Time Series Data Cleaning: A Survey
(Technical Report)

Xi Wang, Chen Wang
School of Software, Tsinghua University, Beijing 100084, China

Abstract—Errors are prevalent in time series data, which is particularly common in the industrial field. Data with errors could not be stored in the database, which results in the loss of data assets. At present, to deal with these time series containing errors, besides keeping original erroneous data, discarding erroneous data and manually checking erroneous data, we can also use the cleaning algorithm widely used in the database to automatically clean the time series data. This survey provides a classification of time series data cleaning techniques and comprehensively reviews the state-of-the-art methods of each type. Besides we summarize data cleaning tools, systems and evaluation criteria from research and industry. Finally, we highlight possible directions time series data cleaning.

I. INTRODUCTION

Time series data can be defined as a sequence of random variables, \( x_1, x_2, \ldots, x_n \), where the random variable \( x_1 \) denotes the value taken by the series at the first time point, the variable \( x_2 \) denotes the value for the second time period, \( x_n \) denotes the value for the \( n \)-th time period, and so on. Time series have been widely used in many fields such as financial economy, meteorology and hydrology, signal processing, industrial manufacturing, and so on. Time series data are important in industry, where there are all kinds of sensor devices capturing data from the industrial environment uninterrupted. Owing to the fact that data of the sensor devices are often unreliable, time series data are often large and dirty. In the financial field, the most important application of time series data is to predict future commodity (stock) price movements. However, time series errors in the financial field are also very prevalent, even some data sets, which are considered quite accurate, still have erroneous data. For instance, the correct rate of stock information on Yahoo is 93%. The costs and risks of errors, conflicts, and inconsistencies in time series have drawn widespread attention from businesses and government agencies. In recent work, the data quality issues in time series data are studied, since they pose unique data quality challenges due to the presence of autocorrelations, trends, seasonality, and gaps in the time series data. According to Shilakes et al., the relevant market growth rate of data quality is about 17%, which is much higher than the 7% annual growth rate of the IT industry. For instance, approximately 30% to 80% of the time and cost are spent on data cleaning in data warehousing project development. The time series errors can be either timestamp errors or observed value errors. In this survey, we focus on the existing methods of dealing with observed value errors, thereby, the time series errors mentioned in the following are observation errors. There are two types of processing methods commonly used in the industry when dealing with time series data errors: (1) Discarding erroneous data. First, the time series is detected via using an anomaly detection algorithm, and then the detected abnormal data are discarded. (2) Cleaning data. Data cleaning is divided into manual cleaning and automatic cleaning. There is no doubt that manual cleaning has a high accuracy rate, but it is difficult to implement because it takes more time and effort.

The existing surveys of data cleaning mainly summarize the methods of dealing with data missing, data inconsistency, data integration and erroneous data in the database. Karkouch et al. review the generation of sensor data, the reasons for the formation of data quality problems, and the techniques for improving data quality. However, Karkouch et al. do not provide a detailed overview of the existing state-of-the-art of erroneous data cleaning. Thereby, we review the state-of-the-art of time series data error value cleaning, which may provide a tutorial for others.

A. Problem Statement

In this study, enlightened by related research on the classification of time series error types, we summarize the common error in time series into three categories, namely, single point big error, single point small error and continuous errors. This article takes the stock price of a stock for 30 consecutive trading days as an example. As shown in Figure 1, the characteristics of these three types of errors are described in detail. The red line in the figure indicates the true price of the stock in 30 consecutive trading days, and the blue line indicates the price of the stock crawled by a website. For various reasons, the observed value may not be the same as the actual value. It can be seen that in the four consecutive trading days of 8-11, the observed values are all 0, and the true values are 1.3, 1.2, 1.1 and 1.15, respectively, on the 20th trading day, the observed value is 2.4 and the true value is 1.3, on the 25th trading day, the observed value is 1 and the true value is 1.3.

(1) Continuous errors. The so-called continuous errors, that is, in the time series, errors occur in several consecutive time points. Specifically, continuous errors can continue to be subdivided into several types, but no longer detailed here. The observed values from the 8th to 11th trading days in Figure 1 are all 0, that is, continuous errors occur here. Continuous errors are common in real life. For instance, when someone is holding a smartphone and walking on the
For the problem of time series data cleaning, the following four difficulties have been discovered through the survey:

1. The amount of data is large and the error rate is high. The main source of time series data is sensor acquisition. Especially in the industrial field, sensors distributed throughout the machine are constantly monitoring the operation of the machine in real-time. These sensors often collect data at a frequency of seconds, and the amount of data collected is quite large. For instance, the sensor collection interval of a wind power company equipment is 7 seconds, each machine has more than 2000 working status data, and more than 30 million pieces of data are collected every day, so the working status data of one day could exceed 60 billion. However, the data collected by the sensor are often not accurate enough, and some because of the physical quantity of the observation is difficult to measure accurately. For instance, in a steel mill, with affecting by environmental disturbances the surface temperature of the continuous casting slab cannot be accurately measured or may cause distortion due to the power of the sensor itself.

2. The reasons for generating time series data errors are complicated. People always try to avoid the generation of time series erroneous data, however, there are various time series errors. Besides the observed errors that we mentioned above for various reasons, Karkouch et al. [12] also explain in detail the IoT data errors generated by various complex environments. IoT data is a common time series of data, and its widespread existence is really in the world. The complex reasons of time series errors also are challenges we face in cleaning and analyzing data that is different from traditional relational data.

3. Time series data are continuously generated and stored. The biggest difference between time series data and relational data is that the time series is continuous. Thereby, for time series data, it is important that the cleaning algorithm supports online operations (real-time operations). The online anomaly detection or cleaning algorithm can monitor the physical quantity in real-time, detect the problem and then promptly alarm or perform a reasonable cleaning. Thereby, the time series cleaning algorithm is not only required to support
online calculation or streaming calculation but also has good throughput.

(4) Minimum modification principle \cite{17}–\cite{19}. Time series data often contain many errors. Most of the widely used time series cleaning methods utilize the principle of smooth filtering. Such methods may change the original data too much, and result in the loss of the information contained in the original data. Data cleaning needs to avoid changing the original correct data. It should be based on the principle of minimum modification, that is, the smaller the change, the better.

C. Organization

Different algorithms tackle these challenges in different ways, which usually include smoothing-based methods, constraint-based methods, and statistical-based methods as shown in Table I. Besides some time series anomaly detection algorithms can also be effectively used to clean data. The remainder of this paper is organized as follows. The aforesaid four types of algorithms are discussed from Section II to V. In Section VII, we introduce existing time series cleaning tools, systems, and evaluation criteria. Finally, we summarize this paper in Section VII and discuss possible future directions.

II. SMOOTHING BASED CLEANING ALGORITHM

Smoothing techniques are often used to eliminate data noise, especially numerical data noise. Low-pass filtering, which filters out the lower frequency of the data set, is a simple algorithm. The characteristic of this type of technology is that the time overhead is small, but because the original data may be modified much, which makes the data distorted and leads to the uncertainty of the analysis results, there are not many applications used in time series cleaning. The research of smoothing technology mainly focuses on algorithms such as Moving Average (MA) \cite{20}, Autoregressive (AR) \cite{4}, \cite{21}, \cite{22} and Kalman filter model \cite{25}–\cite{27}. Thereby, this chapter mainly introduce these three technologies and their extensions.

A. Moving average

The moving average (MA) series algorithm \cite{20} is widely used in time series for smoothing and time series prediction. A simple moving average (SMA) algorithm: Calculate the average of the most recently observed \( N \) time series values, which is used to predict the value at time \( t \). A simple definition as shown in equation (1).

\[
\hat{x}_t = \frac{1}{2n+1} \sum_{i=-n}^{n} x_{i+t} 
\]

In equation (1), \( \hat{x}_t \) is the predicted value of \( x_t \). \( x_t \) represents the true value at time \( t \). \( 2n+1 \) is the window size(count). For a time series \( x(t) = 9.8, 8.5, 5.4, 5.6, 5.9, 9.2, 7.4 \), for simplicity, we use SMA with \( n = 3 \) and \( k = 4 \), then calculate according to equation (2).

\[
x_t = \frac{(x_{t-n} + x_{t-2} + \ldots + x_{t+n})}{(2n+1)} = 7.4
\]

To eliminate errors or noises, the data \( x(t) \) can be considered as a certain time window of sliding window, then continuously calculate the local average over a given interval \( 2n+1 \) to filter out the noise (erroneous data) and get a smoother measurement.

In the weighted moving average (WMA) algorithm, data points at different relative positions in the window have different weights. Generally defined as:

\[
\hat{x}_t = \sum_{i=-n}^{n} \omega_i x_{i+t}
\]

In equation (3), \( \omega_i \) represents the weight of the influence of the \( i \) position data point on the \( t \) position data point, other definitions follow the example above. A simple strategy is that the farther away from the two data points, the smaller the mutual influence. For instance, a natural idea is the reciprocal of the distance between two data points as the weight of their mutual influence. Similarly, the weight of each data point in the exponential weighted moving average (EWMA) algorithm \cite{32} decreases exponentially with increasing distance, which is mainly used for unsteady time series \cite{56}, \cite{57}.

Aiming at the need for the rapid response of sensor data cleaning, Zhuang et al. \cite{31} propose an intelligent weighted moving average algorithm, which calculates weighted moving
averages via collected confidence data from sensors. Zhang et al. [83] propose a method based on multi-threshold control and approximate positive transform to clean the probe vehicle data, and fill the lost data with the weighted average method and exponential smoothing method. Qu et al. [89] first use cluster-based methods for anomaly detection and then use exponentially weighted averaging for data repair, which is used to clean power data in a distributed environment.

### B. Autoregressive

The Autoregressive (AR) Model is a process that uses itself as a regression variable and uses the linear combination of the previous \(k\) random variables to describe the linear regression model of the random variable at the time \(t\). The definition of AR model [21], [90] as shown in equation (4).

\[
\hat{x}_t = \sum_{i=1}^{k} \omega_i x_{t-i} + \epsilon_t + a
\]  

(4)

In equation (4), \(\hat{x}_t\) is the predicted value of \(x_t\), \(x_t\) represents the true value at time \(t\), \(k\) is the order, \(\mu\) is mean value of the process, \(\epsilon_t\) is white noise, \(\omega_i\) is the parameter of the model, \(a\) is a constant.

Park et al. [91] use labeled data \(y\) to propose an autoregressive with exogenous input (ARX) model based on the AR model:

\[
\hat{y}_t = x_t + \sum_{i=1}^{k} \omega_i (y_{t-i} - x_{t-i}) + \epsilon_t
\]  

(5)

In equation (5), \(\hat{y}_t\) is the possible repair of \(x_t\), and others are the same to the aforesaid AR model. Alengrin et al. [24] propose Autoregressive moving average (ARMA) model), which is composed of the AR model and MA model. Besides that, the Gaussian Autoregressive Moving Average model is defined as shown in equation (6) [13].

\[
\Phi(B)Z_t = \theta_0 + \theta(B)x_t
\]  

(6)

In equation (6), \(\Phi(B) = 1 - \Phi_1 B - ... - \Phi_p B^p\) and \(\theta(B) = 1 - \theta_1 B - ... - \theta_q B^q\) are polynomial in \(B\) of degrees \(p\) and \(q\), respectively, \(\theta_0\) is a constant, \(B\) is the backshift operator such that \(B Z_{t-1} = Z_{t-1}\), and \(x_t\) is a sequence of independent Gaussian variates with mean \(\mu=0\) and variance \(\sigma^2_x\). Box et al. [92] propose a more complex Autoregressive Integrated Moving Average (ARIMA) model based on the ARMA model, which is not described in detail here. Akouemo et al. [93] propose a method combining ARX and Artificial Neural Network (ANN) model for cleaning time series, which performs a hypothesis test to detect anomalies the extrema of the residuals, and repairs anomalous data points by using the ARX and ANN models. Dilling et al. [23] clean high-frequency velocity profile data with ARMA model and Chen et al. [94] use the ARIMA model to clean wind power data.

### C. Kalman filter model

Kalman [25] proposes the Kalman filter theory, which can deal with time-varying systems, non-stationary signals, and multi-dimensional signals. Kalman filter creatively incorporates errors (predictive and measurement errors) into the calculation, the errors exist independently and are not affected by measured data. The Kalman model involves probability, random variable, Gaussian Distribution, and State-space Model, etc. Consider that the Kalman model involves too much content, no specific description is given here, and only a simple definition is given. First, we introduce a system of discrete control processes which can be described by a Linear Stochastic Difference equation as shown in equation (7).

\[
x_t = m x_{t-1} + n v_t + p_t
\]  

(7)

Also, the measured values of the system are expressed as shown in equation (8).

\[
y_t = r x_{t-1} + q_t
\]  

(8)

In equation (7) and (8), \(x_t\) is the system state value at time \(t\), \(v_t\) is the control variable value for the system at time \(t\), \(m\) and \(n\) are system parameters, and for multi-model systems, they are matrices, \(q_t\) is the measured value at time \(t\), \(r\) is the parameter of the measurement system, and for multi-measurement systems, \(r\) is a matrix, \(p(k)\) and \(q(k)\) represent the noises of the process and measurement, respectively, and they are assumed to be white Gaussian Noise.

The extended Kalman filter is the most widely used estimation for a recursive nonlinear system because it simply linearizes the nonlinear system models. However, the extended Kalman filter has two drawbacks: linearization can produce unstable filters and it is hard to implement the derivation of the Jacobian matrices. Thereby, Ma et al. [95] present a new method of predicting the Mackey-Glass equation based on the unscented Kalman filter to solve these problems. In the field of signal processing, there are many works [28]–[30] based on Kalman filtering, but these techniques have not been widely used in the field of time series cleaning. Gardner et al. [32] propose a new model, which is based on the Kernel Kalman Filter, to perform various nonlinear time series processing. Zhuang et al. [33] use the Kalman filter model to predict sensor data and smoothed it with WMA.

### D. Trajectory Simplification

The main purpose of trajectory simplification is to reduce the original trajectory from \(N\) trajectory data points to \(M\) trajectory data points, \(M < N\). And retain important location or topological features. The compression algorithm needs to ensure that the sequence of \(M\) points has a minimum compression error (M-ε problem), Such as MRPA [37], Error-Search [38]. The input of the Error-Search algorithm is the target compression ratio \(\lambda\), Original track \(T\). Returns the compression that satisfies the given compression ratio \(\lambda\) with the smallest DAD error. The main steps of the Error-Search algorithm are as follows: (1) According to the original trajectory \(T\), Build search space \(\epsilon\), create a search \(\epsilon\) space based on the opposite direction in a more efficient way Trace \(T\). (2) Error checking, the main task is to determine whether there is a simplified trajectory \(T\), satisfy \(DAD(T) < \epsilon\). Error-Search is designed to retain the direction information of the track. The time complexity of the algorithm is \(O(N^2 \log N)\), the space complexity is \(O(N^2)\). Error-Search is an accurate trajectory...
simplification algorithm that retains direction information, but the time complexity is too high. [38] proposed an approximate algorithm Span-Search algorithm. Span-Search algorithm uses approximate error to replace time complexity $O(N \log^2 N)$ space complexity $O(N)$.

E. Summary and Discussion

As shown in Table II there are many methods based on smoothing, such as the state-space model [26], [35], [36] and Interpolation [12], [14]. The state-space model assumes that the system’s change over time can be determined by an unobservable vector sequence, the relationship between the time series and the observable sequence can be determined by the state-space model. By establishing state equations and observation equations, the state-space model provides a model framework to fully describe the temporal characteristics of dynamic systems. To make this kind of smoothing algorithm have a better effect, many studies [96–98] have also proposed various techniques to estimate the parameters in the above methods. Most smoothing techniques, when cleaning time series, have a small-time overhead, but it is very easy to change the original correct data points, which greatly affects the accuracy of cleaning. In other words, correct data are altered, which can distort the results of the analysis and lead to uncertainty in the results.

III. CONSTRAINT BASED CLEANING ALGORITHM

In this section, we introduce several typical algorithms, which include order dependencies (ODs) [39], sequential dependencies (SDs) [44] and speed constraints [45], for repairing time series errors.

A. Order Dependencies

In relational databases, Order Dependencies (ODs) are simple and effective methods, which have been widely studied [39–42]. We find that ODs are also suitable for solving some time series data cleaning problems. The specific explanation is as follows: Let $x(t) = x_1, x_2, ... x_t$ be a time series, ODs can be expressed by $<, \leq, >, \geq$. For the number of miles traveled by the car $x(t)$, the mileage should increase over time.

Formal representation is as follows: $t_1 < t_2$ then $x_{t_1} \leq x_{t_2}$ where $x(t)$ is mileage, $t$ is timestamp. For instance, consider an example relation instance in Table III. The tuples are sorted on attribute sequence number, which identifies sea level that rapidly increase from hour to hour.

Generally, ODs in the form of equation (9) states that $N$ is strictly increasing with $M$. Such as equation (10).

$$M \rightarrow_{(0, \infty)} N$$

$$\text{hour} \rightarrow_{(0, \infty)} \text{height}$$

ODs and DCs can also be used as an integrity constraint for error detection and data repairing in databases. Wijse [99], [100] extends ODs with a time dimension for temporal databases. Let $I = \{I_1, I_2, I_3, \ldots \}$ be a temporal relation, which can be viewed as a time series of conventional “snapshot” relations, all over the same set of attributes. A trend dependency (TD) allows attributes with linearly ordered domains to be compared over time by using any operator of $\{<, =, >, \leq, \geq, \neq \}$. Consider the constraint is specified over $(I_i, I_{i+1})$ in $I$. For each time point $i$, it requires comparing employee records at time $i$ with records at the next time $i + 1$, such that salaries of employees should never decrease. Lopatenko et al. [43] propose a numerical type data cleaning method based on Denial Constraints (DCs) as constraints, whose principle is similar to this one.

B. Sequential Dependencies

The sequential dependency algorithm proposed by Golab et al. [44] focuses on the difference in values between two consecutive data points in a time series. Golab et al. [43] define the CSD Tableau Discovery Problem as given a relation instance and an embedded SD $M \rightarrow_g N$, to find a tableau $t_r$ of minimum size such that the CSD $(M \rightarrow_g N, t_r)$ has confidence at least a given threshold. A CSD can be $(\text{hour} \rightarrow_{(0, \infty)} \text{height}, [1961.01.01 00:00–2016.01.01 00:00])$.

It states that for any two consecutive hours in [1961.01.01 00:00–2016.01.01 00:00], their distance should always be $> 0$. Generally, a sequential dependency (SD) is in the form of

$$M \rightarrow_g N.$$  \hspace{1cm} (11)

In equation (11), $M \subseteq R$ are ordered attributes, $N \subseteq R$ can be measured by certain distance metrics, and $g$ is an interval. It states that when tuples are sorted on $M$, the distance between the $N$-values of any two consecutive tuples are within interval

| Reference | Method |
|-----------|--------|
| [20]      | MA     |
| [31], [83]| WMA    |
| [28], [88], [89] | EWMA |
| [32]      | AR     |
| [25], [27] | ARX    |
| [26], [34] | ARMA   |
| [22], [23] | Kalman Filter Model |
| [30]      | The Unscented Kalman Filter |
| [35]      | The State-space Model |
| [36]      | Interpolation |
| [96–98]   | To Estimate the Parameters of Model |

| Table II Summary of Smoothing |
| reference | Method |
|----------|--------|
| [20]     | MA     |
| [31], [83]| WMA    |
| [28], [88], [89] | EWMA |
| [32]      | AR     |
| [25], [27] | ARX    |
| [26], [34] | ARMA   |
| [22], [23] | Kalman Filter Model |
| [30]      | The Unscented Kalman Filter |
| [35]      | The State-space Model |
| [36]      | Interpolation |
| [96–98]   | To Estimate the Parameters of Model |

| Table III An Example of Order Dependencies |
| sequence number | time |
|-----------------|-----|
| $t_1$           | 1   |
| $t_2$           | 2   |
| $t_3$           | 3   |
| $t_4$           | 4   |
| $t_5$           | 5   |
| $t_6$           | 6   |
| $t_7$           | 7   |
TABLE IV
AN EXAMPLE RELATION INSTANCE OF TIME SERIES

| time | value |
|------|-------|
| t₁   | 1     |
| t₂   | 2     |
| t₃   | 3     |
| t₄   | 4     |
| t₅   | 5     |
| t₆   | 6     |
| t₇   | 7     |
| t₈   | 8     |
| t₉   | 9     |

- Fischer et al. \[101\] propose the concept of streaming mode to represent the structure and semantic constraints of data streams. The concept contains a variety of semantic information, including not only numeric values, but also attributes between time. The sequential dependency algorithm can be used not only for traditional relational database cleaning, but also for time series cleaning. In fact, there are many dependency-based cleaning algorithms designed for relational databases that are not suitable for time series data cleaning, such as: Functional Dependencies \[102\] (FDs), Conditional Functional Dependencies \[103\], \[104\] (CFDs), Differential Dependencies (DDs) \[105\], \[106\] or Comparable Dependencies (CDs) \[107\], \[108\]. The sequential dependency is one of the few algorithms based on dependency that can be used for time series cleaning.

C. Speed Constraints

To clean time series data, speed constraint-based methods \[45\] considers the restrictions of speed on value changes in a given interval. As we have learned some common sense, e.g., the maximum flying speed of a bird, temperatures in a day, car mileage, etc. Consider with time window T is a pair of minimum speed \(S_{\text{min}}\) and maximum speed \(S_{\text{max}}\) over the time series \(x = x₁, x₂, \ldots, xₙ\), where each \(xᵢ\) is the value of the \(i\)-th data point, with a timestamp \(i\).

For instance, consider time series:
\[x(t) = \{150, 160, 170, 180, 110, 200, 210, 220, 230\}\] where timestamps \(t = \{1, 2, 3, 4, 5, 6, 7, 8, 9\}\). The value attribute corresponds to \(x\), while the time attribute in Table IV denotes the timestamps.

Suppose a window size \(T = 2\), \(S_{\text{min}} = -50\), and \(S_{\text{max}} = 50\) in the speed constraints, for data points \(x₅\) and \(x₄\), \(\frac{110-180}{8-5} = -80 < -50\), Similarly, \(x₇\) and \(x₈\) with speed \(\frac{200-110}{6-5} = 90 > 50\) are violations to \(S_{\text{max}} = 50\). To remedy the violations (denoted by red lines), a repair on \(x₅\) can be performed, i.e., \(x₅' = 190\), which is represented by the blue “+” symbol in Figure 3. As illustrated in Figure 3, the repaired sequence satisfies both the maximum and minimum speed constraints.

Generally, a speed constraint is in the form of equation (12).

\[S = (S_{\text{min}}, S_{\text{max}})\]  \hspace{1cm} (12)

If time series data \(x\) satisfies the speed constraint \(S\), then for any \(xᵢ, xⱼ\) in a time window \(T\), it has \(S_{\text{min}} < \frac{xⱼ - xᵢ}{ⱼ - i} < S_{\text{max}}\). In practical applications, speed constraints are often valid for a specific period of time. For instance, when considering the fastest speed of a car, the time period of the reference is often in hours, and two data points in different years are not considered. The value of the speed constraints \(S\) may be positive (the growth rate of the constraint value) or negative (the rate of decline of the constraint value). Speed constraints are less effective when dealing with small errors, and Wei Yin et al. \[46\] propose a further study of variance constraints, which use the variance threshold \(V\) to measure the degree of dispersion of the time series in a given \(W\) window.

D. Temporal Dependence

Discovering and exploiting temporal dependence is highly desired in many applications. For example, consider health care, analysis of temporal dependence can be of value throughout the entire disease treatment process, from disease prevention to treatment. A temporal dependence contains two components: the causative behavior and the dependent behavior. The users future behavior is affected by the causative behavior and the causative behavior causes the dependent behavior. For one more example, in E-commerce networks, accurate analysis from users time series activity records is of significant importance for advertising, marketing, and psychological studies. Qingchao Ca et al. \[49\] proposed recurrent cohort analysis to group users into cohorts, and Dehua Cheng et al. \[50\] improved the performance of temporal dependence recovery by reversing the time stamps of original time series and combine both time series.

E. Summary and Discussion

In the field of relational databases, there are many cleaning algorithms based on integrity constraints, which are difficult to apply in the field of time series in which the observed values are substantially numerical, because they follow a strict equality relationship. A few methods, which we summarize in Table IV, can be used for time series data cleaning, for instance ODs and SDs can be used to solve problems in some scenarios, such as the number of miles in a car is non-decreasing. Further speed-based constraints can be used to process data such as GPS and stock prices, but only with relevant domain knowledge can give a reasonable constraint. Therefore, the constraint-based cleaning algorithm needs to
be further improved to have better robustness. Similarity rules \cite{47} can capture a more general form of constraints on similarities between data values \cite{109}. Moreover, learning individual models \cite{48} could help in repairing missing data in different scenarios. One possible future direction is to use anomaly detection methods to detect anomalies first, and then treat outliers as missing to repair. We will discuss anomaly detection in Section V.

IV. STATISTICS BASED CLEANING ALGORITHM

Statistical-based cleaning algorithms occupy an important position in the field of data cleaning. Such algorithms use models, which learned from data, to clean data. The statistical-based approach involves a lot of statistical knowledge, but this article focuses on statistical-based data cleaning methods, so we won’t cover statistical-related knowledge in detail.

A. Maximum likelihood

The intuitive idea of the maximum likelihood principle is a random test, if there are several possible outcomes \(x_1, x_2, \ldots, x_t\), if the result \(x_t\) occurs in one test, it is generally considered that the test conditions are favorable for \(x_t\), or think that \(x_t\) has the highest probability of occurrence.

Notation: For a given time series data \(x(t)\), which consists with a probability distribution \(d\), and assume that its probability aggregation function (discrete distribution) is \(F_d\); consider a distribution parameter \(\theta\), sampling \(x_1, x_2, \ldots, x_n\) from this distribution, then use \(F_d\) to calculate its probability \cite{51} as shown in equation \((13)\).

\[
P = (x_1, x_2, \ldots, x_n) = F_d(x_1, x_2, \ldots, x_n|\theta) \tag{13}
\]

Ziekow et al. \cite{52} use the maximum likelihood technique to clean Radio Frequency Identification (RFID) data. Wang et al. \cite{53} propose the first maximum likelihood solution to address the challenge of truth discovery from noisy social sensing data. Yakout et al. \cite{54} argue a new data repairing approach that is based on maximizing the likelihood of replacement data in the given data distribution, which can be modeled using statistical machine learning techniques, but this technology is used to repair the data of the database. For the repairing of time series data errors, Zhang et al. \cite{55} propose a better solution based on maximum likelihood, which solves the problem from the perspective of probability. According to the probability distribution of the speed change of adjacent data points in the time series, the time series cleaning problem can be converted to find a cleaned time series, which is based on the probability of speed change that has the greatest likelihood.

B. Markov model

Markov process is a class of stochastic processes, which means that the transition of each state in the process depends only on the previous \(n\) states. This process is called a \(n\)-order model, where \(n\) is the number that affects the transition state. The simplest Markov process is the first-order process, and the transition of each state depends only on the state before it. Time and state are discrete Markov processes called Markov chains, abbreviated as \(X_n = X(n), n = 0, 1, 2, \ldots\). The Markov chain \cite{58} is a sequence of random variables \(X_1, X_2, X_3, \ldots\). The range of these variables, that is, the set of all their possible values, is called the "state space", and the value of \(X_n\) is the state of time \(n\).

The Markov Model \cite{59, 60} is a statistical model based on Markov chain, which is widely used in speech recognition, part-of-speech automatic annotation, phonetic conversion, probabilistic grammar and other natural language processing applications. In order to find patterns that change over time, the Markov model attempts to build a process model that can generate patterns. \cite{59} and \cite{60} use specific time steps, states, and make Markov assumptions. With these assumptions, this ability to generate a pattern system is a Markov process. A Markov process consists of an initial vector and a state transition matrix. One thing to note about this assumption is that the state transition probability does not change over time.

Hidden Markov Model (HMM) \cite{61, 62} is a statistical model based on Markov Model, which is used to describe a Markov process with implicit unknown parameters. The difficulty is to determine the implicit parameters of the process from observable parameters, and then use these parameters for further analysis, such as prediction of time series data. For instance, after rolling the dice 10 times, we could get a string of numbers, for example we might get such a string of numbers: 1, 4, 5, 3, 1, 6, 2, 4, 5 as shown in Figure 4. This string of numbers is called the visible state chain. But in HMM, we not only have such a string of visible state chains, but also a chain of implied state chains. In this example, the implicit state chain might be: \(D_5, D_3, D_2, D_3, D_4, D_6, D_1, D_5, D_1, D_2\).

Gupta et al. \cite{63} use HMM to predict the price of stocks. Baba et al. \cite{64} argue a data cleaning method based on the HMM, which used to clean RFID data related to geographic location information. In multi-dimensional time series cleaning, HMM has more application space than the single-dimensional cleaning algorithm, because of the correlation between the dimensions.

C. Binomial Sampling

Shawn R. Jeffery et al. \cite{67} propose an adaptive method for cleaning RFID data, which exploits techniques based on sampling and smoothing theory to improve the quality of RFID data. Tag transition detection: tag transition detection refers to

| Reference | Method |
|-----------|--------|
| \cite{39}–\cite{42} | ODs |
| \cite{39}, \cite{109} | Extend ODs |
| \cite{43} | DCs |
| \cite{44} | SDs |
| \cite{102} | FDs |
| \cite{103}, \cite{104} | CFDs |
| \cite{45} | Speed Constraints |
| \cite{46} | Variance Constraints |
| \cite{47} | Similarity Rule Constraints |
| \cite{48} | Learning Individual Models |
the fact that when the position of the tag changes, the cleaning result should reflect the fact that the tag leaves. Let’s introduce a few RFID-related concepts.

(1) Interrogation cycles: the reader’s question-and-answer process for the tag is the basic unit of the reader’s detection tag.

(2) Reader read cycle (epoch, 0.2 sec - 0.25 sec): a collection of multiple interrogation cycles.

Based on the above concept, the following definition: \( W_i \) is smooth window of tag \( i \) and is composed of \( \omega_t \) epoch, \( S_i \) is the window that tag \( i \) is actually detected in the \( W_i \) window, \( Count_t \) indicates the number of inquiry cycles of \( t \), and \( R \) is the corresponding number of \( t \) epoch tag \( i \). For a given time window, suppose the probability that the tag \( i \) may be read in each epoch is \( p_i = \frac{R}{Count_t} \), and the Statistical Smoothing for Unreliable RFID Data (SMURF) \[67\] algorithm treats each epoch’s reading of the tag as a Bernoulli experiment with probability \( p_i \). Therefore, \( p_i \) conforms to the binomial distribution \( B(\omega_t, p_i) \), \( p_i,avg \) is the average read rate in \( S_i \).

Using the model based on the Bernoulli experiment to observe the tag \( i \), if the average reading rate of the tags in \( \omega_t \) epoch is \((1 - p_i,avg)^{\omega_t}\). To ensure the dynamic nature of the tag the size of the sliding window \( W_i \) needs to be satisfied as shown in equation (14).

\[
||S_i| - \omega_i p_i,avg| > 2\sqrt{\omega_i p_i,avg (1 - p_i,avg)}
\] (14)

The SMURF algorithm first sets the initial window size to 1, and then dynamically adjusts the window length based on the actual situation of the read. If the current window meets the integrity requirement \[67\], the SMURF algorithm will detect the status of the tag. When the detection result indicates that the tag status changes, SMURF will adjust the current window length to 1/2 of the original window to react to the tag’s transition. If the calculated window size that satisfies the integrity constraint is greater than the current window size, the algorithm linearly increases the current window size by 2 steps and outputs the point data in the current window. If it is detected that the label does not move, the algorithm outputs the current window midpoint as the output point, and then continues to slide an epoch for the next processing.

SMURF algorithm is widely used to clean RFID data, and many studies \[65\], \[66\] improve it. Leema, A et al. \[65\] study the effect of tag movement speed on data removal results and H Xu et al. \[66\] consider the impact of data redundancy on setting up sliding windows.

D. Spatio-Temporal Probabilistic Model

Besides data cleaning, M. Milani et al. \[68\] propose Spatio-Temporal Probabilistic Model (STPM), this method learns more detailed data patterns from historical data, and then cleans the current data. STPM not only gives joint probability distributions that are updated on the data set at different times, but also distinguishes association updates from association values. STPM based on Dynamic Probabilistic Relational Models (DRPMs), so we need to state DRPMs model first. The DRPMs is a graph model used to represent the relationship between dynamic data sets, its models based on the dependency relationship between attributes, and generally uses conditional probability distribution to calculate the probability of each attribute value in a given parent node value and forms a relationship chain. For instance, when we need to estimate the data at time \( T \), we can only use the data before time \( T \) to infer, namely, the current state depends only on the previous state, which is similar to the Markov Model. STPM extends DRPMs to model update pattern between different time data, and captures spatial and temporal update patterns by modeling updates events to provide update relationships of possible existence, finally detect and repair data.

E. Others

Firstly, we summarize the methods described above in Table (VI). In fact, Bayesian prediction model is a technique based on Bayesian statistics. The Bayesian prediction model utilizes model information, data information, and prior information, so the prediction effect is good, there this model is widely used, including in the field of time series data cleaning. Wang et al. \[56\] establish a cost model for Bayesian analysis which is used to analyze errors in the data. Bergman et al. \[70\] consider the user’s participation and use the user’s feedback on the query results to clean the data. Mayfield et al. \[72\] propose a more complex relationship-dependent network (RDN \[71\]) model to model the probability relationships between attributes. The difference between RDN and traditional relational dependencies (such as Bayesian networks \[57\]) is that RDNs can contain ring structures. The method iteratively cleans the data set.
and observes the change in the probability distribution of the data set before and after each wash. When the probability distribution of the data set converges, the cleaning process is aborted. Zhou et al. [110] argue a technique for accelerating the learning of Gaussian models via using GPU. The article believes that in the case of excessive data, it is not necessary to use all the data to learn the model. Also, the author provides a method of automatic tuning. In order to clean and repair fuel level data, Tian et al. [111] propose a modified Gaussian mixture model (GMM) based on the synchronous iteration method, which uses the particle swarm optimization algorithm and the steepest descent algorithm to optimize the parameters of GMM and uses linear interpolation-based algorithm to correct data errors. Shumway et al. [69] use the EM [112] method, which uses the particle swarm optimization algorithm combined with the spatial state model [27], [35] to predict and smooth the time series.

V. TIME SERIES ANOMALY DETECTION

Gupta et al. [14] investigate the anomaly detection methods for time series data: for a given time series data, there may be two types of outliers, namely single-point anomalies and subsequence anomalies (continuous anomalies). In this section, we first discuss the detection methods of abnormal points and abnormal sequences, next introduce the application of Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm in data cleaning, and then review the abnormal detection methods related to machine learning.

A. Abnormal point detection

For single-point anomalies, the most common idea is to use predictive models for detection. That is, the predicted value of the established model and the observed value for each data point is compared, and if the difference between the two values is greater than a certain threshold, the observed value is considered to be an abnormal value. Specifically, Basu et al. [113] select all data points with timestamps \( t - k \) to \( t + k \) with the timestamp \( t \) as the center point, and the median of these data points is considered to be the predicted value of data points with timestamp \( t \) value. Hill et al. [21] first cluster the data points and take the average of the clusters as the predicted value of the point. The AR model and the ARX model are widely used for anomaly detection in various fields, such as economics, social surveys [3], [4], and so on. The ARX model takes advantage of manually labeled information, so it is more accurate than the AR model when cleaning data. The ARIMA model [114] represents a type of time series model consisting of AR and MA mentioned above, which can be used for data cleaning of non-stationary time series. Kontaki et al. [79] propose continuous monitoring of distance-based outliers over data streams. One of the most widely used definitions is the one based on distance as shown in Figure 5. Here \( k = 4 \), \( q \) is the normal point and \( p \) is the abnormal point.

B. Abnormal sequence detection

Different studies have different definitions of subsequence anomalies. Keogh et al. [75] proposed that a subsequence anomaly, that is, a subsequence has the largest distance from its nearest non-overlapping match. With this definition, the simplest calculation method is to calculate the distance between each subsequence with length \( n \) and other subsequences. Of course, the time complexity of this calculation method is very high. In the later studies, Keogh et al. [115] propose a heuristic algorithm by reordering candidate subsequences and Wei et al. [116] argue an acceleration algorithm using local sensitive hash values. In calculating the distance, the Euclidean distance is usually used, and Keogh [117] further proposes a method using the compression-based similarity measure as the distance function. As shown in Figure 6 the data is divided into multiple sub-sequences that overlap each other. First, calculate the abnormal score of each window, and then calculate the abnormal score (AS) of the whole test sequence according to the abnormal score (AS) of each window. Window-based techniques can better locate anomalies compared to direct output of the entire time series as outliers. There are two main types of methods based on this technique. One is to maintain a normal database [118], [119], and then compare the test sequence with the sequence in the normal database to determine whether it is abnormal; the other is to build an anomalous database [76], [77] and then compare the test sequence with the sequence in the database to detect if it is anomalous. [120] found that the length of the error is unknown and they use grammar induction to aid anomaly detection without any prior knowledge.

TABLE VI
SUMMARY OF STATISTICS

| Reference | Method               |
|-----------|----------------------|
| 51–53, 73, 128 | Maximum Likelihood   |
| 54, 55       | Markov Model         |
| 58–60         | HMM                  |
| 61–63         | SMURF                |
| 64            | STPM                 |
| 65–67         | Bayesian             |
| 56, 57        | GMM                  |
| 70–72         | EM                   |

Fig. 5. An Example Abnormal Point Detection
C. Density-Based Spatial Clustering of Applications with Noise

The DBSCAN algorithm is a clustering method based on density-reachable relationship, which divides the region with sufficient density into clusters and finds clusters of arbitrary shape in the spatial database with noise and defines the cluster as the largest set of points connected by density. Then the algorithm defines the cluster according to the set density threshold as the basis for dividing the cluster, that is, when the threshold is satisfied, it can be considered as a cluster.

The principle of DBSCAN algorithm: (1) DBSCAN searches for clusters by checking the $Eps$ neighborhood of each point in the data set. If the $Eps$ neighborhood of point $p$ contains more points than $MinPts$, create a cluster with $p$ as the core object; (2) Then, DBSCAN iteratively aggregates objects that are directly reachable from these core objects. This process may involve the consolidation of some density-reachable clusters; (3) When no new points are added to any cluster, the process ends.

Where $MinPts$ is the minimum number of neighbor points that a given point becomes the core object in the neighborhood, $Eps$ is the neighborhood radius. For instance, $Eps$ is 0.5 and $MinPts$ is 3, for a given data set, the effect of clustering is as shown in Figure 7. Some noise points can be repaired and clustered into classes adjacent to them. Recent research [122] has shown that after repairing erroneous data. They also perform cleaning experiments on GPS data based on DBSCAN, the accuracy of clustering on spatial data can be improved. But this method cannot solve continuous errors and needs further improvement.

D. Generative Adversarial Networks

With the rapid development of machine learning technology, more and more problems are solved using machine learning. Dan Li et al. [80] use the GANs network to effectively detect anomalies in time series data. GANs trains two models at the same time, which are the generation model for capturing data distribution and the discriminant model for discriminating whether the data are real data or pseudo data as shown in Figure 8.

Given a random variable with a probability of uniform distribution as input, we want to generate a probability distribution of the output as "dog probability distribution". The philosophy of Generative Matching Networks (GMNs), which idea is to train the generative network by directly comparing the generated distribution with the true distribution, is to optimize the network by repeating the following steps:

1. Generate some evenly distributed input;
2. Let these inputs go through the network and collect the generated output;
3. Compare the true "dog probability distribution" with the generated "dog probability distribution" based on the available samples (e.g. calculate the MMD distance between the real dog image sample and the generated image sample);
4. Use backpropagation and gradient descent to compute the errors and update the weights. The purpose of this process is to minimize the loss of the generation model and discriminant.

Dan Li et al. [80] use GANs to detect abnormalities in time series and a natural idea is to use GANs network to repair missing values of time series data. Perhaps more machine learning algorithms are waiting for the cleaning of time series error values. A simple idea is to treat the detected anomaly data as missing data and then repair it. Y. Sun et al. [81] first analyze the similarity between parking space data and parking data, and then use Recurrent GANs to generate parking data as repair data, which provide a new idea for solving the problem of time series data repair. C. Fang et al. [82] propose FuelNet which is based on Convolutional Neural Networks (CNNs) and GANs. FuelNet is used to repair the inconsistent and impute the incomplete fuel consumption rates over time.

E. Long Short-Term Memory

Since Recurrent Neural Network (RNN) also has the problem of gradient disappearance, it is difficult to process long-sequence data. F. A. Gers et al. [123] improve RNN and got the RNN special case Long Short-Term Memory (LSTM), which can avoid the disappearance of the regular RNN gradient. It has been widely used in industry and [83]–[85] use LSTM to perform anomaly detection on time series data.
The left picture is a simple RNN structure diagram, and the right picture is a simple LSTM structure diagram in Figure 9, where given function as shown in equation (15).

\[ F : h, y = f(h, x) \]  

In equation (15), \( x^t \) is the input data in the current state, \( h^{t-1} \) (hidden state) indicates the input of the previous node received, \( y^t \) is the output in the current state and \( h^t \) is the output passed to the next node. As can be seen from the Figure 9, the output \( h^t \) is related to the values of \( x^t \) and \( h^{t-1} \). \( y^t \) is often used to invest in a linear layer (mainly for dimension mapping), and then use softmax to classify the required data. As shown in Figure 9, RNN has only one delivery state \( h^t \), LSTM also has a delivery status \( c^t \) (cell state). There are three main stages within LSTM:

1. Forgotten phase. The forgetting phase is mainly to forget the input that is passed in from the previous node. A simple principle is: forget the unimportant, remember the important one. More specifically, \( z^t \) is calculated as a forgotten gate, which is used to control the previous state \( c^{t-1} \), and then decide whether to retain the data or forget it.

2. Selective memory phase. At this stage, the input is selectively memorized. Mainly to remember the input \( x \), the more important the data needs to be more reserved.

3. Output phase. This phase determines which outputs would be treated as current states. Similar to the normal RNN, the output \( y^t \) is often also obtained by \( h^t \) change.

Filonov, Pavel et al. [83] and Pankaj Malhotra et al. [124] provide recurrent neural networks by providing network time series data. The recurrent neural network network understands what the normal expected network activity is. When an unfamiliar activity from the network is provided to a trained network, it can distinguish whether the activity is expected or invaded.

**F. Summary and Discussion**

In addition to the methods described above, we also summarize some common methods in Table VII. As shown in Table VII, Xing et al. [125] show that the cleaned sequence can improve the accuracy of time series classification. Diao et al. [73] design LOF [126] based online anomaly detection and cleaning algorithm. Zhang et al. [127] propose an iterative minimum cleaning algorithm based on the timing correlation of error time series in continuous errors and keep the principle of minimum modification in data cleaning. The algorithm is effective in cleaning continuous errors in time series data. Qu et al. [69] first use cluster-based methods for anomaly detection and then use exponentially weighted averaging for data repair, which is used to clean power data in a distributed environment. R. Corizzo et al. [74] use detect anomalous geographic data by distance-based method, and then use Gradient-boosted tree (GBT) to repair the anomalous data. Charu C et al. [78] proposed a solution for distributed storage and query for large-scale streaming sensor data, and they examined the problem of historical storage and diagnosis of massive numbers of simultaneous streams. We can conclude that anomaly detection algorithms play an important role in time series data cleaning. It is also becoming more and more important to design anomaly detection algorithms for time series repair, and we discuss future directions in Section VII.

**VI. TOOLS AND EVALUATION CRITERIA**

In this section, we first give an overview of tools to clean time series and then summarize evaluation criteria related to time series cleaning methods.

### A. Tools

There are many tools or systems for data cleaning, but they are not effective on time series cleaning problems. In Table VIII we investigate some tools that might be used for time series cleaning because they [128]–[130], [132] are originally used to solve traditional database cleaning problems. Ding et al. [131] present Cleanits, which is an industrial time series cleaning system and implements an integrated cleaning strategy for detecting and repairing in industrial time series. Cleanits provides a user-friendly interface so users can use results and logging visualization over every cleaning process. Besides, the algorithm design of Cleanits also considers the characteristics of industrial time series and domain knowledge. The ASPA proposed by Rong et al. [133] violates the principle of minimum modification and distort the data, which is not suitable for being used widely. EDCleaner proposed by J. Wang et al. [134] is designed for social network data, detection and cleaning are performed through the characteristics of statistical data fields. Y. Huang et al. [135] propose PACAS which is a framework for data cleaning between service providers and customers. R. Huang et al. [136] present TsOutlier, a new framework for detecting outliers with explanations over IoT data. TsOutlier uses multiple algorithms to detect anomalies in time series data, and supports both batch and streaming processing. EGADS [137] Offers two classes of algorithms for detecting outliers: Plug-in methods and Decomposition-based methods, which is designed for automatic anomaly detection of large-scale time series data.

| Table VII | SUMMARY OF DETECTION |
|-----------|-----------------------|
| Reference | Method |
| [73], [74] | Distance-based |
| [73], [74] | Maintain a Normal Database |
| [73], [79] | Build an Anomalous Database |
| [95] | Clustering |
| [80], [82] | GANs |
| [83]–[85] | LSTM |
or systems, and we discuss further in Future Directions in Section VII-B.

### B. Evaluation criteria

The Root Mean Square (RMS) error [67] is used to evaluate the effectiveness of the cleaning algorithm. Let $x$ denotes the sequence consisting of the true values of the time series, $\bar{x}$ denotes the sequence consisting of the observations after the error is added, and $\hat{x}$ denotes the sequence consisting of the repaired values after the cleaning. Here the RMS error [67] is represented as shown in equation (16).

$$\Delta(x, \hat{x}) = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i - \hat{x}_i)^2} \quad (16)$$

The equation (16) measures the distance between the true value and the cleaned value. The smaller the RMS error, the better the cleaning effect.

Other criteria include error distance between incorrect data and correct data, repaired distance between erroneous data and cleaned results (as shown in equation (17) referring to the minimum modification principle in data repairing).

$$\Delta(\bar{x}, \hat{x}) = \sum_{i=1}^{n} |\bar{x}_i - \hat{x}_i| \quad (17)$$

Dasu et al. [138] propose a statistical distortion method to evaluate the quality of cleaning methods. The proposed method directly observes the numerical distribution in the data set and evaluates the quality according to the variation of the distribution caused by different cleaning methods.

### VII. CONCLUSION AND FUTURE DIRECTIONS

In this paper, we review four types of time series cleaning algorithms, cleaning tools or systems and related research on evaluation criteria. Next, we summarize the full text in Section VII-A and list some advice of future directions in Section VII-B.

#### A. Conclusion

With the development of technology, people gradually realize the value contained in the data. Owing to companies want to derive valuable knowledge from these data, and data analysis has played an increasingly important role in finance, healthcare, natural sciences, and industry. Time series data, as an important data type, is widely found in industrial manufacturing. For instance, a wind power enterprise analyzes sensor data, which are located throughout the wind turbine, to determine whether the fan is in a normal state; transport companies also want to optimize vehicle fleet travel by analyzing vehicle GPS information. However, due to external environmental interference, sensor accuracy, and other issues, time series data often contain many errors that can interfere with subsequent data analysis and cause unpredictable effects.

#### B. Future Directions

As mentioned above, data is an intangible asset and advanced technology helps to fully exploit the potential value of data. Thereby, time series data cleaning methods provide very important technical support for the discovery of these values in processing time series error data. Next we list some advice of future directions based on (15).

The error type illustrates handbook of time series data. At present, data scientists have a very detailed analysis of the errors in the traditional relational database. However, there is still much work to be further studied in the analysis of time series data error types. For instance, this paper roughly divides the types of time series errors into three types, namely...

---

### TABLE VIII

**SOME EXAMPLES OF TOOLS OR SYSTEMS**

| Method      | Detail                                                                 |
|-------------|------------------------------------------------------------------------|
| PIClean [128] | Based on statistics Produce probabilistic errors and probabilistic fixes using low-rank approximation, which implicitly discovers and uses relationships between columns of a dataset for cleaning. |
| HoloClean [129] | Based on statistics Learn the probability model and then select the final data cleaning plan based on the probability distribution. |
| ActiveClean [130] | Based on statistics Allow for progressive and iterative cleaning in statistical modeling problems while preserving convergence guarantees. |
| Cleanits [131] | Anomaly detection Develop reliable data cleaning algorithms by considering features of both industrial time series and domain knowledge. |
| MLClean [132] | Anomaly detection The combination of data cleaning technology and machine learning methods is designed to generate unbiased cleaning data, which is used to train accurate models. |
| ASAP [133] | Smoothing based Develop a new analytics operator called ASAP that automatically smooths streaming time series by adaptively optimizing the trade-off between noise reduction and trend retention. |
| EDCleaner [134] | Based on statistics For social network data, detection and cleaning are performed through the characteristics of statistical data fields. |
| PACAS [135] | Based on statistics Design a framework for data cleaning between service providers and customers. |
| TsOutlier [136] | Anomaly detection Use multiple algorithms to detect anomalies in time series data, and support both batch and streaming processing. |
| EGADS [137] | Anomaly detection Offer two classes of algorithms for detecting outliers: Plug-in methods and Decomposition-based methods, which is designed for automatic anomaly detection of large-scale time series data. |
single point big errors, single point small errors and continuous errors. In fact, in continuous errors, there are also a lot of meticulous types of errors, such as additive errors, innovational errors [13] or missing errors [139]. How to systematically analyze these error types and form time series data error type illustrated handbook is very important. The clear error type helps to develop targeted cleaning algorithms to solve the problem of “GIGO (Garbage in, garbage out.)” that exists in the current field.

The design of time series data cleaning algorithm. Each chapter of this paper reviews some time series error cleaning algorithms, but further optimizations are possible. The existing methods are mostly for a single-dimensional time series (even the GPS data exists two dimensions’ information), but each dimension is cleaned separately during cleaning [45], [55], [127]. To further improve the practicability of the algorithm, it is imperative to consider the cleaning of multidimensional time series. Besides, with the development of machine learning technology, more technical learning techniques should be considered for data cleaning algorithms, which may lead to better cleaning results because of the mathematical support behind them.

The implementation of time series cleaning tool. At present, the mainstream data cleaning tools in the industry are still aimed at relational databases, and these tools are not ideal for processing time series data. As time series data cleaning problems become more serious, how to use the fast-developing distributed technology, high performance computing technology and stream processing technology to implement time series cleaning tools (including research tools and commercial tools) and apply them to real-world scenarios such as industry is also the key work of the next stage.

The algorithm design of time series anomaly detection. In real-world scenarios, efficient anomaly detection algorithms play an irreplaceable role in time series repair. It is difficult to judge the difference between the error value and the true value, so it is necessary to specifically design a time series anomaly detection algorithm that can be applied to an industrial scene. It is worth noting that more research is needed on how to perform anomaly detection, cleaning, and analysis in the case of weak domain knowledge or less labeled data.

Design of data cleaning algorithms for specific application scenarios. With the application of various technologies in the industry, the application scenarios are becoming more and more clear. The requirements for cleaning algorithms in different application scenarios have different focuses. For instance, rather than cleaning errors, another application is to directly answer queries over the possible repairs [140]. Moreover, rather than quantitative values, cleaning qualitative events [141]. [142] is also highly demanded. Finally, the data stored in the Blockchain network [101] are generally structured data, with the development of Blockchain technology, the design of data cleaning algorithms on Blockchain networks is also particularly important.

REFERENCES

[1] Robert H Shumway and David S Stoffer. *Time series analysis and its applications: with R examples*. Springer, 2017.
[2] James Douglas Hamilton. "Time series analysis." 2nd edition. Princeton University Press, 1994.
[3] Peter J Brockwell and Richard A Davis. *Introduction to time series and forecasting*. Springer, 2016.
[4] George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.
[5] Ruey S Tsay. *Outliers, level shifts, and variance changes in time series*. John Wiley & Sons, 2008.
[6] Joseph M Hellerstein. *Quantitative data cleaning for large databases*. *IEEE Trans. Knowl. Data Eng.*, 19(6):985–1006, 2007.
[7] Philip Bohannon, Michael Flaster, Wenfei Fan, and Jiannan Wang. *Data cleaning: Overview and emerging challenges*. In *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 2201–2206, 2016.
[8] Ruey S Tsay, Ihab F Ilyas, Sanjay Krishnan, and Jiannan Wang. *Data cleaning: Overview and emerging challenges*. In *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pages 2201–2206, 2016.
[9] Joseph M Hellerstein. *Quantitative data cleaning for large databases*. *IEEE Trans. Knowl. Data Eng.*, 19(6):985–1006, 2007.
[10] Ruey S Tsay. *Outliers, level shifts, and variance changes in time series*. John Wiley & Sons, 2008.
[11] Philip Bohannon, Michael Flaster, Wenfei Fan, and Rajeev Rastogi. *A cost-based model and effective heuristic for repairing constraints by value modification*. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, Baltimore, Maryland, USA, June 14-16, 2005*, pages 143–154, 2005.
[12] Philip Bohannon, Michael Flaster, Wenfei Fan, and Rajeev Rastogi. *A cost-based model and effective heuristic for repairing constraints by value modification*. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, Baltimore, Maryland, USA, June 14-16, 2005*, pages 143–154, 2005.
[13] Philip Bohannon, Michael Flaster, Wenfei Fan, and Rajeev Rastogi. *A cost-based model and effective heuristic for repairing constraints by value modification*. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, Baltimore, Maryland, USA, June 14-16, 2005*, pages 143–154, 2005.
Z Qu, Y Wang, Chong Wang, Nan Qu, and Jia Yan. A data cleaning model for electric power big data based on spark framework.

Eamonn J. Keogh, Jessica Lin, Sang-Hee Lee, and Helga Van Herle. Anomaly detection using only output cumulants. IEEE Trans. Acoustics, Speech, and Signal Processing, 38(7):1257–1265, 1990.

Gregory L. Plett. Extended kalman filtering for battery management systems of lifepb-basedhev battery packs: Part 3. state and parameter estimation. Journal of Power sources, 134(2):277–292, 2004.

Jef Wijsen. Reasoning about qualitative trends in databases. Inf. Syst., 23(7):463–487, 1998.

Jef Wijsen. Trends in databases: Reasoning and mining. IEEE Trans. Knowl. Data Eng., 13(5):426–438, 2001.

Roberto Casado-Vara, Fernando de la Prieta, Javier Prieto, and Juan M. Corchado. Blockchain framework for iot data quality via edge computing. In Proceedings of the 1st Workshop on Blockchain-enabled Networked Sensor Systems, BlockSys@SenSys 2018, Shenzhen, China, November 4, 2018, pages 19–24, 2018.

Catriel Beeri, Martin Dowd, Ronald Fagin, and Richard Statman. On the structure of Armstrong relations for functional dependencies. Journal of the ACM (JACM), 31(3):30–46, 1984.

Wenfei Fan, Floris Geerts, Xibei Jia, and Anasios Kementsisidis. Conditional functional dependencies for capturing data inconsistencies. ACM Transactions on Database Systems (TODS), 33(2):6, 2008.

Wenfei Fan, Floris Geerts, Jianzhong Li, and Ming Xiong. Discovering conditional functional dependencies. IEEE Trans. Knowl. Data Eng., 19(3):683–698, 2011.

Shaoux Song and Lei Chen. Differential dependencies: Reasoning and discovery. ACM Trans. Database Syst., 36(3):16:1–16:41, 2011.

Shaoux Song, Lei Chen, and Hong Cheng. Efficient determination of distance thresholds for differential dependencies. IEEE Trans. Knowl. Data Eng., 26(9):2170–2192, 2014.

Shaoux Song, Lei Chen, and Philip S. Yu. On data dependencies in dataspaces. In Proceedings of the 27th International Conference on Data Engineering, ICDE 2011, April 11-16, 2011, Hannover, Germany, pages 470–481, 2011.

Shaoux Song, Lei Chen, and Philip S. Yu. Comparable dependencies over heterogeneous data. VLDB J., 22(2):253–274, 2013.

Shaoux Song, Han Zhu, and Lei Chen. Probabilistic correlation-based similarity measure on text records. Inf. Sci., 289:8–24, 2014.

Jingbo Zhou and Anthony K. H. Tung. Smiler: A semi-lazy time series prediction system for sensors. In Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, May 31 - June 4, 2015, pages 1871–1886, 2015.

Daxin Tian, Yukai Zhu, Xuting Duan, Junjie Hu, Zhenguosheng, Min Chen, Jian Wang, and Yumpeng Wang. An effective fuel-level data cleaning and repairing method for vehicle monitor platform. IEEE Trans. Industrial Informatics, 15(1):410–422, 2019.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society: Series B (Methodological), 39(1):1–22, 1977.

International Journal of Database Theory and Application, 9(3):137–150, 2016.

Maksims Volkovs, Fei Chang, Jaroslav Szlichta, and Renee J. Miller. Continuous data cleaning. In IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014, pages 244–255, 2014.

Chad Park, Amanda G. Rutherford, Hoon Sohn, and Charles R. Farrar. An outlier analysis framework for impedance-based structural health monitoring. Journal of Sound and Vibration, 286(1-2):229–250, 2005.

George EP Box and David A. Pierce. Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. Journal of the American statistical Association, 65(332):1509–1526, 1970.

Sanjiv N Akouemo and Richard J. Povinelli. Data cleaning in time series using arx and ann models. IEEE Transactions on Power Systems, 32(5):3352–3359, 2017.

Peiyuan Chen, Troels Pedersen, Birgitta Bak-Jensen, and Zhe Chen. Arima-based time series model of stochastic wind power generation. IEEE transactions on power systems, 25(2):667–676, 2009.

Jie Ma and Jian-fu Teng. Predict chaotic time-series using unscetced kalman filter. In Proceedings of 2004 International Conference on Machine Learning and Cybernetics (IEEE Cat. No. 04EX626), volume 2, pages 687–690, IEEE, 2004.

Ih Chang, George C Tiao, and Chung Chen. Estimation of time series parameters in the presence of outliers. Technometrics, 30(2):193–204, 1988.

Ananthram Swami and Jerry M. Mendel. ARMA parameter estimation using only output cumulants. IEEE Trans. Acoustics, Speech, and Signal Processing, 38(7):1257–1265, 1990.

Gregory L. Plett. Extended kalman filtering for battery management systems of lifepb-basedhev battery packs: Part 3. state and parameter estimation. Journal of Power sources, 134(2):277–292, 2004.

Jef Wijsen. Reasoning about qualitative trends in databases. Inf. Syst., 23(7):463–487, 1998.

Jef Wijsen. Trends in databases: Reasoning and mining. IEEE Trans. Knowl. Data Eng., 13(5):426–438, 2001.
