Exploring Spatio-Temporal Context Dependency for Time Series Data

Xiaoou Ding*, Yingze Li†, Hongzhi Wang*, Chen Wang‡, Qiang Ye§
*†School of Computer Science and Technology, Harbin Institute of Technology.
‡National Engineering Laboratory for Big Data Software, EIRI, Tsinghua University.
§School of Management, Harbin Institute of Technology.
Email: *{dingxiaoou,wangzh}@hit.edu.cn, †1190202126@stu.hit.edu.cn, ‡wang_chen@tsinghua.edu.cn
§yeqiang@hit.edu.cn

***Abstract*—Time series data generated by intelligent devices are suffering serious data quality problems. Though data dependencies have been applied in error detection and data repairing tasks, data dependencies are still insufficient for data quality representation of time series data. Considering the obvious characteristics of time series data about the ordered time window and strong associations from both spatial and temporal aspect, we propose Spatio-Temporal Context Dependency (STCD), one kind of data quality constraints that express inherent context dependencies locating in both temporal and spatial dimension, which expends the data quality semantic expression of time series data. We analyze the complexity of the implication and consistency problems for STCD reasoning, and develop STCDiscover method to discover effective STCDs from data, which consists of functional structure discovery, feasible bound determination, and STCD pattern validation. Experimental results on three real-life datasets and one synthetic dataset verify STCDiscover efficiently discovers high-quality STCD patterns. We also compare the performance of the proposed STCD and the representative data quality constraints with column-only dependencies and row-only constraints dependencies in the error detection tasks for time series data. The results show that detection method with STCD has an average of 12% higher accuracy and 30% higher F1 value than the detection algorithms with other dependencies.**

***Index Terms*—data cleaning, data dependency, numerical data quality management**

I. Introduction

With the wide application of intelligent sensors, data is being accumulated at an unprecedented rate. Sequential data, including multiple time series, are defined as sequences of values that are *continuously* measured and recorded at a specific time or fixed period [1], [2]. The data quality problems of numerical sequence data has drawn more attention from both academia and industry [1], [3]. Data quality problems are seriously prevalent in time series data [4], [5]. Considering the characteristics of the data, it suffers from various types of data errors, *e.g.,* the single error [2], [6], continuous error [1], and contextual error [7], etc.

Researchers have derived constraint forms from functional dependencies (FD) with either *conditional* or *probabilistic* perspectives, and proposed CFD and PFD, which indicate various dependence relationships among multiple attributes in the database [8]. Further, denial constraints (DC) [9] are introduced a form of predicate combination that extends the expressiveness of data quality constraints for categorical data and numerical data. For time series data, Order dependencies (OD) [10] are introduced to express the constraints with different orders, and sequential dependencies (SD) [11] generalize OD to express interesting relationships between ordered determinant attributes and distances on dependent attributes.

TABLE I
Data quality requirement descriptions for time series data

D1. The high oil temperature HTemp should be higher than the low oil temperature LTemp.
D2. Water temperature WTemp_1 increases from hour to hour by at least 1 points and at most 2.5 points.
D3. The changing speed of tank pressure TPres_1 should be within $[-2, 2]$.
D4. WTemp_2 is linearly correlated to the temperature of device ID-1.
D5. APower is approximately equal to $0.7 \cdot \mathsf{U_A} \cdot \mathsf{I_A}$, with the related error no more than 10 points.
D6. The change speed of WTemp_3 is about half as fast as the change speed of gas flow Gflow_3.

Though various data quality constraints have been widely applied in data quality management tasks, it is still insufficient for data quality representation of the time series data. There still remains a number of data quality constraints that cannot captured by the existing data dependencies as presented in the motivated example below.

Example 1: Given part of time series data of a sensor group from a fossil-fuel power station (see Figure 5). According to domain knowledge, data in the current operating condition are expected to meet the following data quality requirements, listed in Table I. We summarize three kinds of observations: ***O1***: Descriptions 1-3 could be formalized into data quality constraints, while *D4-D6* are challenged to explicitly expressed by the existing data dependencies, as presented in Table II. ***O2***: These descriptions cover the dependencies from temporal and spatial perspective, where *D1*,*D4*,*D5* focus on the constraints locating in multiple attributes from the same rows, and *D2*,*D3* reflect the dependencies from the time context, while *D6* involves dependencies from both attributes and the time context. ***O3***: Considering *D5*, data quality constraints are sometimes expected to tolerate relaxation in a certain range of values considering the margin of error or uncertainty in data. ***O4***: Considering another description "*D7* The changing speed of tank pressure TPres_1 should be within $[-8, 8]$", it is found that any instance satisfying *D2* must satisfy *D7*. That means, for any data quality constraints set Σ containing *D2*, it is not necessary for Σ to involve *D7* for concision issues. ■

From the above, it highly requires the extended data dependencies for time series data for three critical aspects: **First**, most of the constraints fails to consider the data quality issues from the perspective of time window, and cannot describe function operations. Though DCs extends the expressiveness of FDs and allow predicates $(>, <)$, it does not involve operators $(+, -, \times, \div)$. **Second**, considering the obvious characteristics of numerical data compared to other types of data, it requires the data quality evaluation mechanism from a certain value range rather than a clear value. **Third**, the existing data quality constraints usually focus on data dependencies on *one* type of description objects, *i.e,* either the dependence on attributes or features (columns in a database table), or the dependence on instances (rows) [12], and fail to consider the dependency patterns of time series data from the aspects of spatio-temporal dimensions.

However, it cannot be ignored that the richer expressiveness of the data dependencies brings more challenges in *i*) inference system for a novel data dependency, and *ii*) the explosive growth of search space in dependency pattern discovery [9], thus, it is challenged to automatically obtain aforementioned dependencies for time series. Motivated by this, we propose Spatio-Temporal Context Dependency (STCD) to improve the applicability of data dependencies for time series data. Our **contributions** in this paper are summarized as follows.

1. (*STCD notation definition*). We propose a novel data dependency notation, spatio-temporal context dependency (STCD), which extends the expressiveness of data quality constraints from inherent context dependencies in both spatial (column) and temporal (rows) dimension for time series data. We address the compatibility of STCD with typical column-based constraints and row-based constraints (see Section II).

2. (*STCD inference*). We study the complexity of the implication and consistency problems for STCD reasoning (Section III), and prove that the complexity of both determination problems of STCDs with general functions are NP-Hard, while there exists PTIME algorithms for STCDs composed of a particular kind of functions, such as linear functions. We introduce the complexity conclusions of both problems for different STCD functions.

3. (*STCD discovery*). We propose STCD discovery problem and provide the solution STCDISOCVER, which consists of functional structure discovery, feasible bound determination, and STCD pattern validation phases (Section IV). We prove that the proposed STCDISOCVER enables interval estimation for one discovered STCD with $1-\delta$ confidence.

4. We conduct experiments on three real dataset and one synthetic dataset in Section V. The result shows that STCDISOCVER is effective in mining dependency patterns from time series data, which removes 14%-42% of the redundant STCD patterns. We compare the performance of STCD in error detection tasks for time series data with representative existing data dependencies, and report that detection with STCD has an average of 12% higher accuracy and 30% higher F1 value than the detection algorithms with other dependencies. **Our extended paper version, codes and dataset samples are available at** Github[1].

II. Overview of STCD

In this section, we first formally introduce the spatio-temporal context dependency (STCD), then discuss the expressiveness of STCD for both dependencies on rows and columns.

A. Spatio-Temporal Context Dependency

$I = \{S_1, ..., S_M\}$ is a M-dimensional numerical sequence data. $S = \langle s_1, ..., s_N \rangle$ is a sequence on instance I. $s_n = \langle x_n, time_n \rangle, (n \in [1, N])$, where x_n is a real-valued number with a time point $time_n$. For brevity, let t_i denote one tuple of I, *i.e.,* $t_i = \langle S_1[i], S_2[i], S_3[i], \ldots, S_M[i] \rangle$.

***Definition** 1:* **Spatio-Temporal Context Dependency (STCD)**. Given a window size w, a STCD φ defined over schema $\mathcal{R}$ of time series data is

$$\varphi : \forall t_i \in \mathcal{R}, ([t_{v_1}.A, t_{v_2}.B, \ldots, t_{v_{n-1}}.Q] \rightarrow^{f}_{(l,u)} t_{vn}.P, f),$$

where $A, B, \ldots, P, Q \in Attr(\mathcal{R})$ and $\forall j \in [1, n], i \leq v_j \leq i+\mathsf{w}$. Let the LHS of φ *i.e.*, $[t_{v_1}.A, t_{v_2}.B, \ldots, t_{v_{n-1}}.Q]$ as X, and the RHS of φ as Y, then it has $\varphi : \left(X \rightarrow^{f}_{(l,u)} Y, f\right)$. Here, $f : \mathbb{R}^{(X)} \rightarrow \mathbb{R}$ is a fixed real-valued function, which takes the set X as the independent variables and maps them to a real value which is close to Y, and the distance between $f(X)$ and Y locates in a tolerable range, *i.e.*, $f(X) - Y \in (l, u)$. An instance I of R satisfies φ if $\forall t_i \in I$, $f(X) - Y \in (l, u)$, denoted by $I \models \varphi$.

Considering the characteristic of the numerical data, we extend the expression of FD and apply a real-valued function f in STCD φ which are derived from domain experts or learned from business knowledge. And we allow relaxation in the satisfaction determination of $f(X) - Y \in (l, u)$ rather than extremely let $f(X) = Y$. Such STCD syntax is much more appropriate for numerical sequence data, especially for data from real IoT scenarios. Given a STCD $\varphi = (X \rightarrow^{f}_{(l,u)} Y, f)$, the negative form of $\varphi, \neg\varphi$ is: $\neg\varphi = (X \rightarrow^{f}_{(-\infty,l)\cup(u,+\infty)} Y, f)$. $\neg\varphi$ indicates that it is always the opposite of φ for any case.

According to Ref. [12], which has classified four kinds of data quality constraints by their dependence on attributes (columns in a database table) and entities or instances (rows), we conclude the semantics expressed by STCD for the four constraint types in Definition 2. It shows that the proposed STCD is sufficiently expressive enough to cover the constraints in each quadrant. Figure 1 depicts the representation of the four types of constraints in [12] with different STCD semantic.

***Definition** 2:* A STCD $\varphi = (X \rightarrow^{f}_{(l,u)} Y, f)$ over a window w is classified into the following categories: 1) φ belongs to **single-row&single-column** constraints iff. $\mathsf{w} = 0$ and $Attr(X) = Attr(Y), |Attr(X)| = 1$, 2) φ belongs to **single-row&multi-column** iff. $\mathsf{w} = 0$ and $\exists A_i, A_j \in Attr(X \cup Y), A_i \neq A_j$, 3) φ belongs to **multi-row&single-column** iff. $\mathsf{w} \geq 1$ and $Attr(X) = Attr(Y), |Attr(X)| = 1$,

[1]https://github.com/LiYingZe/STCD-Spatio-Temporal-Context-Dependency

TABLE II
FORMALIZATION OF STCDS

ID	Existing Constraints	STCD
(1)	Variable DC: $\forall t_i, \neg(t_i.\mathsf{HTemp} > t_i.\mathsf{LTemp})$	$\forall t_i, (t_i.\mathsf{LTemp} \rightarrow^{f}_{(-\infty,0)} t_i.\mathsf{HTemp}, f(t_i.\mathsf{LTemp}) = t_i.\mathsf{LTemp})$
(2)	SD: Time $\rightarrow_{[1,2.5]}$ WTemp_1	$\forall t_i, (t_i.\mathsf{WTemp}_1 \rightarrow^{f}_{(1,2.5)} t_{i+1}.\mathsf{WTemp}_1, f(t_i.\mathsf{WTemp}_1) = t_i.\mathsf{WTemp}_1)$
(3)	SC: $-2 < \frac{t_{i+1}.\mathsf{TPres}_1 - t_i.\mathsf{TPres}_1}{t_{i+1}.T - t_i.T} < 2$	$\forall t_i, (t_i.\mathsf{TPres}_1 \rightarrow^{f}_{(-2,2)} t_{i+1}.\mathsf{TPres}_1, f(t_i.\mathsf{TPres}_1) = t_i.\mathsf{TPres}_1)$
(4)	None	$\forall t_i, (t_i.\mathsf{DTemp} \rightarrow^{f}_{(-2,2)} t_i.\mathsf{WTemp}_2, f(t_i.\mathsf{DTemp}) = 2 \cdot t_i.\mathsf{DTemp} + 20)$
(5)	None	$\forall t_i, ([t_i.\mathsf{U_A}, t_i.\mathsf{I_A}] \rightarrow^{f}_{(-10,10)} t_i.\mathsf{APower}, f(t_i.\mathsf{U_A}, \mathsf{I_A}) = 0.7 \cdot t_i.\mathsf{U_A} \cdot t_i.\mathsf{I_A})$
(6)	None	$\forall t_i, (t_i.\mathsf{WTemp}_3, t_i.\mathsf{Gflow}_3, t_{i+1}.\mathsf{Gflow}_3 \rightarrow^{f}_{(-0.1,0.1)} t_{i+1}.\mathsf{WTemp}_3, f = t_i.\mathsf{WTemp}_3 + \frac{1}{2}(t_{i+1}.\mathsf{Gflow}_3 - t_i.\mathsf{Gflow}_3))$

and 4) φ belongs to **multi-row&multi-column** iff. $\mathsf{w} \geq 1$ and $\exists A_i, A_j \in Attr(X \cup Y), A_i \neq A_j$.

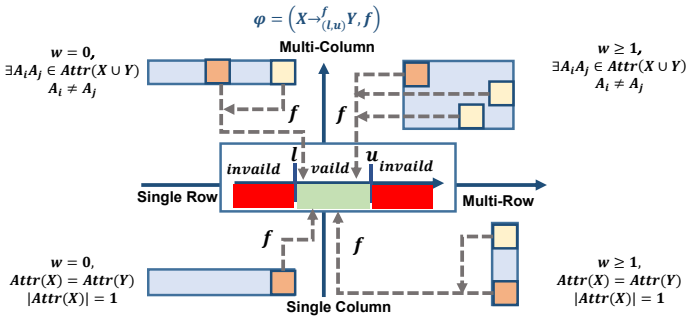


Fig. 1. Demonstration of four types of constraints

B. STCD vs Existing Data Dependencies

According to Definition 2, STCD is expressive to describe dependencies on a single column, a single row, or both. We summarize the compatibility of STCD with the existing typical data dependencies for numerical data in Table III. To take ODs, SDs, and SCs as the dependency instances on *columns*, STCD is compatible of these dependencies with a time window w and one attribute A involved in φ. It achieves the formalization of the *temporal* context dependencies.

For the dependency on *rows*, we take DCs for instance, which is well-known for its expressiveness for covering interesting ICs for either tuple level and table level. STCD is compatible with some special cases of VDCs and CDCs. For any DC φ_{DC} with arbitrary number of predicates, there is always an equivalent STCD φ_{eq} corresponding to φ_{DC}, as DC's conjunctive combination of predicates could be simulated by the product of multiple eigenfunctions. That is, given $\varphi_{DC} : \forall t_i, \neg(\wedge_{j=1}^{k} P_j(t_i))$ where $P_j(t_i) = t_i.A \phi_j t_i.B$, $A, B \in \{Attr(\mathcal{R}), \mathsf{Const}\}$, $\phi_j \in \{\neq, >, <, \leq, \geq, =\}$. Its equivalent STCD is constructed as $\varphi_{eq} = (X \rightarrow^{f}_{(0,0)} Y, f = f_1 \cdot f_2 \cdot \ldots f_k + Y)$ where $\forall P_j \in \varphi_{DC}$, eigenfunctions f_j could be constructed through $P_j's$ comparison operation ϕ_j, satisfying $f_j = 0$ when P_j is false. For example, if $\phi_j = \text{“} \neq \text{”}$, $f_j = t_j.A - t_j.B$; if $\phi_j = \text{“} \geq \text{”}$, $f_j = t_j.B - t_j.A + |t_j.A - t_j.B|$. For any instance $I \models \varphi_{DC}$, at least one P_j is false, therefore $f_j = 0, f(X) = 0 + Y$, $f(X) - Y = 0$, thus it has $I \models \varphi_{eq}$.

The difference between STCD and general DC is that STCD only checks elements in the time window, while general DC validates every possible tuple pair in the relation. Considering the large amount of time series data, *e.g.* to check the monotonicity of thousands of records, it is unrealistic to compare each two tuple pairs, while one could achieve the checking with STCD in a sliding window via once visit of data. It indicates the benefit of STCD for time series.

TABLE III
ENTAILMENTS BETWEEN STCD AND THE EXISTING DATA DEPENDENCIES

Data Dependencies	Equivalent STCD
OD: Time$^{\leq}$ $\rightarrow$ Age$^{\leq}$	$\forall t_i, (t_i.\mathsf{Age} \rightarrow^{f}_{(-\infty,0)} t_{i+1}.\mathsf{Age}, f = t_i.\mathsf{Age})$
SD: Time $\rightarrow_{5,7}$ Price	$\forall t_i, (t_i.\mathsf{Price} \rightarrow^{f}_{(5,7)} t_{i+1}.\mathsf{Price}, f = t_i.\mathsf{Price})$
SC: $\frac{t_{i+1}.x - t_i.x}{t_{i+1}.T - t_i.T} \leq 4$	$\forall t_i, (t_i.x \rightarrow^{f}_{(-\infty,0)} t_{i+1}.x, f = t_i.x + 4 \cdot (t_{i+1}.T - t_i.T))$
CDC: $\neg(t.\mathsf{Tax} \leq 9)$	$\forall t_i, ([t_i.\mathsf{Tax}] \rightarrow^{f}_{(-\infty,9)} t_i.\mathsf{Tax}, f = 2 \cdot t_i.\mathsf{Tax})$
VDC: $\neg(t.\mathsf{Open} \leq t.\mathsf{High})$	$\forall t_i, (t_i.\mathsf{Open} \rightarrow^{f}_{(-\infty,0)} t_i.\mathsf{High}, f = t_i.\mathsf{Open})$

C. Feasible Region Demonstration of STCD

We then visually demonstrate the feasible region of the data *w.r.t* one STCD φ and the STCD set Σ.

Feasible domain of φ. All the data in I satisfying $\varphi = (X \rightarrow^{f}_{(l,u)} Y, f)$ forms the feasible domain of φ, denoted by $\varphi.Fes$. Let each element in $\varphi.X \cup \varphi.Y$ constitute a separate dimension, and the cartesian product of these dimensions forms the space $S_1 = t_{v1}.A \times t_{v2}.B \times \ldots \times t_{vn-1}.Q \times t_{vn}.P$. In this way, $\varphi.Fes$ is expressed directly as the region formed by the intersection of surface $f(\varphi.X) - \varphi.Y - l = 0$ and surface $f(\varphi.X) - \varphi.Y - u = 0$.

Feasible domain of Σ. $\Sigma.Fes$ denotes the feasible domain of Σ, which consists of all the data satisfying the set $\Sigma = \{\varphi_1, \varphi_2, \ldots \varphi_k\}$. Let each element in $\varphi_1.X \cup \varphi_1.Y \cup \varphi_2.X \cup \varphi_2.Y \cup \cdots \varphi_k.X \cup \varphi_k.Y$ constitute a separate dimension, and the cartesian product of these dimensions forms the space S_2'. Since the data satisfying Σ must satisfy any a STCD in Σ, it has $\Sigma.Fes = \varphi_1.Fes' \cap \varphi_2.Fes' \cap \cdots \varphi_k.Fes'$, thus, $\varphi_i.Fes'$ could be obtained by performing a dimension raising procedure from S_i to S' by holding the values of $\varphi_i.Fes$ constant in S_i, while freely varying the others in S'.

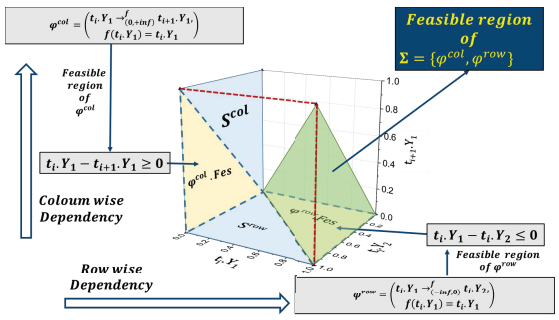


Fig. 2. Demonstration of $\Sigma.Fes$

***Example** 2:* Given two STCD in Σ: $\varphi^{col} = (t_i.Y_1 \rightarrow^{f}_{(0,+\infty)} t_{i+1}.Y_1, f(t_i.Y_1) = t_i.Y_1)$ and $\varphi^{row} = (t_i.Y_1 \rightarrow^{f}_{(-\infty,0)} t_i.Y_2, f(t_i.Y_1) = t_i.Y_1)$. The areas in yellow in Figure 2 present the feasible region $\varphi_{row}.Fes$ and $\varphi_{col}.Fes$. $\Sigma.Fes$ exactly corresponds to the intersection of yellow areas, as shown in green. $I \models \Sigma$ indicates that data values of $t_i.Y_1, t_i.Y_2, t_{i+1}.Y_1$ from I locates in this green area.

III. Analysis of STCD Reasoning

Consider STCD set $\Sigma = \{\varphi_1, \varphi_2, \varphi_3\}$ share the same mapping pattern $(A_1 \rightarrow^{f}_{(l,u)} A_2, f = 2 \cdot A_1)$ and different value domain $\varphi_1.(u,l) = (-inf, 5)$, $\varphi_2.(u,l) = (25, +inf)$, and $\varphi_3.(u,l) = (50, +inf)$. It is obvious that the demand from φ_1 and φ_2 are in conflict, for one tuple could never find an instance I satisfying the both φ_1 and φ_2 at the same time. In addition, if φ_3 exists in Σ, then we do not need φ_2 any longer. Considering such cases, we discuss implication and consistency problems for STCD in Section III-A and Section III-B, and then introduce the difficulties of STCD determination in Section III-C.

A. The Implication Problem

We first discuss the implication issues of STCD in Definition 3. Since Axiom-based approach are applied in the implication reasoning of traditional dependencies, *e.g.,* Armstrong Axioms for FDs [13], however, the mapping function f in STCD for time series data is different from the discrete injective in general dependencies for categorical data. Consider a STCD set with the following form $\Sigma = \{\varphi_i | \varphi_i = (X \rightarrow^{f}_{l_i,u_i} Y, f), i = (1, 2, \ldots, n)\}$, and a new STCD $\varphi_x = (X \rightarrow^{f}_{l_x,u_x} Y, f)\}$. We could convert the implication problem into an equivalent description in Proposition 1 from the perspective of the feasible region.

***Definition** 3:* (**STCD Implication**). Given a set of STCD Σ and data instances I of schema $\mathcal{R}$, Σ implies one STCD φ, *i.e.,* $\Sigma \models \varphi$, if $\forall I \in \mathcal{R}$, if $I \models \Sigma$, then $I \models \varphi$. The STCD implication testing problem is to identify whether one STCD φ could be implicated by a given Σ.

***Proposition** 1: Given Σ, data instances I, and a new STCD φ_x, Σ implies φ_x if and only if $\Sigma.Fes \subseteq \varphi_x.Fes'$,* i.e. *the feasible domain of Σ is a subset of the feasible domain φ_x.*

Proof: (*Necessity*) If Σ implies φ_x, then for any point d in $\Sigma.Fes$, one could construct a small data instance I_d based on this particular d's value which has $I_d \models \Sigma$. According to Definition 3, $I_d \models \Sigma$ shall derives $I_d \models \varphi_x$, which means $d \in \varphi_x.Fes'$. Therefore, $\forall d \in \Sigma.Fes, d \in \varphi_x.Fes'$, the necessity holds. (*Sufficiency*) Similarly, if $\Sigma.Fes \subseteq \varphi_x.Fes'$, then $\forall d \in \Sigma.Fes, d \in \varphi_x.Fes'$. Any $I \models \Sigma$ can be viewed as a group of points in $\Sigma.Fes$, since $\Sigma.Fes \subseteq \varphi_x.Fes'$, these points are in $\varphi_x.Fes'$. This proves that when φ_x is implied by Σ, $\forall I \in \mathcal{R}$, if $I \models \Sigma$, then $I \models \varphi$. The sufficiency holds. ■

Accordingly, we then discuss the symbolic inference system for STCD. (**Reflexivity**) Given Σ and φ_x, if $\exists \varphi_i \in \Sigma$, having $\varphi_i.f = \varphi_x.f$ and $l_x \leq l_i \leq u_i \leq u_x$, then Σ implies φ_x. (**Transitivity**) Given $\Sigma, \varphi_x, \varphi_y$, if Σ implies φ_x and the set $\{\varphi_x\}$ implies φ_y then Σ implies φ_y. (**Augmentation**) Given Σ and φ_x, if $\exists \varphi_i \in \Sigma, \varphi_x = (\varphi_i.X_i \cup \varphi_i.X_{new} \rightarrow^{f}_{(l_i,u_i)} Y_i, f(\varphi_i.X_i \cup \varphi_i.X_{new}) = f_i(\varphi_i.X_i))$, then Σ implies φ_x.

***Proposition** 2: The above inference based on these axioms is **correct but incomplete**.*

The proof of proposition 2 is provided in Appendix A. The reason for the incompleteness is, the axiomatic rules fail to solve the implication determination containing different *function structures*, which makes STCD implication determination problem more challenged when compared with other data dependencies implication determination *e.g.*, CFD, PFD [14]. To solve STCD implication problems, we propose a constraint-based inference in Definition 4.

***Definition** 4:* **The Inference Equation of Implication Testing**: Given STCD set $\Sigma = \{\varphi_i | \varphi_i = (X_i \rightarrow^{f_i}_{l_i,u_i} Y_i, f_i), i = (1, 2, \ldots, n)\}$, the testing objective $\varphi_x = (X_x \rightarrow^{f_x}_{l,u} Y_x, f_x)\}$. The inference equation of implication testing is listed below with two sides in Table IV.

TABLE IV
Implication Inference Equations for STCD

for LHS	for RHS
$z^L = \max(l_x - \varphi_x.f_x(X_x) + Y_x)$	$z^R = \max(\varphi_x.f_x(X_x) - Y_x - u_x)$
$s.t. \forall i = 1, 2 \ldots n$	$s.t. \forall i = 1, 2 \ldots n$
$\varphi_i.f_i(X_i) - Y_i - u_i \leq 0,$	$\varphi_i.f_i(X_i) - Y_i - u_i \leq 0,$
$l_i - \varphi_i.f_i(X_i) + Y_i \leq 0$	$l_i - \varphi_i.f_i(X_i) + Y_i \leq 0$

From the above, one can determine that φ_x could be implied by Σ if $z^L \leq 0$ and $z^R \leq 0$. ■

***Proposition** 3: The above inference is **correct and complete**.*

Due to the limited space, we outline the proof of the completeness issue here, and put the formal proof in Appendix B. The above inference equation assists to make the inference system to be complete for implication of STCDs. For any φ_x implied by Σ, $\Sigma.Fes \subseteq \varphi_x.Fes$. It shows any point in $\Sigma.Fes$ is also in $\varphi_x.Fes$. That means, (1) $\forall \varphi_i \in \Sigma, l_x \leq \varphi_i.f_i(X_i) - Y_i \leq u_i$ and (2) $l_x \leq \varphi_x.f_x(X_x) - Y_x \leq u_x$. Case (2) could be transformed into two optimizing problem: $i).\max(l_x - \varphi_x.f_x(X_x) + Y_x) \leq 0$ and $ii).\max(\varphi_x.f_x(X_x) - Y_x - u_x) \leq 0$. The former one determines $f_x(X_x) - Y_x$ is above the lower bound l_x, and the latter determines $f_x(X_x) - Y_x$ is below u_x. Thus, the implication is converted to the equation on two sides. The determination of the problem is complete if the equations on both sides are solved correctly.

Accordingly, we develop the implication checking method STCD-IC in Algorithm 1. We first merge STCDs into constraints (Lines 1-6), and use the constrained optimization functions to get the maximum value under the constraints (Lines 7-8). We then determine whether both the objectives are less than 0, and if so, φ_x is identified to be implied by Σ. The complexity of Algorithm 1 mainly depends on the complexity of the optimization algorithm adopted in lines 7-8 and the structure of the STCD function. It will be further discussed in Section III-C.

B. The Consistency Problem

For consistency issues, given a set of STCD Σ, schema $\mathcal{R}$, Σ is determined to be consistent if $\exists I \in \mathcal{R}$ and $I \models \Sigma$. We also carry out a similar transformation of programming equation for consistency analysis.

Algorithm 1: IMPLICATIONTEST

Input: STCD set Σ, new STCD φ_x
Output: True, if Σ implies φ_x; False, if not

$Cons \leftarrow \phi$;
foreach $\varphi_i \in \Sigma$ **do**
  $Cons.add(\text{“}\varphi_i.f_i(X_i) - Y_i - u_i \leq 0\text{”})$;
  $Cons.add(\text{“} - \varphi_i.f_i(X_i) + Y_i + l_i \leq 0\text{”})$;
$ObjLeft = \text{“} - \varphi_x.f_x(X_x) + Y_x + l_x\text{”}$;
$ObjRight = \text{“}\varphi_x.f_x(X_x) - Y_x - u_x\text{”}$;
$z^L \leftarrow \max(ObjLeft, constraint \leftarrow Cons)$;
$z^R \leftarrow \max(ObjRight, constraint \leftarrow Cons)$;
if $z^L \leq 0 \wedge z^R \leq 0$ **then**
  return True;
return False;

***Proposition** 4: Given Schema $\mathcal{R}$(Attr, Σ), Σ is consistent if and only if $\Sigma.Fes \neq \phi$.*

Proof sketch: Similar with the proof of implication issues, if the set Σ is consistent, we could one instance $I \in \mathcal{R}, I \models \Sigma$. Since I forms a point d in the space spanned by Σ, it has $d \in \Sigma.Fes$, thus, $\Sigma.Fes \neq \phi$. We show the formal proof in Appendix C. ■

Accordingly, we are able to express Proposition 4 from the perspective of inference equation. Given STCD set $\Sigma = \{\varphi_i = (X_i \rightarrow^{f_i}_{l_i,u_i} Y_i, f_i) | i = (1, 2, \ldots, n)\}$, pick one $x \in \{1, 2 \ldots n\}$, the consistency inference equation could be obtained by the following steps: 1) update $z^L = \min(l_x - \varphi_x.f_x(X_x) + Y_x)$, 2) update $z^R = \min(\varphi_x.f_x(X_x) - Y_x - u_x)$, and 3) add $l_x - \varphi_x.f_x(X_x) + Y_x \leq 0$ into the constraints of the RHS. Σ is consistent when $z^L \leq 0, z^R \leq 0$.

C. Duality and complexity

In this section, we discuss the dual nature and complexity of the two fundamental problems described of STCD.

Duality Analysis. Since there are some common features (*e.g.*, inference equation and feasible domain) between the complication and inconsistency determination of STCD, we note that the inference equation's structure of the consistency problem is just the dual form of the implication problem. Accordingly, we propose the properties of both fundamental problems of STCD below.

***Theorem** 1: For any inconsistent set Σ, choose any STCD $\varphi_x \in \Sigma$, the rest STCDs formed set $\Sigma' = \Sigma \backslash \varphi_x$, which implies $\varphi'_x s$ negation: $\neg\varphi_x$.*

Proof. We need to prove $(\Sigma').Fes \subseteq (\neg(\varphi_x)).Fes$. Since the tolerable range of $(\neg(\varphi_x))$ is the union of $(-\infty, l_x)$ and $(u_x, +\infty)$, we divide $(\neg(\varphi_x)).Fes$ by constructing two virtual STCDs for $\neg\varphi_x$: $\neg(\varphi_x)^L = (X \rightarrow^f_{(-\infty,l_x)} Y, f), \neg(\varphi_x)^R = (X \rightarrow^f_{(u_x,+\infty)} Y, f)$, let $\neg\varphi_x.Fes = (\neg\varphi_x)^R.Fes \cup (\neg\varphi_x)^L.Fes$, and the goal becomes to prove that $(\Sigma').Fes \subseteq ((\neg\varphi_x)^R.Fes \cup (\neg\varphi_x)^L.Fes)$.

As Σ is inconsistent, there are two cases according to the inference equation of consistency testing: Case 1. $z^L = \min(l_x - \varphi_x.f_x(X_x) + Y_x) \geq 0$, and Case 2. $z^R = \min(\varphi_x.f_x(X_x) - Y_x - u_x) \geq 0 \wedge z^L < 0$.

TABLE V
THE FAMILY OF DEPENDENCIES

Dependencies	Implication	Consistency
Linear-STCD	PTIME	PTIME
General-STCD	NP-hard	NP-hard
CFD	coNP-complete	NP-complete
CIND	O(1)	EXPTIME-complete
FD	O(1)	O(1)

For Case 1, $z^L \geq 0$, which is equal to $\max(\varphi_x.f(X) - Y - l_x) \leq 0$. Consider the implication inference equation of Σ' to $\neg(\varphi_x)^L$, it is always satisfied since under the condition that $\forall i = 1, 2, \ldots n, l_i \leq \varphi_i.f(X) - Y \leq u_i$, two premises are satisfied: $z1^L = \max(-\infty - \varphi_x.f(X) + Y) \leq 0$ and $z1^R = \max(\varphi_x.f(X) - Y - l) \leq 0$. Since $z1^L \leq 0, z1^R \leq 0$, it means Σ' implies $\neg(\varphi_x)^L$, then $\Sigma'.Fes \subseteq (\neg(\varphi_x)^L).Fes$, thus $\Sigma'.Fes \subseteq (\neg(\varphi_x)).Fes$ holds.

For Case 2, $z^R \geq 0 \wedge z^L < 0$. Similarly, $\Sigma' \cup (X \rightarrow^f_{(l_x,+\infty)} Y, f)$ implies the right negation of φ_x: $\neg(\varphi_x)^R = (X \rightarrow^f_{(u_x,+\infty)} Y, f)$, which means $(\Sigma'.Fes \cap (X \rightarrow^f_{(l_x,+\infty)} Y, f).Fes) \subseteq \neg(\varphi_x)^R.Fes$. We aim to proof that $(\Sigma').Fes \subseteq ((\neg\varphi_x)^R.Fes \cup (\neg\varphi_x)^L.Fes)$ and $\forall d \in \Sigma'.Fes$, and there are two possibilities: (a). Data point $d \in \Sigma'.Fes \cap (X \rightarrow^f_{(l_x,+\infty)} Y, f).Fes$, and (b). $d \in \Sigma'.Fes \backslash (X \rightarrow^f_{(l_x,+\infty)} Y, f).Fes$. For (a), since $(\Sigma'.Fes \cap (X \rightarrow^f_{(l_x,+\infty)} Y, f).Fes) \subseteq \neg(\varphi_x)^R.Fes$, $d \in \neg(\varphi_x)^R.Fes$. For (b), since $d \in \Sigma'.Fes - (X \rightarrow^f_{(l_x,+\infty)} Y, f).Fes$, $d \in \Sigma'.Fes \cap (X \rightarrow^f_{(-\infty,l_x)} Y, f).Fes$, then $d \in \neg(\varphi_x)^L.Fes$. To sum up, $\forall d \in \Sigma'.Fes, d \in ((\neg\varphi_x)^R.Fes \cup (\neg\varphi_x)^L.Fes)$, it has $(\Sigma').Fes \subseteq (\neg(\varphi_x)).Fes$.

From the above, we deduce that in either case $\Sigma'.Fes \subseteq (\neg(\varphi_x)).Fes$ is true, and therefore, Proposition 1 holds. ■

Accordingly, the duality is not only in the inference equation's form, but also reflected in the semantic conflict of any inconsistent STCD set. We note that Theorem 1 can be applied as a materialization of the duality conclusion [15]. That is, an inconsistent STCD set will generate *unsatisfying* dependencies.

Complexity Analysis. We propose the complexity of implication and consistency determination in Theorem 2. Since the complexity results depends on the type of the objective function in STCD, we conclude the complexity analysis in Figure 3. We stress that it costs polynomial time for the implication and consistency analysis of STCDs with linear functions or some particular class of functions. Combined the theoretical foundations of STCD and the existing dependencies, we summarize the complexity comparison in Table V.

***Theorem** 2: Both the implication and consistency determination problems of STCD with general functions is NP-hard, but there exists algorithm of PTIME for the STCD composed of some particular kind of functions as listed in Figure 3.*

Proof. According to Section III-B, we can reduce the implication and satisfiability problem into a unified programming

Objective Functions	Set's Function	Programming Problem	Complexity
Linear	Linear	Linear Programming(LP)	PTIME
Fraction	Linear	Linear Fractional Programming(LFP)	PTIME
Positive Definite Quadratic	Linear	Convex Quadratic Programming(QP)	PTIME
Convex	Convex	Convex Programming	NP-Hard
NonPositive Definite Quadratic	Linear	Quadratic Programming(QP)	NP-Hard
Quadratic	Quadratic	QCQP	NP-Hard

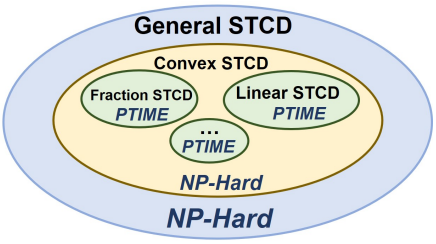


Fig. 3. Complexity Analysis

problem in PTIME (Lines 1-6 in Algorithm 1).

$$\max \quad z = f_x, \tag{1}$$
$$\text{s.t.} \quad f_i \leq 0 \quad (\forall i = 1, 2, \dots n) \tag{2}$$

According to the classification of related problems in constraint optimization theory, the above unified programming problem belongs to Constraint Satisfaction Problem (CSP) whose complexity depends on the functions it involves *i.e.,* f_x and f_i. In general, this problem is NP-Hard, since many of its subproblems are NP-Hard (e.g., general QCQP [16]). Luckily, some Convex CSP are PTIME solvable, such as linear programming [17], linear fractional programming [18], and convex quadratic programming [19]. Therefore, the proposed two fundamental problems are NP-Hard, as they are PTIME reducible to general CSP. Fortunately, with the assumption that they only contain some certain functions (e.g., linear), they can be PTIME reducible to easier problems (e.g., linear programming), which proves that it exists PTIME algorithm for paticular STCD's implication and consistency testing. ■

From the above, the theoretical solving complexity of implication and consistency testing depend on the nature of the internal function f of STCDs. If the internal function mapping is linear or conforms to some functions in Figure 3, then the two algorithms will be PTIME. If not, the algorithm will not be able to guarantee the solution in the deterministic time.

IV. Discovering STCD

We propose the STCD discovery problem in Section IV-A, introduce the discovery solution STCDiscover with three main phases in Section IV-B, and discuss the feasible bound with probabilistic guarantees in Section IV-C.

A. STCD Discovery Problem

Minimal and Vaild set of STCD Σ. Σ is *minimal* if there does not exist any φ in Σ which is implied by the set composed of the other STCDs in Σ, *i.e.,* $\nexists\varphi$ in Σ, φ is implied by $\Sigma\backslash\varphi$. Σ is *valid* if it satisfies consistency, *i.e.,* $\nexists\varphi_x \in \Sigma, \neg\varphi_x$ is implied by $\Sigma\backslash\varphi_x$. Accordingly, we formalize the STCD discovery problem in Definition 5.

***Definition** 5:* A STCD φ holds approximately with $1-\delta$ probability, if $\forall t_i \in \mathcal{R}$, the probability confidence that the φ holds is not less than $1-\delta$, *i.e.*, $\Pr(l \leq f(X) - Y \leq u) \geq 1-\delta$. The **STCD discovery problem** is to find a valid minimal set Σ of STCDs that holds on $\mathcal{R}$'s instance I, where any φ, which is $1-\delta$ holds approximately on $\mathcal{R}$, is implied by Σ. ■

Below we discuss the similarities and differences between STCD discovery problem and the existing dependency discovery problem. For the similarity, both problems are expected to achieve such search objectives [20], including *i*) *Coverage*. Any STCD φ holding on I could be implied by Σ, *ii*) *Minimality*. Removing any STCD φ from Σ would make φ cannot be implied by $\Sigma\backslash\varphi$, and *iii*) *Validation*. Σ is consistent and there is not violation of Σ on I. The difference is that STCD discovery problem aims to find out φs that *approximately* hold in probability, due to the characteristics of numerical data.

We stress that the proposed STCD discovery problem is challenged to solve for the following aspects: 1) exponential search spaces of the potential function structure f in STCD, and 2) the appropriate value domain *w.r.t* $f.(l, u)$.

For the concern of the search space, we analysis the size of the search space from the perspective of tree structures. We construct a functional expression tree for each φ, whose leaf nodes denote the independent variables X, and the inner nodes denote the operators, as shown in Figure 4. The mapping process of $f : X \rightarrow Y$ is implemented with a bottom-up approach. We briefly introduce the construction of the tree: We take the constant mapping $f^1(X) = X$ with tree depth 1 as a leaf node, and recursively define the functional expression tree f^k with the maximum depth k. For the operator Δ, real-valued constant C, the expression tree $f^k = f^{k-1}\Delta v (k \geq 1)$, where $v \in \{f^m, C\}, (m < k)$.

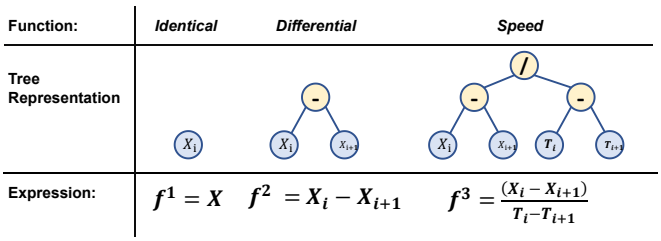


Fig. 4. Examples of functional expression trees

Given the M-dimensional data I with window size w, a functional expression tree of k with only binary operators can be considered as a binary tree of depth $k+1$, where the elements in w constitute the leaf nodes and B kinds of operators constitute the non-leaf nodes. Let $n(k)$ denote the size of the search space when the maximum tree depth is k . When $k = 1$, each element may be used as an independent variable, *i.e.,* $n(1) = M \cdot w$, and when $k = k_0 + 1, (k_0 \geq 0)$, then the k-order functional expression tree is split into left and right subtrees,

and it has $n(k_0+1) = \mathsf{B} \cdot n(k_0)^2$. We finally obtain the total amount of possible functions $n(k) = \mathsf{B}^{2^k-1} \cdot (M \cdot w)^{2^k}$.

$n(k)$ obviously presents an exponential expression space! Specifically, given $w = 10$ and $M = 6$ with four kinds of operators $(+, -, \times, \div)$, all the possible expressions at tree depth 1 are $M \cdot w = 6 \cdot 10 = 60$. When the depth of tree reaches 5, the number of search spaces reaches $n(5) = 4^{2^4-1} \cdot (10 \cdot 6)^{2^4} = 3.02 \cdot 10^{37}$! Obviously, it is quite challenged to perform function expression search and dependency construction in such a large solution space.

When it comes to the determination of $f.(l, u)$ in one STCD, it is not easy to compute the appropriate range of the function values. The size of the interval length (l, u) reflects the generalization ability of STCD discovery methods. If $f.(l, u)$ of one φ is set too loosely, it is likely that all instances satisfy φ, while a strict domain of $f.(l, u)$ would cause part of clean data to be falsely identified as violations *w.r.t* φ. When the instance-driven discovery approaches are applied, the range $f.(l, u)$ obtained by the support metric on the training set may have lower generalization on other datasets. Thus, it is usually not practical to obtain an accurate (l, u) from small-scale training data. Fortunately, it is feasible to obtain (l, u) which is approximately accurate probabilistically when satisfying a given confidence threshold.

Faced with these challenges, we provide our STCD discovery algorithms in the next section.

B. STCD discovery algorithm

Considering the large search space of potential function expression, we design an efficient discovery method following two basic principles: 1) *Data-driven*. The discovered patterns should be compatible with the domain characteristics of the data. To take the power data scenario as an example, the expression $R = U/I$ is more suitable than a velocity pattern $V = \frac{x_i - x_{i+1}}{t_i - t_{i+1}}$, and 2) *Conciseness*. The discovered expression are expected to be structurally conciseness. For example, the expression $f(X) = A_1 \cdot A_2$ is better than $f(X) = A_1 \cdot \frac{1}{2-A_2^5}$, while they both shows that Y has positive correlation with either A_1 or A_2.

From the above, we adopt a data-driven supervised learning method to search the expression space, and design pruning strategies to achieve conciseness. Figure 5 outlines the whole discovery procedure, which consists of three steps.

Step1: Functional structure discovery. Given data I with window size w, we first discover the structure of function f by iteratively selecting one position in w as the dependent variable Y, and the other positions as independent variables X. According to the complexity analysis in Section III, we design different strategies for discovering *linear* functions and *general* functions. For linear function discovery, we adopt the Successive Halving strategy [21], [22] and execute descending sampling to select the terms from window w to the linear regression process, and then compute the regression coefficients on the selected terms. Linear regression returns the average loss and expression's length as the information guiding the halving strategy. For general function discovery, the genetic algorithm-based strategy is applied to search the space on the functional expression tree. We insert the depth of the tree into the *fitness* function, and apply a length-pruning strategy to guarantee the conciseness of the expression in STCDs.

Algorithm 2: STCDISCOVER

Input: Data instance I with window size w, confidence $1-\delta$, parsimony coefficient λ, and sample ratio τ
Output: STCD set Σ according to Definition 5

1 $\Sigma \leftarrow \emptyset$, $D_{train} \leftarrow$ SAMPLE(I, τ);
2 **foreach** *i from* 0 *to* w **do**
3 **foreach** *j from* 1 *to* $|Attr(I)|$ **do**
4 $Y_0 \leftarrow < i, j >$;
5 $f_0, Loss \leftarrow$ FUNCDISCOVER(D_{train}, λ) with Algorithm 3 or Algorithm 4 ;
6 $l, u \leftarrow$ BOUNDSET$(\delta, Loss)$;
7 $\varphi_x \leftarrow (X \rightarrow_{(l,u)} Y_0, f_0)$;
8 **if** IMPLICATIONTEST$(\Sigma, \varphi_x) =$ **False** and CONSISTENCYTEST$(\Sigma \cup \varphi_x) =$ **True then**
9 $\Sigma \leftarrow \Sigma \cup \varphi_x$;
10 **return** Σ

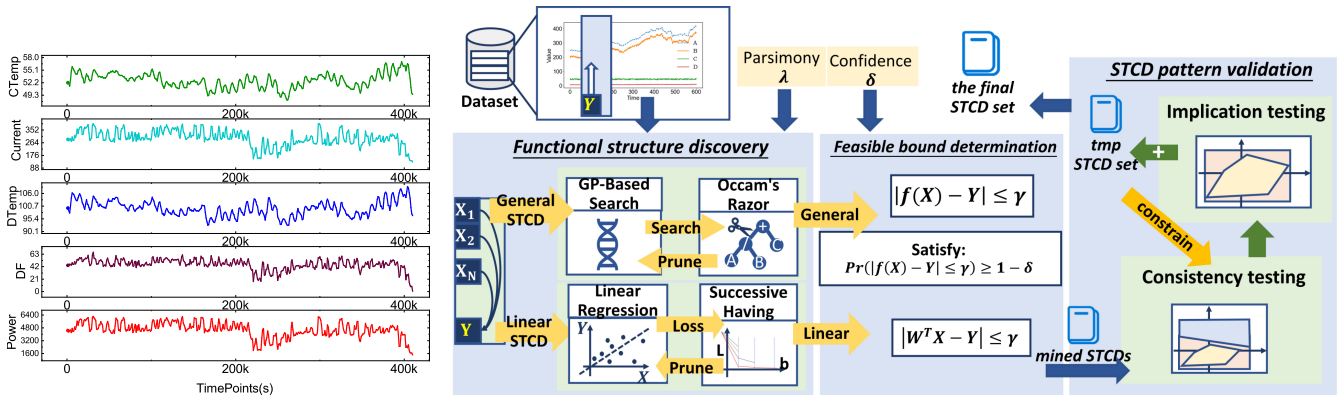
Step2: Feasible bound determination. After the functional patterns have been discovered, we adopt the confidence bounds analysis method based on PAC theory to compute the approximately correct bounds (l, u) *w.r.t* $\varphi.f$ in probability based on a $1 - \delta$ confidence. Specifically, considering that linear and nonlinear structures have different pseudo-dimensions definition, different bounds are set for linear and nonlinear structures.

Step3: STCD pattern validation. We validate the implication and consistency of the discovered φ according to the discussion in Section III. We determine to include the current STCD φ to the existing set Σ if $\Sigma \cup \{\varphi\}$ is consistent, and at the same time φ is not implied by Σ.

We iteratively perform the above steps until all possible positions of Y in w have been visited. We provide the pseudocode of STCD discovery process in Algorithm 2. In the double-layer loop lines 2-8, we iteratively choose the data to be Y from one position in w, with the rest data in w to be the independent variable X. We execute algorithm FUNCDISCOVER to obtain the function $f : X \rightarrow Y$, which will be discussed below. After determining f, we generate upper and lower bounds that satisfy $1-\delta$ confidence level with the BOUNDSET function (see Section IV-C). We then perform both implication and consistency testing for the generated φ_x, and add φ_x into the current Σ after the validation.

Since the linear function has unique properties whose implication and consistency issues could be determined in polynomial time, we design algorithm DISC-LF specifically for linear function discovery, and propose the general algorithm DISC-GF, as discussed below.

1) Linear STCD function mining: Algorithm 3 presents the discovering process of linear STCD functions. It first randomly generates multiple sets of attributes from time window as candidate independent variables $PossibleX$, and then it applies the successive halving strategy to select the best candidate X



(a) Sample data

(b) STCDISCOVER procedure

Fig. 5. STCD discovery framework overview

Algorithm 3: DISC-LF

Input: Y, Data instance I with window size w, parsimony coeficient λ
Output: possible linear function$f : X \to Y$.

1 $PossibleX \leftarrow$ RANDOMGENERATE(w), $n \leftarrow |PossibleX|$, $funcList \leftarrow []$;
2 **for** $k = 0, 1, 2, \ldots, \lceil \log_2(n) \rceil - 1$ **do**
3 $Loss \leftarrow \{\}$, $funcTmp \leftarrow \{\}$;
4 **for** X_0 *in* $PossibleX$ **do**
5 $budget \leftarrow \lfloor \frac{\#(I)}{n \cdot \lceil \log_2(n) \rceil} \rfloor$;
6 $Loss[X_0], funcTmp[X_0] \leftarrow$ LINEARREGRESSION$(X_0, Y_0, budget)$;
7 $Loss[X_0] + \leftarrow (|X_0| \cdot \lambda)$;
8 $PossibleX, funcList \leftarrow$ SELECTTOPHALF$(PossibleX, Loss, funTmp)$;
9 **return** $funcList$

Algorithm 4: DISC-GF

Input: Y, Data instance I with window size w, parsimony coeficient λ, Maximum depth K, generations r
Output: General STCD functions

1 $Population \leftarrow$ INITIALIZEPOP(K, w)// randomly generate expression tree population;
2 **foreach** *i in range(r)* **do**
3 **foreach** *inividual in Population* **do**
4 GETFITNESS$(individual, Y, I, \lambda)$;
5 $Winner \leftarrow$ SELECTWINNER$(Population)$;
6 $Population \leftarrow$ CROSSMUTATE$(Winner)$//crossOver;
7 $Population' \leftarrow$ PRUNE$(Population, K)$;
8 **return** $Population'$;

with minimum loss by gradually halving of possible search Spaces in $PossibleX$ and increasing the budget (lines 2-8). We apply the Bandit strategy [21], [23] in this task, which are considered to be effective in practices.

Note that the ridge regression LASSO [24] is a similar approach with the propose STCDISCOVER, which designs the regular coefficients as L1 penalty terms. However, the ridge regression is not applied in our method, for the reason that the distribution of the dependencies represented by STCD is sparse in many cases, while LASSO tends to perform poorly under sparse prior knowledge according to Ref. [25]. Our algorithm involves the length of STCD into the loss function and achieves conciseness by the successive halving pruning method.

2) General STCD function mining: As symbolic regression is a supervised-learning strategy which can be applied in the expression space search problems, we treat the discovery of general STCD functions as a special case of symbolic regression problem. Since genetic algorithms are demonstrated to have better robustness in the field of symbolic regression [26], we use a Genetic algorithm-based symbolic regression method for heuristic search of expressions. The method first randomly initializes a number of functional expression trees as a population, where each expression tree is treated as an individual, and each expression tree has its own fitness accordingly to the historical data. We iteratively calculate the fitness of expressions on the sample and select expressions with higher fitness for crossover variation (swapping subtrees (crossover), changing the sign of internal operation nodes (variation)). The search process stops after certain times of iterations. During the process, we measure the fitness of the individual expression(Ind) with Equation (1).

$$fit(Ind) = -\left(\frac{1}{m} \cdot \sum_{i=1}^{m} (D_i[Y] - Ind(X))^2 + \lambda \cdot length(Ind)\right) \tag{1}$$

Here, D is the training data, and Y is the dependent variable attribute. Using the mean square error (MSE) in the fitness function, the shape of the expression tree eventually became more suitable to the original data, and we add the parsimony coefficient (λ) to penalize overly complex formulas for expression trees of different lengths .

We outline the discovery process in Algorithm 4. It initializes to randomly generate a population of expression trees with maximum depth k, and then iteratively executes the genetic algorithm to discover possible expression population by computing the fitness of each individual. It performs superiority and cross mutation in lines 5-6, and computes the candidate population. Similar with Algorithm 3, we obtain the

expression results after a pruning function on the population.

C. Theoretical analysis of the feasible bound

After we obtain an inner functional mapping of a discovered STCD, we need further pay attention to the feasible bound of the function, *i.e.,* $f.(l,u)$ to guarantee the correctness of the data in relation $\mathcal{R}$. Note again that it is always unrealistic to obtain an exact bound $f.(l,u)$ of φ from the limited-scale training data. In this case, we aim to determine an approximately exact solution $f.(l,u)$ with probabilistic guarantees, and provide our results in Theorem 3.

***Theorem** 3: For the general function f, one could obtain STCD $\varphi : (X \rightarrow^{f}_{(-\gamma,\gamma)} Y, f(X))$ by the interval estimation with $1-\delta$ confidence, where the feasible bound γ is calculated by Equation (2).*

$$\gamma = \widehat{R(D)} + b\sqrt{\frac{2\cdot(d)\cdot\log\frac{e\cdot m}{d}}{m}} + b\sqrt{\frac{\log\frac{1}{\delta}}{2m}} \tag{2}$$

where $\widehat{R(D)}$ denotes the empirical loss over the sample training data D, m denotes the size of D. b is the upper bound of the loss function, e is Euler number, and d is the Pseudo-dimension.

Proof. Since the generalization error of the regression model is bounded in PAC theory [27], we take the STCD function as a learned regression hypothesis. With an important inequality of bounded infinite regression hypothesis [27], *i.e.*, for any $\delta > 0$, with probability at least $1-\delta$ over the choice of a sample of size m, the following inequality holds for all hypothesis h: $R(h) \leq \widehat{R(D)} + b\sqrt{\frac{2\cdot(d)\cdot\log\frac{e\cdot m}{d}}{m}} + b\sqrt{\frac{\log\frac{1}{\delta}}{2m}}$. Here, the left hand side is the generalization error, *i.e.*, $|Y - f(X)|$ in the proposed cases. And the right hand side $\widehat{R(D)}$ is calculated according to the loss of function in training data loss on average.

With absolute losses, the inequality can be written as follows: $|Y - f(X)| \leq \widehat{R(D)} + b\sqrt{\frac{2\cdot(d)\cdot\log\frac{em}{d}}{m}} + b\sqrt{\frac{\log\frac{1}{\delta}}{2m}}$. Therefore, by setting γ to be the right hand side, we obtain $\Pr(|f(X) - Y| \geq \gamma) \leq \delta$, which is equal to: $\Pr(-\gamma \leq f(X) - Y \leq \gamma) \geq 1 - \delta$. That is, for general function f, through this particular interval estimation, one can obtain the STCD $(X \rightarrow^{f}_{(-\gamma,\gamma)} Y, f(X))$ with $1-\delta$ confidence, where $\gamma = \hat{R(D)} + b\sqrt{\frac{2\cdot(d)\cdot\log\frac{em}{d}}{m}} + b\sqrt{\frac{\log\frac{1}{\delta}}{2m}}$. ■

According to Theorem 3 and the pseudo-dimension's definition in [27], we can derive two corollaries that help us to actually determine $f.(l,u)$ for one STCD.

Corollary 1. For a linear functional mapping: $f(X) = W^T X + v$, by the above interval estimation, the following STCD $(X \rightarrow^{f}_{(-\gamma,\gamma)} Y, f(X))$ can be obtained with a confidence of $1-\delta$, where : $\gamma = \widehat{R(D)} + b\sqrt{\frac{2\cdot(|X+1|)\cdot\log\frac{em}{|X+1|}}{m}} + b\sqrt{\frac{\log\frac{1}{\delta}}{2m}}$.

Corollary 2. For a general functional mapping: $f(X) = \sum_1^T f_i(X)$, by the above interval estimation, the following STCD $(X \rightarrow^{f}_{(-\gamma,\gamma)} Y, f(X))$ can be obtained with a confidence of $1-\delta$,where : $\gamma = \widehat{R(D)} + b\sqrt{\frac{2\cdot(|T|)\cdot\log\frac{em}{|T|}}{m}} + b\sqrt{\frac{\log\frac{1}{\delta}}{2m}}$.

Proof. Corollary 1 is obvious to be true according to the definition of pseudo dimension in [27]. We briefly explain the theoretical derivation of Corollary 2: Since $\frac{\partial\gamma}{\partial d}$ depends mainly on $\frac{\partial(\frac{2\cdot(d)\cdot\log\frac{em}{d}}{m})}{\partial d} = \log e\cdot m/2 - 1$, when the training data is large, the derivative is greater than 0. The reaction is monotonically increasing on pseudo dimension. Greater pseudo dimension estimation results in greater $\gamma' > \gamma$, which derives $\Pr(|Y - f(X)| \geq \gamma') \leq \Pr(|Y - f(X)| \geq \gamma) \leq \delta$. Here, we use T as our estimation. Since the terms number of the expression is larger than the original pseudo dimension, d with T provides a greater γ', and it let $(X \rightarrow^{f}_{(-\gamma',\gamma')} Y, f(X))$ satisfied.

V. EXPERIMENTAL STUDY

In this section, we report the experimental study of the proposed methods. All experiments run on a computer with 16GB RAM and AMD Ryzen7 5800H CPU.

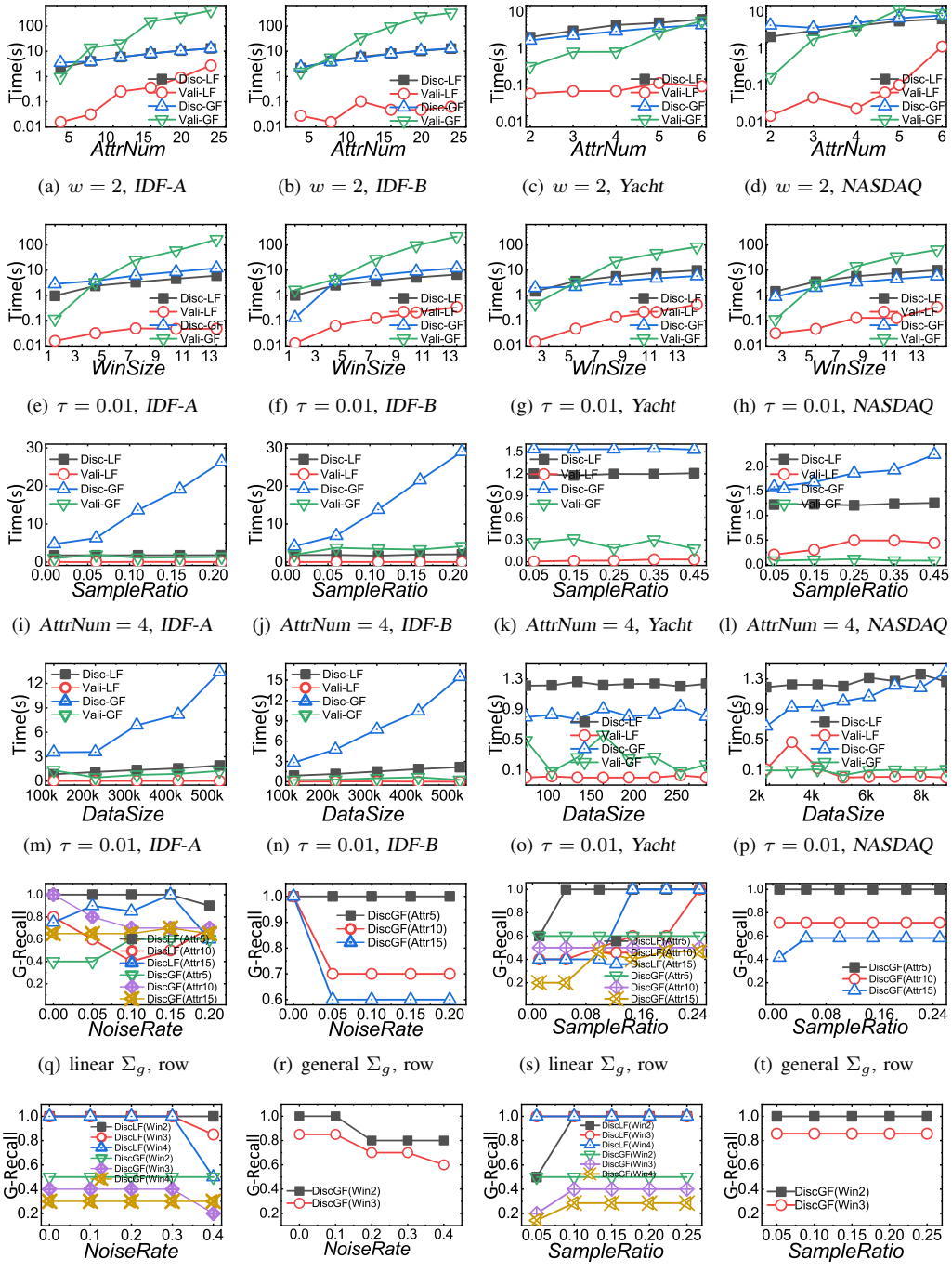
A. Experimental settings

Datasets. We employ three real-life datasets and one synthetic dataset in the experiments. (1) *NASDAQ* is extracted from historical NASDAQ index, which is well-known in numerical data cleaning literatures [12], [30]. Each record is arranged into fields representing Date, Opening Price, High, Low, Close, AdjClose, and Volume of the day. (2) *Yacht* Hydrodynamics dataset comes from UCI dataset for benchmarks studies *w.r.t* time series data, with attributes including beam-draught Ratio (BDR), froude number (Froude), and residuary resistance per unit weight (R_r). All the attributes are dimensionless, and *Yacht* is only used for STCD discovery evaluation on rows, for it has no time-related information. (3) *IDF* has 80 sensors recording the working conditions of an induced draft fan-machine group from a fossil-fuel power plant. We have analyzed data on more than 1M time points for 5 consecutive months with 40 attributes, including Power (Pow), temperature sensors DTemp, CTemp1, CTemp2, force sensors DF, air pressure sensors Pressure, etc.

Criteria. We evaluate the proposed algorithms from three perspectives: 1) Running time are used in efficiency evaluation, 2) *G-Recall* applied in [9] measures the effectiveness in discovering golden STCDs. We conclude the golden STCDs for each dataset by Σ_g, which have been designed by domain experts. *G-Recall* = $\frac{\#(\Sigma_g \text{implied} \Sigma_s)}{\#(\Sigma_g)}$ denotes the number of STCDs in discovered Σ_s that are implied by the golden rule set Σ_g over the total number of golden STCDs, and 3) *Accuracy* and *F1*-measure metrics are used to evaluate the error data detection effectiveness with the discovered STCDs. $Acc = \frac{TP+TN}{TP+TN+FP+FN}$ and $F1 = \frac{2\cdot Prec\cdot Rec}{Prec+Rec}$.

B. Efficiency Evaluation

In this section, we report the efficiency of algorithms under four critical parameters, *i.e.,* the number of attributes (*AttrNum*), the window size w, the total data amount, and the sampling rate τ in Algorithm STCDISCOVER which limits the search space of dependent variables. To make it clear, we introduce the time cost of the linear STCD function discovery



(a) $w = 2$, *IDF-A* (b) $w = 2$, *IDF-B* (c) $w = 2$, *Yacht* (d) $w = 2$, *NASDAQ*

(e) $\tau = 0.01$, *IDF-A* (f) $\tau = 0.01$, *IDF-B* (g) $\tau = 0.01$, *Yacht* (h) $\tau = 0.01$, *NASDAQ*

(i) *AttrNum* $= 4$, *IDF-A* (j) *AttrNum* $= 4$, *IDF-B* (k) *AttrNum* $= 4$, *Yacht* (l) *AttrNum* $= 4$, *NASDAQ*

(m) $\tau = 0.01$, *IDF-A* (n) $\tau = 0.01$, *IDF-B* (o) $\tau = 0.01$, *Yacht* (p) $\tau = 0.01$, *NASDAQ*

(q) linear Σ_g, row (r) general Σ_g, row (s) linear Σ_g, row (t) general Σ_g, row

(u) linear Σ_g, column (v) general Σ_g, column (w) linear Σ_g, column (x) general Σ_g, column

Fig. 6. Comparison experimental results with varying parameter values

(e.g., Algorithm 3) and general STCD function discovery (e.g., Algorithm 4), respectively. And we report the time cost of two main phases, two phases, *i.e.,* STCD function discovery and STCD pattern validation.

1) EXP-1: Efficiency on varying w and AttrNum: Figure 6(a-h) shows the results on the condition that sampling ratio $\tau = 0.1$. Both DISC-LF and DISC-GF increase linearly with a growing w or *AttrNum*, while the STCD validation steps *i.e.,* VALI-LF and VALI-GF trend to grow a little rapidly in time costs. For the function discover phase, it verifies that the time cost increases linearly with the given window size or the attribute numbers, with all else equal. It shows less difference in the trend of time variation and total time cost of both DISC-LF and DISC-GF algorithms, because the total budget required by the two algorithms is similar.

For the STCD validation phase, the implication testing is inherently difficult as mentioned in Section III, so the validation time costs grow fast with either the increasing w

TABLE VI
CASE STUDY OF STCDS

	DataSet	Discovered STCDs	Explanation
1	*IDF*	$(t.\{\mathsf{DTemp}, \mathsf{CTemp1}, \mathsf{DF}\} \rightarrow^{f}_{(-128,128)} t.\mathsf{Power}, f = (\mathsf{DTemp} + \mathsf{CTemp1} \cdot \mathsf{DF}))$	Generator's power is determined by temperature T and force F, and higher temperature reflects greater power produced.
2	*IDF*	$(t.\mathsf{CTemp1} \rightarrow^{f}_{(-0.11,0.11)} t.\mathsf{Pressure}, f = \log(t.\mathsf{CTemp1}))$	Similar to Tentens formula [28] in thermodynamics, the exponential relationship between temperature and pressure is mined.
3	*IDF*	$(t.\{\mathsf{CTemp}, \mathsf{DF}\} \rightarrow^{f}_{(-6.9,6.9)} t.\mathsf{Current}, f = \mathsf{CTemp} \cdot \mathsf{DF})$	Greater temperature reflects greater power. As $I = P/U$ in electricity, $\mathsf{Current} \propto \mathsf{P} \propto (\mathsf{T} \cdot \mathsf{F})$ under constant voltage
4	*IDF*	$(t.\mathsf{CTemp1} \rightarrow^{f}_{(-0.02,0.02)} t.\mathsf{CTemp2}, f = t.\mathsf{CTemp1} - 0.317)$	The temperature of nearby sensors is linearly correlated.
5	*IDF*	$(t_i.\mathsf{Power}_i \rightarrow^{f}_{(-32.4,32.4)} t_{i+1}.\mathsf{Power}_{i+1}, f = t_i.\mathsf{Power} - 0.805)$	Sequential Dependency on Generator's Power.
6	*Yacht*	$(t.\mathsf{F}_r \rightarrow^{f}_{(-38.45,38.45)} t.\mathsf{R}_R, f = \frac{-0.47}{\mathsf{F}_r^2} + 19.53)$	According to Froude's law [29], $\mathsf{R}_R \propto \frac{1}{\mathsf{F}_r^{-2}}$.
7	*NASDAQ*	$(t_i.\mathsf{Open} \rightarrow^{f}_{(-63.28,63.28)} t_{i+1}.\mathsf{Open}, f = t_i.\mathsf{Open} - 0.08)$	Sequential Dependency on Stock's Opening prices, and the changes of opening prices are obtained in a certain interval
8	*NASDAQ*	$(t_i.\mathsf{Close} \rightarrow^{f}_{(-0.04,0.04)} t_{i+1}.\mathsf{AdjClose}, f = \mathsf{Close}_i + 0.04)$	The adjusted closing price is often close to the closing price

or *AttrNum*. In general, VALI-LF takes less time than VALI-GF, which mainly due to the fact that the implication test for linear functions could be directly solved by the package of linear programming optimization, while the implication test for nonlinear functions is more difficult and requires a larger time cost.

2) EXP-2: Efficiency on varying sample ratio and data amount: Figure 6(i-p) shows the results on the condition that $w = 2$. Both DISC-LF and DISC-GF increase with a growing τ or *DataSize*, while the STCD's validation step tends to remain stable within a certain range. For the function discovery phase, sampling ratio and data amount directly affect the input scale of DISC-LF and DISC-GF, which leads to the increase of the time cost. As the DISC-GF's genetic programming needs to traverse the training data for each individual in every iteration, it often brings more overhead than the DISC-LF.

For the STCD validation phase, with $w = 2$, *AttrNum* $= 4$, VALI-LF and VALI-GF tend to be within a certain overhead. Due to randomness exists in STCD discovery process (*i.e.,* the Genetic population), different STCD patterns are mined each time, which leads to a certain fluctuation in VALI-LF and VALI-GF's performance. We note that the time cost of STCD discovery increases with *DataSize*, which could be accelerated by setting appropriate sampling ratio τ.

C. Effectiveness Evaluation

1) EXP-3: Implication reduction and case study: Since the implication reduction is the important guarantee of the simplicity of the STCD set Σ, we report the total number of the discovered STCD before and after the reduction according to the implication test discussed in Section III-A. It shows that the proposed Algorithm 1 could effectively detect those STCDs that are implied in the existing Σ, which in turn improves the efficiency of the error date detection and repairing in the follow steps for data cleaning.

We present part of the discovered STCDs for each dataset in Table VIII. It shows that the proposed STCDISCOVER algorithm could effectively uncover hidden dependency patterns in the sequential data, and achieve high expressiveness for data dependencies. The discovered STCDs include not only dependencies between multiple attributes, but also the contextual dependencies in the same attribute. For dataset *IDF* and *NSADAQ*, STCDISCOVER has the ability to find out complex dependency expressions between multiple attributes, which assist to further understand the characteristics of the data.

TABLE VII
STCDS BEFORE AND AFTER IMPLICATION

DataSet	# STCDs Before	# STCDs After	% Reduction
NSADAQ	70	40	42%
Yacht	70	52	25%
IDF-A	370	318	14%
IDF-B	370	295	20%

Below, we will evaluate the quality of STCDISCOVER with sets of golden rules, referring to DCs validation in [9]. We evaluate the results of discovering dependency on rows and on columns on the synthetic dataset extended from *IDF* in Exp-4 and Exp-5, respectively, to clearly present the discovery ability of either spatial and temporal dependencies.

2) EXP-4: Evaluation of discovering dependency on rows: We report the discovery *G-Recall* of DISC-LF and DISC-GF with the golden STCD sets consisting of either linear function patterns only and general function patterns between attributes with $w = 1$. Figure 6(q,r) reports the discovery quality with varying error rate (*noi%*) in data. Since DISC-LF could not be applied in nonlinear function discovery, we only shows the result of DISC-GF with the general Σ_g in Figure 6(r), and similar cases exist in Figure 6(r, v, x). With only linear function in Σ_g, DISC-LF performs good when the total number of attributes is small, and the gap between DISC-LF and DISC-GF indicates the superiority of the specially-designed DISC-LF for the sequential datsets dominated by linear-functional relationships. With a general Σ_g, Figure 6(r) shows that *G-Recall* of DISC-GF decreases when error rate reaches 0.05 in the dataset with much more attributes, and then stays stable. It is because that the hypothesis space grows exponentially with *AttrNum*, while the distribution of the gold standard becomes more sparse, and in turn affects the discovery effectiveness.

Figure 6(s,t) reports the discovery quality with varying sample rate τ. Not surprising that *G-Recall* of algorithms increases significantly and then stabilizes with a growing τ. It shows that *G-Recall* would decrease obviously with a larger

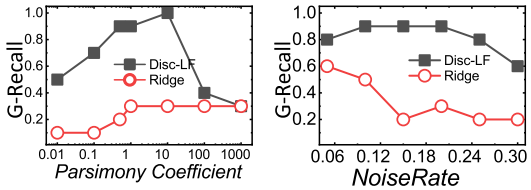


Fig. 7. DISC-LF vs Ridge Regression

AttrNum in both cases. However when only using linear functions in Σ_g, DISC-LF performs better than DISC-GF with the growth in *AttrNum*. This confirms the robustness of our designed DISC-LF to the growing number of attributes.

3) EXP-5: Evaluation of discovering dependency on column: Similarly, we design the golden rules on column according to SDs [11], and report the results *w.r.t* discovering quality on columns in Figure 6(u-x) with $AttrNum = 10$. It shows the excellent result of DISC-LF when the noise rate no larger than 0.3, and whatever τ value is. We note that the high *G-Recall* on the condition $w = 4$, and $noi\% = 0.3$ shows the potential for multi-dimensional sequential data applications. DISC-GF could perform a low-level steady score against $noi\%$, which shows that DISC-GF is not a good choice faced with the linear Σ_g. For the cases with general Σ_g, DISC-LF could maintain 0.7 in *G-Recall* with $w \leq 3$ and $noi\% = 0.3$. It shows that the window size has an obvious effect on the discovery Recall, which inspires us to explore the optimization of DISC-GF in our future work.

D. Comparison experiments

In this section, we evaluate the effectiveness of the proposed STCDISCOVER in comparison experiments.

1) EXP-7: DISC-LF *vs Ridge Regression:* Figure 7 shows the performance result of the proposed DISC-LF and ridge regression in linear function discovery tasks. We evaluate the performance with varying parsimony coefficient in Fig. 7(a) and the performance with varying noise rate in data in Fig. 7(b). It shows that DISC-LF with linear regression method performs better that the ridge regression method. It indicates that the simple ridge regression algorithm often fails to accurately calculate the coefficients in the function of STCD φ, which trends to assign a *uniform* coefficient to each element involved in the function, and results in a linear function of excessive length. The half-by-half pruning algorithm used in this paper could effectively avoid such cases, and it shows that DISC-LF achieves the best performance when parsimony coefficient equal to 10, and it maintains stable results when $noi\% \leq 0.2$, and has obvious superiority than the ridge regression algorithm.

2) EXP-8: STCD vs the existing constraints in error data detection: We measure the performance of STCD with two representative existing data quality constraints, *i.e.,* DC and SD for numerical data in error detection tasks. We consider the original clean datasets as ground truth, and inject dirty data into different time intervals, where we consider three classical errors in sequential data, *i.e.,* single error point, continuous errors, and contextual errors among multi-attributes [1], [31].

TABLE VIII
ERRORS DETECTION WITH DATA DEPENDENCIES

	Noise Rate	*NASDAQ*		*IDF-A*		*IDF-B*		*Yacht*	
		Acc	F1	Acc	F1	Acc	F1	Acc	F1
STCD	0.2	0.92	0.93	0.73	0.7	0.68	0.62	0.91	0.77
	0.4	0.92	0.97	0.67	0.76	0.61	0.71	0.84	0.74
DC	0.2	0.89	0.89	0.61	0.31	0.52	0.55	0.89	0.66
	0.4	0.9	0.94	0.42	0.37	0.59	0.7	0.78	0.62
SD	0.2	0.72	0.77	0.66	0.59	0.65	0.57	-	-
	0.4	0.73	0.81	0.58	0.67	0.56	0.65	-	-

The experimental results show that the STCD-based and DC-based error data detection methods have similar performance on the four datasets. Both methods perform better on *NSADAQ* and *Yacht*, while their scores are slightly lower on *IDF*. This indicates that both STCD and DC have high detection effectiveness in the cases that the dependencies in the data are not quite complex. For SD, it is somewhat surprisingly that detection effectiveness with column-only constraints (*i.e.*, SD) is better than the ones with row-only constraints (*i.e.*, DC). It confirms again the necessity to consider temporal context dependencies. The proposed STCD not only expresses more complex constraints on rows for sequential data, but also be compatible with SDs defined on columns. Thus, the STCD-based detection method could both effectively identify the single point errors and collective errors in time series data.

VI. RELATED WORK

Traditional data dependencies. Researchers have conducted a lot of theoretical and applied research on relational data quality rule expression and violation detection in data cleaning community. Data dependencies discovery problem has been proposed to find valuable patterns from data, which aims to obtain the minimum set of all correct and non-trivial rules in data. Different discovery techniques including schema-driven approaches(e.g., TANE, CTANE), instance-driven approaches (e.g., FASTFD, FASTCFD) (see [20] as a survey). In 2013, Denial constraints (DC) have been proposed to improve the expressiveness for relational data quality, and achieve "greater-than" and "less-than" comparison between attribute values. DC discovery algorithms such as FASTDC [9] and Hydra [32] compute the predicate space for data, and filter the evidence set for computing DC pattern by confidence and coverage metrics. Since it is always costly to discover exact data dependencies, approximate (or relaxed) dependencies discovery problem is studied to allow relaxation in constraint satisfaction. Recent works including the approximate discovery of FDs (e.g., AID-FD [33], RFDs [34]), CFDs, and DCs (e.g., ADCMiner [35]).

Dependencies for sequential data. Data dependencies issues in sequential data have drawn more attention recently, especially in IoT data quality applications. Golab et al. proposed the concept of sequential dependence [39] to describe the requirement of numerical differences between consecutive data points. Considering the quality requirements of data in terms of temporal attributes, Ref. [12] proposed four types of normalized data quality rules and provided data cleaning framework for temporal numerical data. The existing research mainly focus on the two types of single-entity multi-attribute

and multi-entity single-attribute, and there is still insufficient research on multi-entity multi-attribute rules.

Ref. [11] extended the idea of constraints from dependencies defined on relational database, and proposes sequential dependencies (SD) to describe the semantics of numerical data. Surveys [1], [2] and tutorials [3] are presented to discuss the repairing method for time series data. Recently, Song et al. proposed a repair method combining speed constraints and acceleration constraints [36] for discovering single-point error data with different magnitudes of change on a single time series. Due to the limitation in semantic expressiveness, fewer repair methods based on multi-entity and multi-attribute constraints are carried out at present.

Symbolic Regression. As a sub-field of machine learning, human-understandable mathematical expressions could be learned by Symbolic Regression (SR) procedure [37], [38], and SR problem recently, has been proved to be NP-hard. Genetic algorithm [37]–[39] and deep learning models [40]–[42] are widely applied in SR tasks to search for an appropriate expression structure. Since there lacks unified evaluation criterion, La Cava [26] proposed a comprehensive benchmark composed of 20 SR algorithms and 250 datasets. It shows that the performance of GP-based symbolic regression algorithm is better than that of neural network method in real-life datasets, while both the performance are similar in synthetic data.

VII. Conclusions

In this paper, we proposed a novel data dependency notation, spatio-temporal context dependency (STCD), and extended the expressiveness of data quality constraints from dependence on both attributes and instances for time series data. We discussed the implication and consistency problems for STCD, and developed algorithm STCDiscover to discover STCD from time series data. Experimental results on real-life and synthetic data present the effectiveness and efficiency of STCDiscover. The proposed method could uncover the inherent dependencies in both spatio-temporal context for time series, which are challenged to be captured by the existing data quality constraints. Future work includes approximate STCD discovery and efficient cleaning method of STCD violations in time series data.

Appendix

A. Proof of Proposition 2

Proof. We first prove the correctness of the inference system and then prove its incompleteness by giving a counter example.

(Correctness): For **Reflexivity**, if $\exists\varphi_i \in \Sigma$, having $\varphi_i.f = \varphi_x.f$ and $l_x \leq l_i \leq u_i \leq u_x$, then $\varphi_i.Fes \subseteq \varphi_x.Fes$. Since $\varphi_i \in \Sigma$, then $\Sigma.Fes \subseteq \varphi_i.Fes$. Hence, combining Proposition 1, we can deduce that Σ implies φ_x. For **Transitivity**, under the assumption that Σ implies φ_x and the set $\{\varphi_x\}$ implies φ_y, We can deduce that $\varphi_y.Fes \subseteq \varphi_x.Fes \subseteq \Sigma.Fes$. From the transitivity of the set, we can derive $\varphi_y.Fes \subseteq \Sigma.Fes$. Hence, it can be deduced that Σ implies φ_y. For **Augmentation**, since it does not affect the built-in function structure $\varphi_i.f$ and threshold $\varphi_i.(l_i, u_i)$, for any point d in $\varphi_i.Fes'$, d is also in $\varphi_x.Fes$. It means $\varphi_i.Fes' \subseteq \varphi_x.Fes$. Since $\varphi_i \in \Sigma$, then $\Sigma.Fes \subseteq \varphi_i.Fes$, making $\Sigma.Fes \subseteq \varphi_x.Fes$ which means φ_x is implied by Σ.

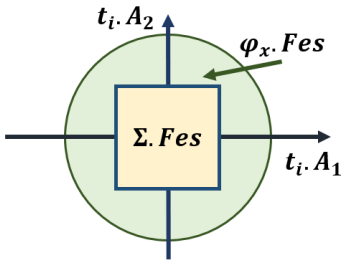


Fig. 8. Demonstration of counterexample

(Incompleteness): Consider the following example: $\Sigma = \{(t_i.A_1 \rightarrow^{f_1}_{(-0.5,0.5)} t_i.A_1, f_1 = 2 \cdot A_1), (t_i.A_2 \rightarrow^{f_2}_{(-0.5,0.5)} t_i.A_2, f_2 = 2 \cdot A_2)\}$, and $\varphi_x = (t_i.A_1, t_i.A_2 \rightarrow^{f_3}_{(0,1)} t_i.A_2, f_3 = (t_i.A_1)^2 + (t_i.A_2)^2 + t_i.A_2)$. φ_x is implied by Σ since in the space spanned by $t_i.A_1, t_i.A_2$, $\Sigma.Fes$ is constrained in a square of side length 1 centered at the origin point $(t_i.A_1 = 0, t_i.A_2 = 0)$, while $\varphi_x.Fes$ is constrained in a circle of radius 1 centered at the origin point. From Figure 8, we can find that the big circle contains the small square, so $\Sigma.Fes \subseteq \varphi_x.Fes$, and it has Σ implies φ_x. However, no matter how hard we try to reason with the above axiomatic system, we still can't get Σ implies φ_x. This counter example proves that inference based on these axioms is incomplete. ■

From the above proof process, we get that the symbolic inference system can not go deep into the function's structure, which leads to the incompleteness of the inference.

B. Proof of Proposition 3

Proof. We first prove the correctness of the inference system and then prove its completeness.

(Correctness): The goal of the correctness issue is to show that under $\forall\varphi_i \in \Sigma, l_x \leq \varphi_i.f_i(X_i) - Y_i \leq u_i$, if $z^L \leq 0$ and $z^R \leq 0$ then φ_x is implied by Σ. As $z^L \leq 0$ and $z^R \leq 0$ is equal to $\max(\varphi_x.f_x(X_x) - Y_x - u_x) \leq 0$ and $\max(\varphi_x.f_x(X_x) - Y_x - u_x) \leq 0$, we can deduce that $l_x \leq \varphi_x.f_x(X_x) - Y_x \leq u_x$ is always satisfied under the preconditions that $\forall\varphi_i \in \Sigma, l_x \leq \varphi_i.f_i(X_i) - Y_i \leq u_i$. The result tells that $\Sigma.Fes \subseteq \varphi_x.Fes$, as pick any point in $\Sigma.Fes$, it shall satisfy $l_x \leq \varphi_x.f_x(X_x) - Y_x \leq u_x$. According to Proposition 1, $\Sigma.Fes \subseteq \varphi_x.Fes$ shall deduce that φ_x is implied by Σ.

(Completeness): The goal of the completeness issue is to show that any φ_x implied by Σ, $z^L \leq 0$ and $z^R \leq 0$ is always satisfied under $\forall\varphi_i \in \Sigma, l_x \leq \varphi_i.f_i(X_i) - Y_i \leq u_i$. If φ_x is implied by Σ, then $\Sigma.Fes \subseteq \varphi_x.Fes$. Pick any point $d \in \Sigma.Fes$, d is also in $\varphi_x.Fes$. That means, (1) $\forall\varphi_i \in \Sigma, l_x \leq \varphi_i.f_i(X_i) - Y_i \leq u_i$ and (2) $l_x \leq \varphi_x.f_x(X_x) - Y_x \leq u_x$. Case (1) and (2) reveal two optimizing problem: $i).\max(l_x - \varphi_x.f_x(X_x) + Y_x) \leq 0$ and $ii).\max(\varphi_x.f_x(X_x) - Y_x - u_x) \leq 0$, which is satisfied under the condition that $\forall\varphi_i \in \Sigma, l_x \leq$

$\varphi_i.f_i(X_i) - Y_i \leq u_i$. It can be derived that $z^L \leq 0$ and $z^R \leq 0$ are always satisfied under $\forall \varphi_i \in \Sigma, l_x \leq \varphi_i.f_i(X_i) - Y_i \leq u_i$, and then the completeness holds. ■

From the above, it is not difficult to find that the complete inference system we designed can reason against the counter example in Fig 8. Plugging the parameters into the equation, we are able to calculate that $z^L = 0$ and $z^R = -0.5$ under the precondition that $\forall \varphi_i \in \Sigma, l_x \leq \varphi_i.f_i(X_i) - Y_i \leq u_i$. Therefore, φ_x is implied by Σ through the designed inference system.

C. Proof of Proposition 4

Proof: (*Necessity*) If the STCD set Σ is consistent, one can find an instance I_d that satisfies Σ. Base on the data instance, we can construct the corresponding point d. Since $I_d \models \Sigma$, point $d \in \Sigma.Fes$. Therefore, $\Sigma.Fes \neq \phi$, the necessity holds. (*Sufficiency*) If $\Sigma.Fes \neq \phi$, then $\exists d \in \Sigma.Fes$. From point d, we can construct the corresponding data instance I_d having $I_d \models \Sigma$. According to the definition of consistency, Σ is consistent. ■

REFERENCES

[1] A. Karkouch, H. Mousannif, H. A. Moatassime, and T. Noël, "Data quality in internet of things: A state-of-the-art survey," *J. Netw. Comput. Appl.*, vol. 73, pp. 57–81, 2016.

[2] X. Wang and C. Wang, "Time series data cleaning: A survey," *IEEE Access*, vol. 8, pp. 1866–1881, 2020.

[3] S. Song and A. Zhang, "Iot data quality," in *CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020*, M. d'Aquin, S. Dietze, C. Hauff, E. Curry, and P. Cudré-Mauroux, Eds. ACM, 2020, pp. 3517–3518.

[4] X. Ding, H. Wang, J. Su, Z. Li, J. Li, and H. Gao, "Cleanits: A data cleaning system for industrial time series," *PVLDB*, vol. 12, no. 12, pp. 1786–1789, 2019.

[5] Z. Liang, H. Wang, X. Ding, and T. Mu, "Industrial time series determinative anomaly detection based on constraint hypergraph," *Knowl. Based Syst.*, vol. 233, p. 107548, 2021. [Online]. Available: https://doi.org/10.1016/j.knosys.2021.107548

[6] S. Song, A. Zhang, J. Wang, and P. S. Yu, "SCREEN: stream data cleaning under speed constraints," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, May 31 - June 4, 2015*, pp. 827–841.

[7] J. Liang and S. Parthasarathy, "Robust contextual outlier detection: Where context meets sparsity," in *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*, S. Mukhopadhyay, C. Zhai, E. Bertino, F. Crestani, J. Mostafa, J. Tang, L. Si, X. Zhou, Y. Chang, Y. Li, and P. Sondhi, Eds. ACM, 2016, pp. 2167–2172.

[8] S. Song, F. Gao, R. Huang, and C. Wang, "Data dependencies extended for variety and veracity: A family tree," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 10, pp. 4717–4736, 2022. [Online]. Available: https://doi.org/10.1109/TKDE.2020.3046443

[9] X. Chu, I. F. Ilyas, and P. Papotti, "Discovering denial constraints," *Proc. VLDB Endow.*, vol. 6, no. 13, pp. 1498–1509, 2013. [Online]. Available: http://www.vldb.org/pvldb/vol6/p1498-papotti.pdf

[10] P. Langer and F. Naumann, "Efficient order dependency detection," *VLDB J.*, vol. 25, no. 2, pp. 223–241, 2016.

[11] L. Golab, H. J. Karloff, F. Korn, A. Saha, and D. Srivastava, "Sequential dependencies," *PVLDB*, vol. 2, no. 1, pp. 574–585, 2009.

[12] T. Dasu, R. Duan, and D. Srivastava, "Data quality for temporal streams," *IEEE Data Eng. Bull.*, vol. 39, no. 2, pp. 78–92, 2016.

[13] S. Abiteboul, R. Hull, and V. Vianu, *Foundations of Databases*. Addison-Wesley, 1995. [Online]. Available: http://webdam.inria.fr/Alice/

[14] A. A. Qahtan, N. Tang, M. Ouzzani, Y. Cao, and M. Stonebraker, "Pattern functional dependencies for data cleaning," *Proc. VLDB Endow.*, vol. 13, no. 5, pp. 684–697, 2020. [Online]. Available: http://www.vldb.org/pvldb/vol13/p684-qahtan.pdf

[15] M. Baudinet, J. Chomicki, and P. Wolper, "Constraint-generating dependencies," *J. Comput. Syst. Sci.*, vol. 59, no. 1, pp. 94–115, 1999. [Online]. Available: https://doi.org/10.1006/jcss.1999.1632

[16] K. Huang and N. D. Sidiropoulos, "Consensus-admm for general quadratically constrained quadratic programming," *IEEE Transactions on Signal Processing*, vol. 64, no. 20, pp. 5297–5310, 2016.

[17] A. Schrijver, *Theory of linear and integer programming*. John Wiley & Sons, 1998.

[18] E. B. Bajalinov, *Linear-fractional programming theory, methods, applications and software*. Springer Science & Business Media, 2003, vol. 84.

[19] M. K. Kozlov, S. P. Tarasov, and L. G. Khachiyan, "The polynomial solvability of convex quadratic programming," *USSR Computational Mathematics and Mathematical Physics*, vol. 20, no. 5, pp. 223–228, 1980.

[20] I. F. Ilyas and X. Chu, *Data Cleaning*. ACM, 2019. [Online]. Available: https://doi.org/10.1145/3310205

[21] K. G. Jamieson and A. Talwalkar, "Non-stochastic best arm identification and hyperparameter optimization," in *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS 2016, Cadiz, Spain, May 9-11, 2016*, 2016, pp. 240–248.

[22] L. Li, K. G. Jamieson, G. DeSalvo, A. Rostamizadeh, and A. Talwalkar, "Hyperband: A novel bandit-based approach to hyperparameter optimization," *J. Mach. Learn. Res.*, vol. 18, pp. 185:1–185:52, 2017.

[23] Z. S. Karnin, T. Koren, and O. Somekh, "Almost optimal exploration in multi-armed bandits," in *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, 2013, pp. 1238–1246.

[24] M. R. Osborne and P. B. A. Turlach, "On the lasso and its dual," *Journal of Computational & Graphical Statistics*, vol. 9, no. 2, pp. 319–337, 2000.

[25] C. M. Carvalho, N. G. Polson, and J. G. Scott, "Handling sparsity via the horseshoe," in *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics, AISTATS*, 2009, pp. 73–80.

[26] W. L. Cava, P. Orzechowski, B. Burlacu, F. O. de Francca, M. Virgolin, Y. Jin, M. Kommenda, and J. H. Moore, "Contemporary symbolic regression methods and their relative performance," *ArXiv*, vol. abs/2107.14351, 2021.

[27] M. Mohri, A. Rostamizadeh, and A. Talwalkar, *Foundations of machine learning*. MIT press, 2018.

[28] D. Bolton, "The computation of equivalent potential temperature," *Monthly weather review*, vol. 108, no. 7, pp. 1046–1053, 1980.

[29] C. L. Vaughan and M. J. OMalley, "Froude and the contribution of naval architecture to our understanding of bipedal locomotion," *Gait & posture*, vol. 21, no. 3, pp. 350–362, 2005.

[30] A. Zhang, S. Song, J. Wang, and P. S. Yu, "Time series data cleaning: From anomaly detection to anomaly repairing," *PVLDB*, vol. 10, no. 10, pp. 1046–1057, 2017.

[31] S. K. Jensen, T. B. Pedersen, and C. Thomsen, "Time series management systems: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 29, no. 11, pp. 2581–2600, 2017.

[32] T. Bleifuß, S. Kruse, and F. Naumann, "Efficient denial constraint discovery with hydra," *Proceedings of the VLDB Endowment*, vol. 11, no. 3, pp. 311–323, 2017.

[33] T. Bleifuß, S. Bülow, J. Frohnhofen, J. Risch, G. Wiese, S. Kruse, T. Papenbrock, and F. Naumann, "Approximate discovery of functional dependencies for large datasets," in *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016*, 2016, pp. 1803–1812.

[34] L. Caruccio, V. Deufemia, F. Naumann, and G. Polese, "Discovering relaxed functional dependencies based on multi-attribute dominance," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 9, pp. 3212–3228, 2021.

[35] E. Livshits, A. Heidari, I. F. Ilyas, and B. Kimelfeld, "Approximate denial constraints," *Proc. VLDB Endow.*, vol. 13, no. 10, pp. 1682–1695, 2020.

[36] S. Song, F. Gao, A. Zhang, J. Wang, and P. S. Yu, "Stream data cleaning under speed and acceleration constraints," *ACM Trans. Database Syst.*, vol. 46, no. 3, pp. 10:1–10:44, 2021.

[37] J. R. Koza, "Genetic programming as a means for programming computers by natural selection," *Statistics and computing*, vol. 4, no. 2, pp. 87–112, 1994.

[38] M. Schmidt and H. Lipson, "Distilling free-form natural laws from experimental data," *science*, vol. 324, no. 5923, pp. 81–85, 2009.

[39] M. Virgolin, T. Alderliesten, C. Witteveen, and P. A. N. Bosman, "Improving model-based genetic programming for symbolic regression of small expressions," *Evolutionary Computation*, vol. 29, pp. 211–237, 2021.
[40] S.-M. Udrescu and M. Tegmark, "Ai feynman: A physics-inspired method for symbolic regression," *Science Advances*, vol. 6, 2020.
[41] B. K. Petersen and M. Landajuela, "Deep symbolic regression: Recovering mathematical expressions from data via risk-seeking policy gradients," *arXiv: Learning*, 2021.
[42] S. d'Ascoli, P.-A. Kamienny, G. Lample, and F. Charton, "Deep symbolic regression for recurrent sequences," *arXiv preprint arXiv:2201.04600*, 2022.