APPENDIX

A. Proofs

1) Proof of Proposition 1:

Proof Sketch. The sum of distances is easy to check in polynomial time, i.e., the problem is in NP. To prove the NP-Hardness, we show a reduction from the decision version of the Traveling Salesman Problem (TSP), one of Karp's 21 NP-Complete problems [26].

We first introduce the decision version of the TSP problem: Given an undirected weighted graph $G = (V, E)$ with n vertices and their pairwise distances, the problem is to decide whether there exists a path of total weight less than or equal to L, which visits each vertex exactly once. In addition, if all the vertices are in a plane (i.e., the distances satisfy the triangle inequality), existing studies also prove that the TSP problem is still NP-Complete under l_1 and l_2 distances [18], [22], [33].

Given an instance of the TSP problem in a plane, we then construct an instance of our Master Data Ordering Problem with 2 dimensions: For each vertice $v_i \in V$, we use $(v_i[1], v_i[2])$ to denote its coordinates, and then construct a tuple $y_i = (v_i[1], v_i[2]) \in Y$ corresponding to v_i. Therefore, the distances between vertices v_i, v_j are also equal to the distances between tuples y_i, y_j.

Following Problem 3, the decision version of our problem is to decide if there exists a master data order with the sum of the distances less than or equal to K. Let $K = L$, we prove that these two instances are equivalent. Note that the distances of two instances are equal, to find a path of total weight less than or equal to L in TSP problem, we then order the tuples following the order of the vertices in the path. Therefore, the path in TSP has total weight less than or equal to L, if and only if there exists an order of tuples with the sum of the distances less than or equal to K. To conclude, the NP-completeness of the decision problem is proved. □

2) Proof of Lemma 2:

Proof. First, we consider the distance between x and y in the j-th dimension. When $Y_i^{min}[j] < x[j] < Y_i^{max}[j]$, $x[j]$ is covered by the range of $Y_i[j]$. Thereby, $\theta_j(x, Y_i) = 0$ in this case. When $x[j] < Y_i^{min}[j]$, $x[j]$ is out of the value range of the j-th dimension. Then the value closest to $x[j]$ is $Y_i^{min}[j]$, which gives the minimum distance $|x[j] - Y_i^{min}[j]|$. Similarly, we could get the distance bound of the j-th dimension when $x[j] > Y_i^{max}[j]$. In summary, we have

$$|x[j] - y[j]| \geq \theta_j(x, Y_i), \quad \forall y \in Y_i, \quad 0 \leq j \leq l$$

The distance $\delta(x, y)$ is thus bounded by $\theta(x, Y_i)$, for $y \in Y_i$.

$$\delta(x, y) = \sum_{j=1}^{l} |x[j] - y[j]| \geq \sum_{j=1}^{l} \theta_j(x, Y_i) = \theta(x, Y_i)$$

□

3) Proof of Proposition 3:

Proof. Suppose that there exists a neighbor y^* of x in cluster Y_i, i.e., $\delta(x, y^*) \leq \delta_\kappa$. On the other hand, according to the Lemma 2, $\forall y \in Y_i, \delta(x, y) \geq \theta(x, Y_i)$. It follows $\delta(x, y^*) \geq \theta(x, Y_i)$. We thus obtain $\theta(x, Y_i) \leq \delta_\kappa$. □

4) Proof of Proposition 4:

Proof. The proposition can be proved similarly following the proof of Proposition 3. □

5) Proof of Proposition 6:

Proof. Note that the smoothing tuples in our algorithm are selected from the valid modification set V_i. Thereby, the modification x_i' of x_i exists if and only if V_i is not empty. We discuss two cases of window W_i.

If $W_i = \emptyset$, the smoothness constraint $\mathcal{S}$ of x_i is naturally satisfied. Therefore, we can always find a candidate from $\text{Nei}(x_i)$ in the master data to modify x_i safely.

If $W_i \neq \emptyset$, then the candidate modification set C_i involves $\text{Nei}(x_j')$, for $x_j' \in W_i$, i.e., $x_j' \in C_i$ as well. Due to the streaming computation, x_j is smoothed before x_i. That is, x_j' in the window satisfies the smoothness constraint with each other, referring to the previous smoothing steps. Therefore, there always exists $x_j' \in W_i$ as the valid modification of x_i. □

6) Proof of Proposition 7:

Proof. For window size $\omega = 0$, the temporal smoothness of the time series is no longer considered. The window W_i of each tuple x_i is always empty and the candidate set C_i is $\text{Nei}(x_i)$. Since the modifications no longer affect each other, we have $V_i = C_i$. The one with the minimum distance to x_i in V_i returned by Algorithm 1 is indeed the optimal solution. □

B. Algorithms

Algorithm 3: Master Data Ordering

```
Input: cluster $Y_i$ of master data with $l$ columns
Output: ordered data $Y_i'$
$\{o_1, \ldots, o_l\} \leftarrow$ columns in decrease order of variances;
for $y_p \in Y_i$ do
    for $y_q \in Y_i$, $p < q$ do
        need_swap $\leftarrow$ False;
        for index $\leftarrow 1$ to $l$ do
            if $y_p[o_{index}] > y_q[o_{index}]$ then
                need_swap $\leftarrow$ True;
                break;
            if $y_p[o_{index}] < y_q[o_{index}]$ then
                need_swap $\leftarrow$ False;
                break;
        if need_swap then
            swap($y_p$, $y_q$);
return $Y_i'$;
```

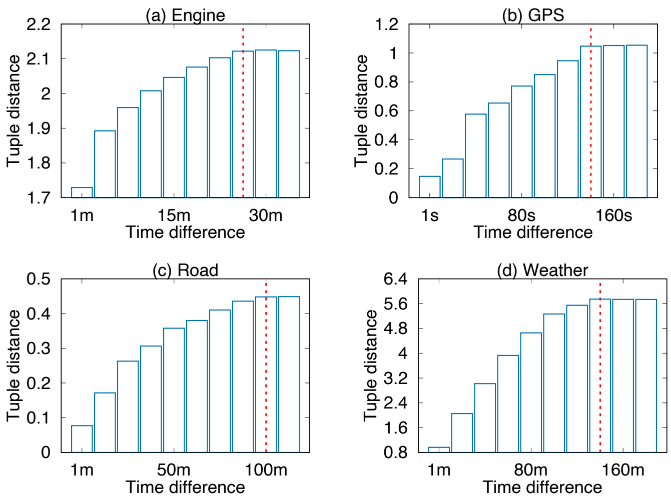

Fig. 14: Determining window size ω

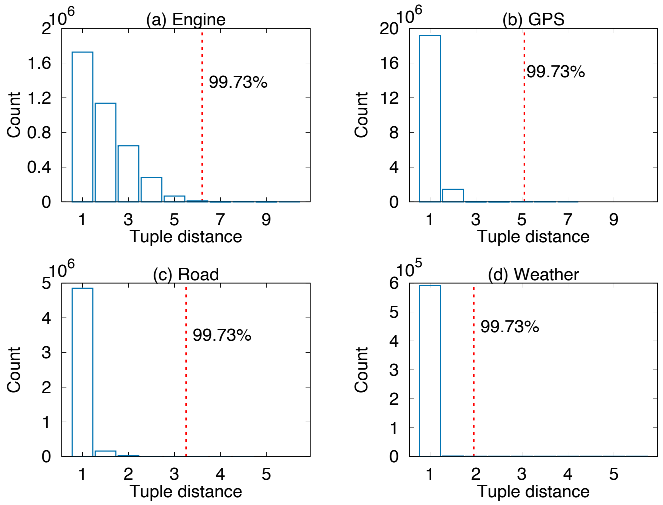

Fig. 15: Determining distance threshold η

C. Parameter Determination Experiments

We discuss below how to determine the parameters of our smoothing algorithm in practice, i.e., window size ω, distance threshold η, and the number of neighbors κ.

1) Determining Window Size ω: A smaller window size ω means a weaker smoothness constraint. On the other hand, a too large window size ω is not necessary, since the values of time series are often stable as the examples shown in Figure 1.

In this sense, we observe the distances of two tuples with various time differences. As illustrated in Figures 10a-c, if the median tuple distance no longer increases by further enlarging the time difference, such a time difference is sufficient as the window size ω.

As shown in Figure 10a Engine, the tuple distance does not increase largely for a time difference greater than 20m. Besides, The corresponding RMSE and time cost are also pleasant in Figure 10a. Thereby, the window size $\omega = 20$ is appropriate.

2) Determining Distance Threshold η: Another parameter in the smoothness constraint is on the distances of tuples in a window. For a too small distance threshold η, it may be over-constrained, leading to over-change in smoothing. On the other hand, the smoothness constraint with too large η has no effect, since most data satisfy it including errors.

Intuitively, we may observe the distance distribution of tuples in windows, in Figures 11a-c, and determine the distance threshold η by the three-sigma rule of thumb [35].

For instance, Figure 11a suggests a distance threshold η about 6 in Engine. The corresponding RMSE is indeed low in Figure 11d. Similar results are also observed in the other two datasets, demonstrating the effectiveness of determining η.

3) Determining # Neighbors κ: The greater the number κ of neighbors from the master data, the higher the likelihood that accurate smoothing will be included in the candidate set. The corresponding time cost will be higher as well, as illustrated in Figure 12e.

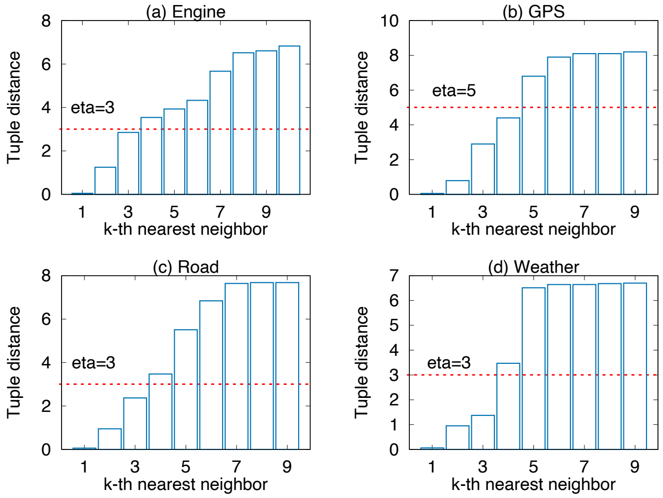

Fig. 16: Determining # neighbors κ

To determine an appropriate κ, we observe the tuple distance of the κ-th nearest neighbor in the master data, in Figures 12a-c. To satisfy the smooth constraint, the suggested candidates of κ-nearest neighbors are expected to have distances within η.

For instance, in Figure 12a, the 3rd nearest neighbors have distances less than $\eta = 6$. That is, a number of neighbors $\kappa = 3$ is sufficient. Similarly, Figures 12b and c suggest $\kappa = 5$ and 3, respectively, leading to low RMSE in Figure 12d.