i, c_i, x'_{i-1}, x'_i)$ is not optimal and can be pruned. The conclusion is proved. □

## *Proof of Proposition 4*

*Proof.* According to formula (6), the total number of $c'_i$ to consider is $\frac{\delta}{H} = d$. Similarly, referring to formula (7), the total number of $x'_i$ to consider is $2\lfloor \frac{\theta_i}{H} \rfloor + 1$. Since $\delta$ is the bound of repair cost, it follows the maximum number of $x'_i$ to consider, $2\lfloor \frac{\delta}{H} \rfloor + 1 = 2d+1$. Considering all $D(i, c'_i, x'_{i-1}, x'_i)$, the dynamic algorithm runs in $O(n(2d+1)^3 d)$ time with $O(n(2d+1)^2 d)$ space. The conclusion is proved. □

## *Proof of Proposition 5*

*Proof.* According to formula (9), for any $u'_i$, we have

$$L(u'_i) \leq (L'(u'_i) + 1) \cdot K.$$

**Table 4: Significance on STOCK with error number 600, data size 1282, $\theta = 5$, $\delta = 1800$**

| $p$-value<br>significant | DPL | SG | SCREEN | QP |
|---|---|---|---|---|
| DPL | - | 0.040 | 8.238E-10 | 3.989E-6 |
| SG | yes | - | 2.427E-10 | 2.201E-4 |
| SCREEN | yes | yes | - | 1.784E-11 |
| QP | yes | yes | yes | - |

**Table 5: Significance on GPS with error number 360, data size 4712, $\theta = 38$, $\delta = 660$**

| $p$-value<br>significant | DPL | SG | SCREEN | QP |
|---|---|---|---|---|
| DPL | - | 1.348E-5 | 1.575E-8 | 7.835E-8 |
| SG | yes | - | 1.935E-5 | 4.282E-10 |
| SCREEN | yes | yes | - | 3.340E-15 |
| QP | yes | yes | yes | - |

**Table 6: Significance on ENGINE with error number unknown, data size 464, $\theta = 20$, $\delta = 1000$**

| $p$-value<br>significant | DPL | RAW | SG | SCREEN | QP |
|---|---|---|---|---|---|
| DPL | - | 1.6E-12 | 0.015 | 2.2E-7 | 0.013 |
| RAW | yes | - | 1.1E-14 | 0.011 | 5.3E-6 |
| SG | yes | yes | - | 2.434E-6 | 0.007 |
| SCREEN | yes | yes | yes | - | 2.0E-6 |
| QP | yes | yes | yes | yes | - |

Let $x^*$ denote the optimal solution. We have

$$\begin{aligned} L(x^*) - K \cdot L'(x^*) &= \sum_{i=2}^{n-1} \left( L(u_i^*) - K \cdot L'(u_i^*) \right) \\ &\leq \sum_{i=2}^{n-1} K \\ &= (n-2) \cdot K \end{aligned}$$

It follows

$$\begin{aligned} L(x') &\geq K \cdot L'(x') \\ &\geq K \cdot L'(x^*) \\ &\geq L(x^*) - (n-2) \cdot K \\ &= L(x^*) + (n-2) \cdot \varepsilon \cdot \log p_{\max} \\ &\geq (1+\varepsilon) \cdot L(x^*) \end{aligned}$$

For each $i$, the range of likelihood that needs to be considered is

$$l^{end} - l^{start} = (i-2) \cdot \log \frac{p_{\max}}{p_{\min}}.$$

With the approximate likelihood $L'$ defined in formula (9), only a finite number of approximate likelihoods that need to be considered for each $i$, i.e.,

$$\begin{aligned} \lfloor \frac{(i-2) \cdot \log \frac{p_{\max}}{p_{\min}}}{K} \rfloor &= (i-2) \lfloor - \frac{\log \frac{p_{\max}}{p_{\min}}}{\varepsilon \cdot \log p_{\max}} \rfloor \qquad (14) \\ &= (i-2) \lfloor \frac{1}{\varepsilon} ( \frac{\log p_{\min}}{\log p_{\max}} - 1) \rfloor \end{aligned}$$

**Table 7: Significance on SYNTHETIC-clean with error number 0, data size 500, $\theta = 5$, $\delta = 225$**

| $p$-value significant | DPL | SG | SCREEN | QP |
|---|---|---|---|---|
| DPL | - | 8.349E-11 | 2.331E-12 | 1.315E-7 |
| SG | yes | - | 3.734E-13 | 7.495E-16 |
| SCREEN | yes | yes | - | 5.876E-19 |
| QP | yes | yes | yes | - |

**Table 8: Significance on SYNTHETIC-inject with error number 150, data size 500, $\theta = 5$, $\delta = 225$**

| $p$-value significant | DPL | SG | SCREEN | QP |
|---|---|---|---|---|
| DPL | - | 0.002 | 1.333E-11 | 3.998E-5 |
| SG | yes | - | 8.039E-11 | 8.179E-6 |
| SCREEN | yes | yes | - | 3.297E-12 |
| QP | yes | yes | yes | - |

**Table 9: Significance on SYNTHETIC-clean prediction with error number 0, data size 500, $\theta = 5$, $\delta = 150$**

| $p$-value significant | DPL | RAW | SG | SCREEN | QP |
|---|---|---|---|---|---|
| DPL | - | 0.005 | 2.7E-5 | 4.4E-10 | 1.1E-6 |
| RAW | yes | - | 7.6E-20 | 4.4E-12 | 1.1E-12 |
| SG | yes | yes | - | 4.0E-11 | 8.5E-9 |
| SCREEN | yes | yes | yes | - | 1.4E-10 |
| QP | yes | yes | yes | yes | - |

**Table 10: Significance on SYNTHETIC-inject prediction with error number 150, data size 500, $\theta = 5$, $\delta = 600$**

| $p$-value significant | DPL | RAW | SG | SCREEN | QP |
|---|---|---|---|---|---|
| DPL | - | 7.8E-34 | 4.3E-26 | 3.2E-9 | 7.4E-4 |
| RAW | yes | - | 2.1E-27 | 4.4E-12 | 1.7E-12 |
| SG | yes | yes | - | 1.9E-10 | 6.3E-9 |
| SCREEN | yes | yes | yes | - | 2.3E-9 |
| QP | yes | yes | yes | yes | - |

It concludes $O(n^2\theta^3_{\max})$ time and $O(n^2\theta^2_{\max})$ space. □

### Proof of Proposition 6

*Proof.* For any repair $x'$ with $C(i, l_i, x'_{i-1}, x'_i)$, we have

$$\begin{aligned} L(x') &= L(x'_{1\dots i}) + L(x'_{i-1\dots n}) \\ &= l_i + L(x'_{i-1\dots n}) \\ &\le l_i + L^u(x_{i-1\dots n}) \\ &\le L^w(x) \le L(x^*). \end{aligned}$$

That is, $x'$ is not optimal and can be safely pruned. The conclusion is proved. □

### Proof of Corollary 7

*Proof.* According to formula (9), for any $u'_i$, we have

$$L(u'_i) \le (L'(u'_i) + 1) \cdot K.$$

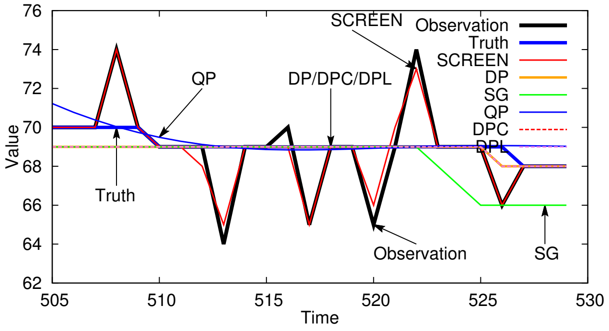

**Figure 23: Case study on STOCK repair results**

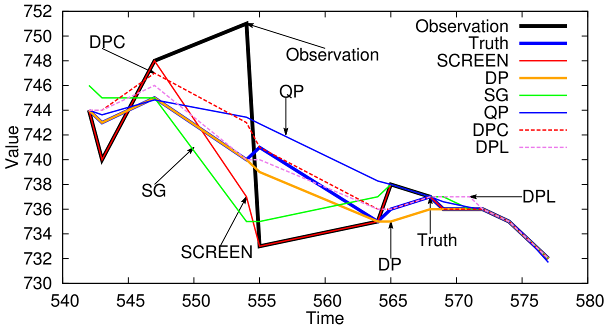

**Figure 24: Case study on GPS repair results**

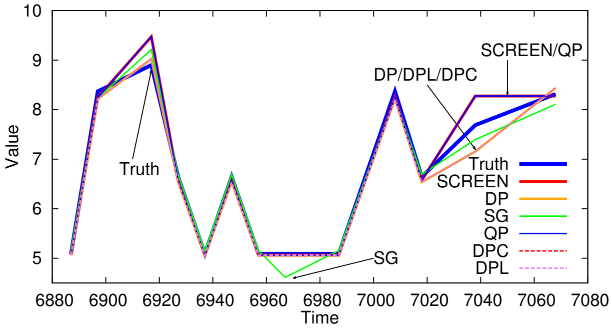

**Figure 25: Case study on ENGINE prediction results**

For any repair $x'$ with $C(i, l'_i, x'_{i-1}, x'_i)$, we have

$$\begin{aligned} L(x') &= L(x'_{1\dots i}) + L(x'_{i-1\dots n}) \\ &= \sum_{j=2}^{i-1} L(u'_j) + L(x'_{i-1\dots n}) \\ &\le \sum_{j=2}^{i-1} K \cdot \big(L'(u'_j) + 1\big) + L(x'_{i-1\dots n}) \\ &= K \cdot l'_i + K \cdot (i-2) + L(x'_{i-1\dots n}) \\ &\le K \cdot (l'_i + i - 2) + L^u(x_{i-1\dots n}) \\ &\le L^w(x) \le L(x^*). \end{aligned}$$

That is, $x'$ is not optimal and can be safely pruned. The conclusion is proved. □

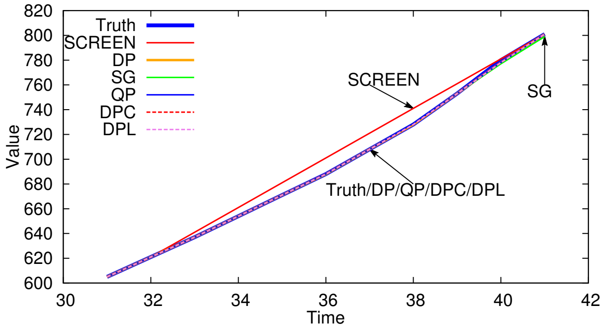

**Figure 26: Case study on SYNTHETIC-clean repair results**

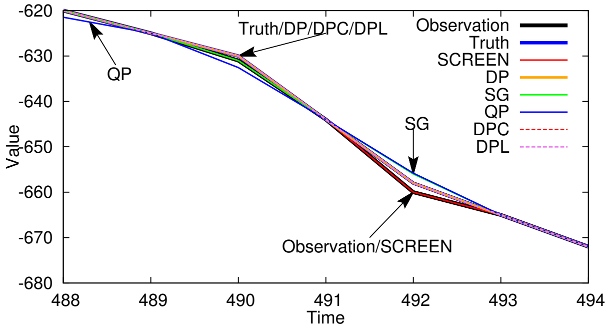

**Figure 27: Case study on SYNTHETIC-injected repair results**

### *Proof of Proposition 8*

*Proof.* According to formula (12), we have

$$\begin{aligned} L(x) &= \sum_{i=2}^{n-1} \log P(u_i) \\ &= \sum_{i=2}^{n-1} \log \left( \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{u_i^2}{2\sigma^2}} \right) \\ &= \sum_{i=2}^{n-1} \left( \log \frac{1}{\sigma\sqrt{2\pi}} - \frac{u_i^2}{2\sigma^2} \log e \right) \\ &= (n-2) \log \frac{1}{\sigma\sqrt{2\pi}} - \frac{\log e}{2\sigma^2} \sum_{i=2}^{n-1} u_i^2 \end{aligned}$$

It follows $L(x') - L(x'') \geq 0$ if and only if $\sum_{i=2}^{n-1}(u_i')^2 - \sum_{i=2}^{n-1}(u_i'')^2 \leq 0$, where $u_i', u_i''$ are speed changes w.r.t. $x_i', x_i''$, respectively. □

### *Proof of Proposition 9*

*Proof.* The greedy computation terminates in at most $\delta$ iterations. By amortizing the likelihood values over a constant domain, data points with low likelihood could be directly obtained. The algorithm runs in $O(\max(n, \delta))$ time. □

## B. STATISTICAL SIGNIFICANCE ON EXPERIMENTS

In order to demonstrate that the differences in algorithms are real in all the experiments, we report the statistical significance calculations [12]. Consider the null hypothesis that two compared algorithms are the same. Let the significance level be $\alpha = 0.05$. We run each experiment 10 times, and conduct the Student's Paired t-test by using APACHE math lib[4]. Tables 4, 5, 6, 7, 8, 9 and 10 report the calculated $p$-values of all the experiments, as well as the corresponding determination of whether the null hypothesis should be rejected or not. As shown, all the results are statistically significant, i.e., rejecting the null hypothesis with $p < 0.05$.

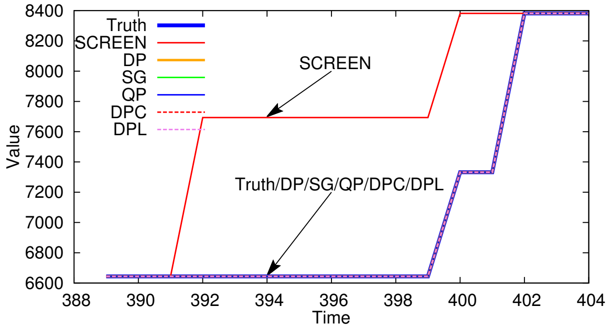

**Figure 28: Case study on SYNTHETIC-clean prediction results**

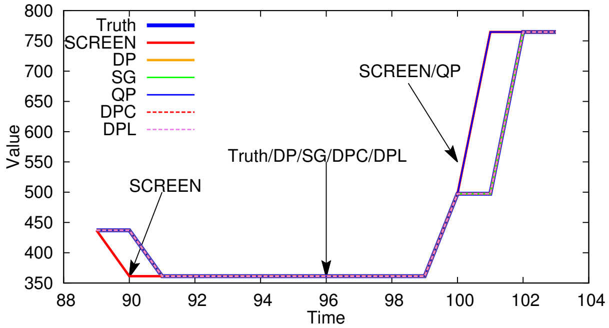

**Figure 29: Case study on SYNTHETIC-injected prediction results**

## C. CASE STUDY ON EXPERIMENTS

To show the differences between the competing approaches, Figures 23, 24, 25, 26, 27, 28 and 29 present case studies on all datasets. (1) The prior method SCREEN cannot capture "small" errors, e.g., as observed at time 508 in Figure 23, since it satisfies the max/min speed constraints. (2) While SCREEN successfully detects the large spike error at time 520 in Figure 23, the max/min speed based repair makes the result only a bit closer to the truth, not as close as our proposals (DP, QP, SG, etc.). (3) Generally, the exact DP shows better performance than the approximate QP and SG, e.g., in Figure 24. (4) The difference of methods in prediction application in Figure 25 is not as significant as the results directly on repairs, which verifies the accuracy study in Figure 17. (5) For the experiments on SYNTHETIC-clean data in Figure 26, almost all the methods propose to not (or slightly) modify the data.

[4] https://commons.apache.org/proper/commons-math/