

Interactive Data Editing of Time-Dependent Data in Visual Analysis

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Visual Computing

eingereicht von

Christian Möllinger

Matrikelnummer 0725979

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung
Betreuer/in: Ao.Univ.Prof. Univ.-Doz. Dipl.-Ing. Dr.techn. Eduard Gröller
Mitwirkung: Dipl.-Ing. Dr. Harald Piringer, VRVis Zentrum für Virtual Reality und Visualisierung
Forschungs-GmbH

Wien, 28.06.2014

(Unterschrift Verfasser/in)

(Unterschrift Betreuer/in)

Erklärung zur Verfassung der Arbeit

Christian Möllinger
Am Kaisermühlendamm 85, 1220 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit - einschließlich Tabellen, Karten und Abbildungen -, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

(Ort, Datum)

(Unterschrift Verfasser/in)

Abstract

In the so called information age, data is widely available. Sources include data collection as a byproduct (e.g., log files on a server, or as a more concrete example, movement profiles of smartphone users) and data generation for particular purposes (e.g., simulation runs, data gathered from sensors). To benefit from this huge amount of available data, the data must be analyzed and relevant information must be extracted. Visual Analytics has become an important approach to identify and extract relevant information from data, especially with big data sets.

However, data can also contain erroneous values for different reasons, e.g., caused by defect sensors. In data warehousing projects, transforming and edit the data into a usable state can account up for 80% of the cost and the development time.

This diploma thesis focuses on time-dependent data and presents an extension for the existing visual analytics framework VISPLORE, to support the user in the process of data editing. Using plausibility rules, the user can define data checks and imputation strategies. Three different overviews, a data-based overview, a group-based overview, and a rule-based overview provide insight into the structure of implausible data values and the defined plausibility rules. Implausible values can be imputed using the defined imputation strategies and existing visualization techniques are extended to enable the user getting an overview of the modified values. Real-world data is used to demonstrate two use-cases. Limitations of the provided overviews, e.g., scalability for a large number of plausibility rules, are discussed and ideas for future work are outlined.

Kurzfassung

Im so genannten Informationszeitalter sind Daten in großen Mengen verfügbar. Daten können implizit als Nebenprodukt entstehen (z.B. Logfiles auf einem Server, Bewegungsprofile von Handynutzern) oder explizit für bestimmte Zwecke generiert und gesammelt werden (z.B. Simulationsläufe oder Messwerte von Sensoren). Um von diesen Daten zu profitieren, ist es notwendig diese zu analysieren und die relevanten Informationen zu extrahieren. Visual Analytics ist ein Ansatz um diese Erkenntnisse aus großen Datenmengen ziehen zu können.

Allerdings können Daten auch Fehler enthalten, z.B. durch technische Defekte in den verwendeten Sensoren. Die notwendigen Schritte, um Daten in das korrekte Format zu bringen und insbesondere Datenfehler entsprechend zu berücksichtigen, kann laut Studien bis zu 80% der Zeit und Kosten von Data Warehouseing Projekten ausmachen.

Diese Diplomarbeit präsentiert eine Regel-basierte Erweiterung für das Visual Analytics Framework VISPLORE, um den Benutzer im Prozess der Datenbereinigung zu unterstützen. Mithilfe von Plausibilitätsregeln kann der Benutzer Bedingungen und Korrekturmethode für die Daten definieren. Drei verschiedene Übersichten, eine Daten-basierte Übersicht, eine Gruppen-basierte Übersicht und eine Regel-basierte Übersicht geben dem Benutzer Auskunft über die Struktur der erkannten Datenfehler und den definierten Plausibilitätsregeln. Fehlerhafte Daten können mithilfe der definierten Korrekturmethode editiert werden, und danach auch mit den Originalwerten verglichen werden. Zwei Fallbeispiele mit echten Daten demonstrieren die Benutzung und Funktion der implementierten Erweiterung. Limitierungen, wie z.B. die schlechte Skalierung der Übersichten für eine große Anzahl an Plausibilitätsregeln, werden diskutiert und Ideen für weitere Arbeiten ausgeführt.

Contents

1	Introduction	1
1.1	Visualization	2
1.2	Visual Analytics	4
1.3	Data Editing	4
1.4	Terminology	5
1.5	Scope	7
1.6	Structure of the Thesis	7
2	Problem Characterization and Goals	13
2.1	Problem Characterization	13
2.2	Goals	13
3	Related Work	17
3.1	Visual Analytics	17
3.2	Data Editing	19
3.3	Used Framework: Visplore	27
4	Plausibility Rules	35
4.1	Motivation	35
4.2	Rule Structure	35
4.3	Rules	36
4.4	Data Check	36
4.5	Imputation Strategies	38
4.6	Suggestion Strategies	38
4.7	Rule Set	39
5	Overview and Management	41
5.1	Motivation	41
5.2	Data-based Overview	42
5.3	Group-based Overview	45
5.4	Rule-based Overview	48
6	Imputation	55
6.1	Motivation	55
6.2	Applying Plausibility Rules	55
6.3	Created Meta Data	55
7	Integration into VISPLORE	57

7.1	Implemented Prototypes	57
7.2	Configuring Available Data Checks, Imputation and Suggestion Strategies	62
7.3	Creating Rules	63
7.4	The Rule Manager View	65
7.5	Inspecting Imputed Data	66
8	Case Study	73
8.1	The Data	73
8.2	Use Case 1: Content Specialist + Data Analyst	75
8.3	Use Case 2: Immediate Imputation of Implausible Values	81
8.4	Summary	81
9	Discussion and Future Work	93
9.1	Achieved Goals	93
9.2	Limitations	95
9.3	Future Work	95
10	Summary and Conclusion	97
	List of Figures	98
	List of Tables	101
	Bibliography	103

Introduction

In the last decades, technological progress changed the way how data is generated, used and processed. The ever-expanding performance of electronic hardware and computers (Moore's law [Moo98]), the ubiquitous use of information technology in many application areas and the availability of sensors or logging methods led to the situation that data is recorded and generated at an incredible rate. Examples for such recorded data are logged network traffic, banking transactions or emails, where at least 210 billions of emails are recorded per day [KKEM10]. An example for generated data would be simulation runs in research areas like engineering or climate research, which generate terabytes of results. These high volumes of available data provide an enormous potential for gaining knowledge and supporting decision-making.

To benefit from these high volumes of data, relevant information must be identified and extracted, which is clearly not a straight-forward task:

"Today, the acquisition of raw data is not longer the driving problem: It is the ability to identify methods and models, which can turn the data into reliable and provable knowledge."
[KAF⁺08]

Graphical representation or *visualization* is a technique to get a fast overview of data and recognize the relevant information quickly ("a picture is worth 10.000 words"). A short overview about visualization is given in Section 1.1. However, visualization by itself does not answer automatically all upcoming questions. Further data analysis can support the insights, but it also depends on the background of the data which parts are relevant and which are not. While visualizations were rather static and mostly used to present existing knowledge to an auditorium until the 20th century, the age of computers and the improvements in electronic data processing made it possible to provide interactive visualization. In the same time, automated data analysis using statistical methods and electronic data processing evolved. But a fully-automated analysis process has the problem that the background or context of the data, which is known to the user, has no influence on that analysis. The emerging research area *Visual Analytics* tries to combine interactive visualization with semi-automated data analysis, where the user takes an important part in the process and can control which direction the analysis is taking. Section 1.2 gives a detailed introduction into Visual Analytics.

But visualization as well as Visual Analytics or data analysis in general, rely on the quality of the available data. Although raw data can be acquired in high volumes for virtually every branch of industry or business, it must be considered that these data can also contain errors. For example, mechanical failures in recording or storage devices like sensors or hard-disks can lead to wrong or missing values. Human

generated input data (e.g., surveys) can contain wrong values due to typing errors or reading errors (e.g., 'O' or '0'). Already transformed or pre-processed data can contain errors due to erroneous transformation rules, software bugs or storage failures. There are many possible causes for such data errors, but in any case, such data errors may have serious influence on the analysis of the data. For example, significant outliers, missing values or other irregular sensor measurements may lead to wrong results for statistical evaluations which remain undetected. While analysis using statistical models may silently lead to wrong results, visual analysis can facilitate the recognition of these data errors.

Hence, the challenge to recognize and correct such data errors is a well-known problem in data analysis, and thus it is also a big research topic with many different approaches and solutions. Section 1.3 gives an introduction into data editing in general. The goal of this thesis is to use the strengths of Visual Analytics to provide an intuitive and interactive way to detect and handle data errors in time-dependent data. We will provide an extension to an existing visual analytics application, which targets the overall workflow of error handling. It assists the user in recognizing data quality issues and their occurrence patterns. Furthermore, it allows the user to impute data values within the visual analytics application. Imputed values are meant to be "correct" or at least "better" for a specific context and intended use. And finally, existing visualization techniques are extended to enable the user getting an overview of the modified values as well as getting a detailed comparison of imputed and original values.

1.1 Visualization

In general, the meaning of the word visualization has changed over the last decades. According to Ware [War00] and Keim et al. [KMS⁺08], visualization today means "a graphical representation of data or concepts" and has three major goals:

1. **Presentation**

To present facts and knowledge which are already known and to communicate them to other people in an understandable way.

2. **Confirmatory Analysis**

To confirm or reject existing hypotheses.

3. **Exploratory Analysis**

To search for new information like trends or structures, without having an initial hypothesis.

The former use of visualization were mostly static images like maps or charts, to bring something to mind. The main motivation was that some information could be easily understood by people. For example, maps describe geographic information which would need a long verbal or textual description, charts mostly describe some numerical information in an abstract way.

A popular and well documented exception [Gil58] for using visualization not only to present already known information, can be seen in Figure 1.1. In this case, visualization was used to gain new and a-priori unknown knowledge: In 1844 many people got contracted with a deadly disease in London and the actual cause was unanswered for a long time. At some point, all cases of death were sketched in a geographical map of a district, resulting in a visible cluster around a specific water pump. John Snow then discovered the contaminated water pump as the cause of a Cholera epidemic.

The development of electronic computers facilitates fast image generation compared to printed graphics, which are quite limited in this regard. Together with the rise of exploratory data analysis in the statistics research community [Tuk77], this resulted in a move from static images to interactive visualizations. While common types of visualizations like scatterplots or boxplots were originally developed by statistical data analysts [CM84], a new research area has emerged in the last two decades. This new research area concerns visualizations generated by computers and can be subdivided into *Scientific Visualization* and *Information Visualization*.

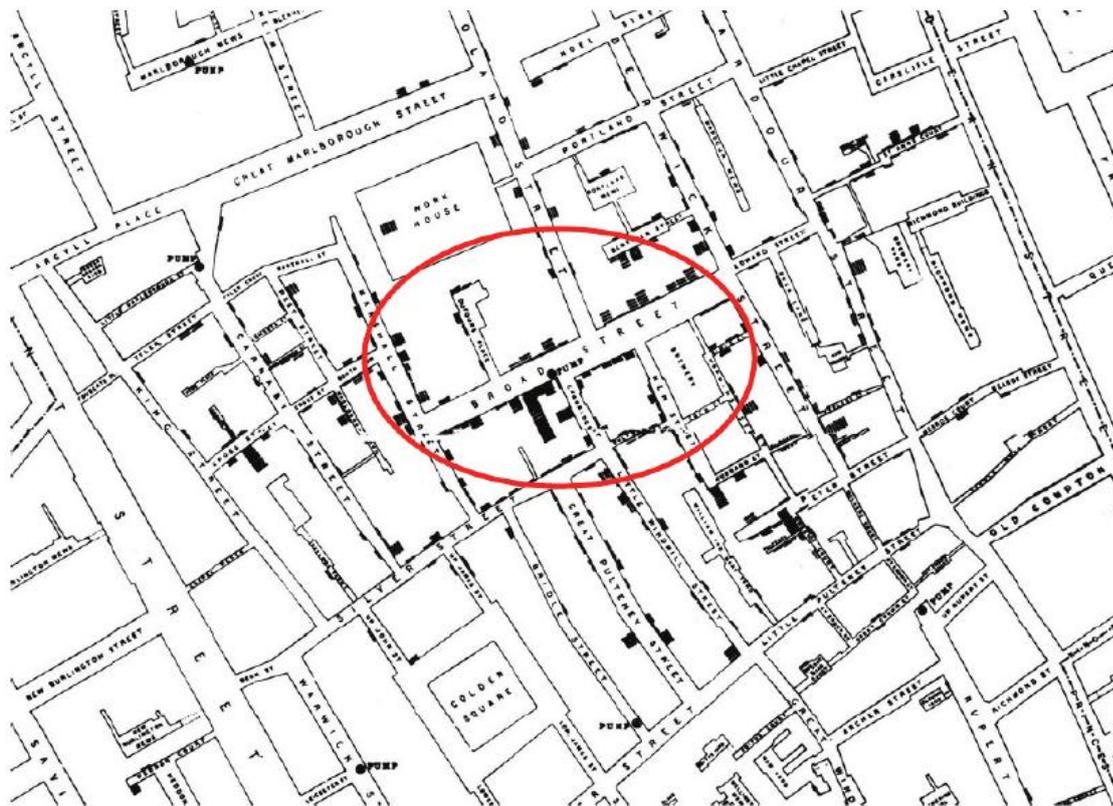


Figure 1.1: In 1844, many people suffered from a deadly disease. This visualization helped to discover a contaminated water pump as the cause for the deaths. The bars represent the number of deaths in geographic areas, and it can be seen that the deaths are clustered around the encircled pump [Gil58].

Scientific visualization deals with data which has an inherent spatial 2D or 3D mapping. Typically it is further divided in *Volume Visualization* and *Flow Visualization* or other areas, depending on the data context. Figure 1.2 shows an example for volume visualization, while Figure 1.3 shows an example for flow visualization.

Compared to scientific visualization, information visualization deals with data that does not have an inherent spatial context [HJ05], e.g., survey data. Information visualization uses abstractions or visual metaphors to represent the data respectively the interesting information. Figure 1.4 shows an interactive information visualization to compare multiple biological DNA trees as an example [BvLH⁺11].

According to Rhyne et al. [RTM⁺03] this discrimination based on the underlying data is not strict, instead it depends on the information which should be extracted. A taxonomy of visualization techniques formed on a higher level than the underlying data is given by Tory and Möller [TM04]. They proposed a model-based approach to discriminate visualizations, build on the design model of a visualization. The design model is the set of assumptions made by the algorithms about the data which should be visualized. It is more flexible than the data based discrimination and considers the user's conceptual model. Figure 1.5 shows this high level taxonomy based on examples.

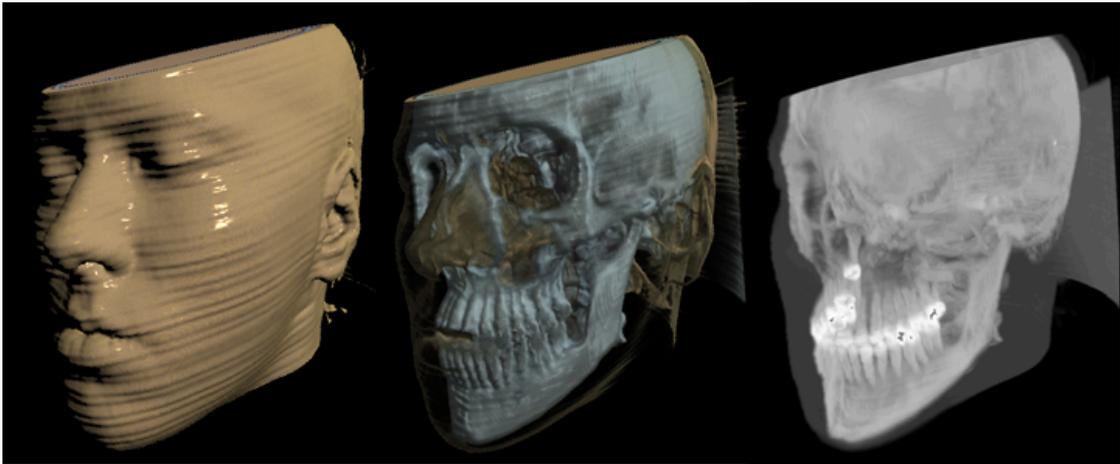


Figure 1.2: Volume rendering of a human head, which was scanned with magnetic resonance imaging. The three visualizations represent the same scan, but three different parameter settings for the visualization method. Screenshot taken from an own student project.

1.2 Visual Analytics

"Visual Analytics is the science of analytical reasoning facilitated by interactive visual interfaces." [TC05].

Visual Analytics (also called *Visual Analysis*, but *Visual Analysis* is also a registered trademark used by a company [vis]) is a young research topic established in 2005, which focuses on a tight integration of interactive visual and automated data analysis. It is an integrative discipline and combines various related research areas such as visualization, data mining, data management, data fusion, statistics and cognitive science. It faces the problem that today, huge amounts of data are generated all along. To gain profit from this data, relevant information has to be found, which either delivers new insights into a process or lets companies make decisions based on that information [KKEM10]. To extract such relevant information, the data must be analyzed. Electronic data processing can analyze huge amounts of data but on the other hand, human abilities provide general knowledge and perception to connect components and generate new knowledge, as shown in Figure 1.6.

The key advantage of *Visual Analytics* is the combination of automatic data processing and the strengths of human abilities. The user has the ultimate authority in directing the data analysis but he is strongly supported by interactive graphical visualizations and semi-automatic processes. Interactive visualization provides the necessary tools for the user to analyze data in his way. Figure 1.7 shows the process of visual data exploration as it can be done with visual analytics tools.

1.3 Data Editing

Data editing, also called *data cleaning*, *data scrubbing* or *data cleansing* means the process of detecting and correcting erroneous data [RD00][KCH⁺03][GGAM12]. There are many possible reasons for data to be erroneous. Data can have a wrong format on a detailed level (e.g., DD/MM/YY instead of MM/DD/YY for date information) or on a higher level (e.g., wrong database structure), values can be wrong or missing due to several causes (e.g. if the value "2" describes the age of a criminal, it can be assumed that it is wrong), they can contain duplicate entries, inconsistent entries or any other type of values that are wrong

and unusable. A detailed taxonomy of dirty data was published by Kim et al. [KCH⁺03] and refined for time-oriented data by Gschwandtner et al. [GGAM12].

The overall process of data editing, as described by Müller et al. [MF03] and Rahm et al. [RD00], consists of several steps:

- **Data auditing**

The first step is auditing the data to find the types of anomalies which occur in the data. The results of the auditing step support the specification of integrity constraints, which depend on the application domain and are specified by domain experts, and domain formats.

- **Workflow specification**

Recognition and elimination of anomalies is done by a sequence of operations on the data, which is called the data cleansing workflow. The workflow is specified using the results of the auditing step. One of the main challenges consists in the specification of a cleansing workflow which automatically eliminates all anomalies in the data.

- **Verification of the workflow**

The correctness and effectiveness of the workflow is tested and evaluated after the specification and before the execution of the workflow. While the verification is defined as separate step by Rahm et al., Müller et al. see the verification as an integral part of the workflow specification and thus not defined as a separate step in their publication.

- **Workflow execution**

After the specification and verification the cleansing workflow is executed. Difficult cases may require interaction with the expert, who has to decide whether data is erroneous or not and specify or select the correct modification from a set of solutions. This interaction is expensive and time consuming and often avoided by logging, so that the expert can inspect these cases manually after the execution of the workflow.

- **Post-processing**

After executing the cleansing workflow, the results are inspected by the expert to verify the correctness of the specified operations. This can also result in a new cycle in the data cleansing process.

Based on this process definition, the goal of this thesis is to design and develop interactive tools to support the process of data editing. Specifically, the focus is on facilitating the detection of data errors with visualization and providing possibilities to define rules how data errors should be corrected or rather imputed.

1.4 Terminology

In this section, some important terms which are used in this diploma thesis are defined.

1.4.1 Data Attribute / Data Entry / Data Row / Data Column

Data can be structured in different ways. This section describes the data structure and the terms used in this thesis.

In Table 1.1, an example data set is shown. The data is structured as follows:

- **Data attribute**

A data set can have one or more data attributes, (e.g., *Time*, *Temperature*, *Gas consumption*, *Wind speed*) which describe the given data values. Furthermore, data values can be categorized

<i>Index</i>	<i>Time</i>	<i>Temperature</i>	<i>Gas consumption</i>	<i>Wind speed</i>
1	2013-10-01 00:00	13	MISSING	10
2	2013-10-01 01:00	12	1100	3
3	2013-10-01 02:00	12	1000	6
4	2013-10-01 03:00	11	1000	9
5	2013-10-01 04:00	11	1100	3

Table 1.1: An example data set showing the structure of the data as it is used in this thesis.

into different types. We differentiate between categorical data and numerical data, and each data column contains either categorical or numerical data.

- **Data entry**

A data entry equates to a single value and can also be an empty value, which then would be interpreted as MISSING. Data entries can be either missing due to problems with the data quality or because some data is not available for well grounded reasons, for example if an additional measurement sensor was installed in the year 2012, but the data includes all measurement values since 2010.

- **Data row**

A data row is a tuple of data entries, which describe the given attributes, e.g., (2013-10-01 00:00, 13, MISSING, 10).

- **Data column**

A data column is a vector containing all data entries specifying a data attribute, e.g., for *Temperature* that would be (13, 12, 12, 11, 11).

- **Meta data**

Data columns can also have additional meta information which is important to interpret and visualize the data correctly. Examples would be used data units (e.g., °C for *Temperature* and km/h for *Wind speed*).

1.4.2 Time-Dependent Data

Section 1.4.1 describes multivariate data in a general sense. Many parts of the concepts described in this thesis can be applied on general multivariate data. Justified by the concrete use case power economy, the focus is set on time-dependent data as a special case of multivariate data, which means that one of the data attributes describes discrete timestamps. An example of such data would be measurement values of temperature sensors which were written to a log file together with the corresponding timestamps. Such data can be formally described as

$$D = \{(t_1, v_1), (t_2, v_2), \dots, (t_n, v_n)\} \quad (1.1)$$

$$v_i = (v_i^1, v_i^2, \dots, v_i^m) \quad \text{for } i \in 1..n \quad (1.2)$$

where t_i is a discrete timestamp and v_i describes the corresponding values, e.g., the temperature measurements of m different sensors. In other words and according to Section 1.4.1, t is a data column containing all timestamps, v^x is a data column containing the corresponding values for $x \in 1..m$ and (t_k, v_k) describes a data row with $k \in 0..n$. m describes the number of columns without the column containing the

timestamps and n describes the number of rows. Additionally, we expect the data in a regular time raster, so that the following holds:

$$t_{i+1} - t_i = c \quad (1.3)$$

where c is a constant value. A detailed taxonomy of time-oriented data was published by Müller et. al [MS03].

1.5 Scope

This thesis describes a framework to support the workflow of data editing in the context of a Visual Analytics application. The background and state of the art considering Visual Analytics as well as the background and the state of the art of data editing is summarized and reviewed in order to develop such a framework.

The focus of this work is the overall workflow. Although data editing and the detection of data errors in general has a strong connection to statistical data analysis, the academic use and improvement of statistical methods is not part of this thesis. The used algorithms to detect data errors or to calculate new values for values which should be edited are only prototypes and placeholders.

1.6 Structure of the Thesis

This thesis is structured in the following chapters:

- Chapter 2 gives an overview of the problem, the expected workflows and a list of the goals of this thesis.
- In Chapter 3, the related work is briefly discussed, together with an overview of the used framework and techniques.
- Chapter 4 presents the concept of plausibility rules as used in this thesis,
- followed by an approach to provide overview and management in Chapter 5.
- Chapter 6 describes the process of data imputation as provided by our approach.
- Afterwards, Chapter 7 describes the integration into VISPLORE.
- An evaluation of our approach with real-world data is done in Chapter 8,
- and Chapter 9 provides a discussion about the results and experiences with our method, together with thoughts on future work,
- Finally, Chapter 10 summarizes the thesis.

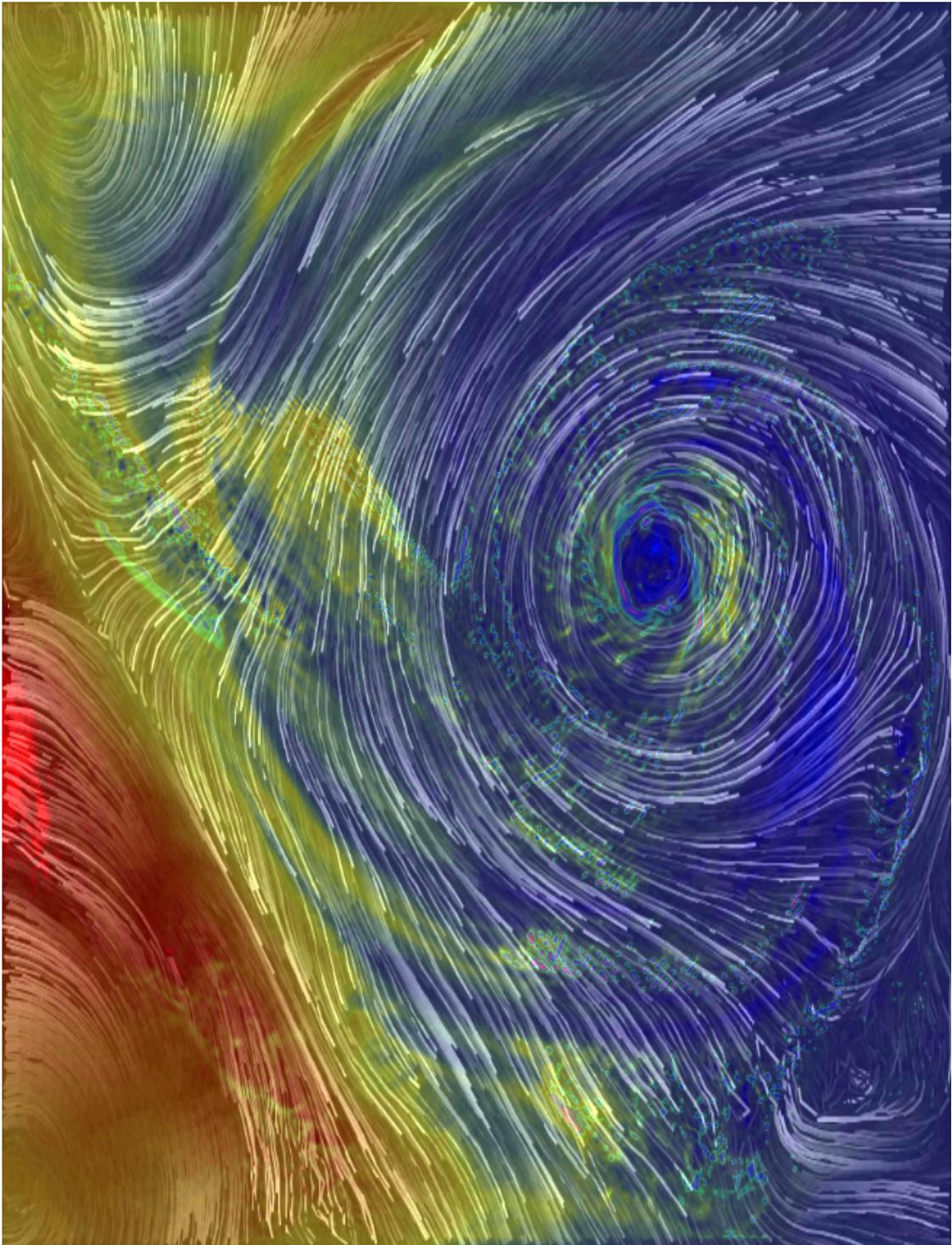


Figure 1.3: An example for flow visualization, showing a specific weather situation. The data set contains a 2D grid, each point in the grid has the direction and the velocity of the wind as additional information. Streamlines [JL97] are used to visualize the flow of the wind. Screenshot taken from an own student project.

		<i>Display Attributes</i>		
		<i>Given</i>	<i>Constrained</i>	<i>Chosen</i>
<i>Continuous</i>	Images (e.g., medical)		Distortions of given / continuous ideas (e.g., flattened medical structures, 2D geographic maps, fish-eye lens views)	Continuous (high-dimensional) mathematical functions
	Fluid / gas flow, pressure distributions			Continuous time-varying data, when time is mapped to a spatial dimension
	Molecular structures (distributions of mass, charge, etc.)		Arrangement of numeric variable values	Regression analyses
	Globe – distribution data (e.g., elevation levels)			
<i>Discrete</i>	Classified data / images (e.g., segmented medical images)		Distortions of given / discrete ideas (e.g., 2D geographic maps, fish-eye lens views)	Discrete time-varying data, when time is mapped to a spatial dimension
	Air traffic positions			Arbitrary entity-relationship data (e.g., file structures)
	Molecular structures (exact positions of components)		Arrangement of ordinal or numeric variable values	Arbitrary multi-dimensional data (e.g., employment statistics)
	Globe – discrete entity data (e.g., city locations)			

Figure 1.5: High-level visualization taxonomy, illustrated by examples. Design models are classified based on whether they are discrete or continuous and by how much the algorithm designer chooses display attributes (spatialization, timing, colour, and transparency). Examples show different constraints on spatialization [TM04].

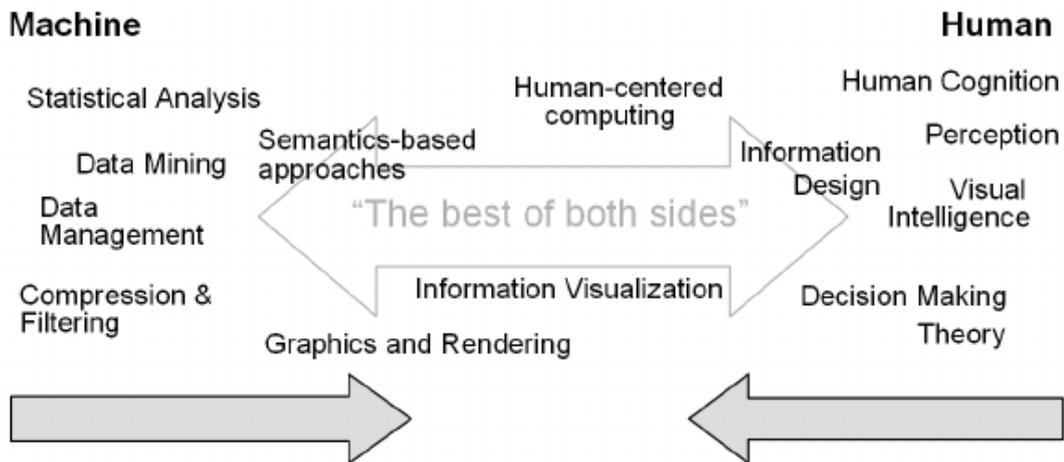


Figure 1.6: The abilities of a machine differ from the human abilities. Visual Analytics tries to combine the best of both [KAF⁺08].

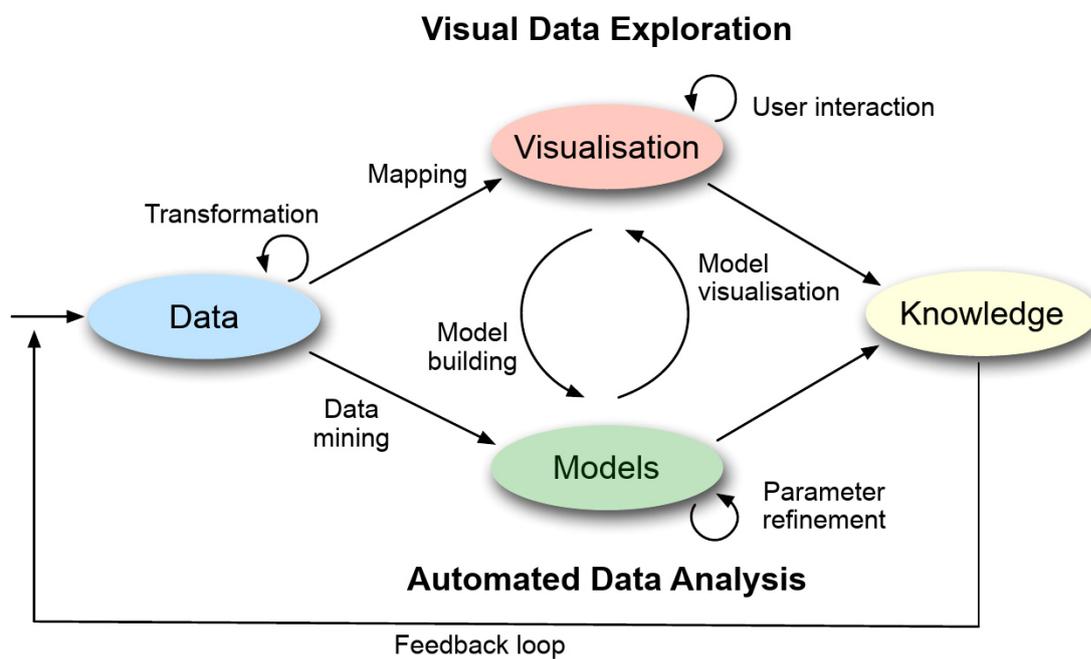


Figure 1.7: The Visual Data Exploration process. It is characterised by interaction between data, visualisation, models of the data, and the user to discover knowledge [KKEM10].

Problem Characterization and Goals

This chapter gives a characterization of the problem statement together with a basic workflow, which we have identified as a potential use case when dealing with data errors and different application areas. The problem characterization together with the workflow are used to identify the corresponding goals, which are listed in an own section afterwards.

2.1 Problem Characterization

The decision whether values are data errors respectively not plausible depends heavily on the context of the data. A value of -99 may be plausible if it describes the temperature in $^{\circ}C$ of a frozen bacterial culture, but it seems not plausible if it describes the measured temperature in a daily trend, where all other values are around $20^{\circ}C$.

On the other hand, additional background information may affect the evaluation of values, whether they are plausible or not. A popular example is the daily trend of realized flights of an european airport. In april 2010, there arise negative outliers which seem to be not plausible. With the additional background information that the volcano Eyjafjallajökull erupted in these days and the air traffic was grounded for security reasons, the values seem to be plausible. So one of the main problems with automatic error detection is, that it can not be done in a general way, it requires context and background information to success.

We have chosen an approach based on plausibility rules to determine the plausibility of values. Such plausibility rules can be defined by the data analyst or, in a preprocessing step, by a content specialist. Using such rules, some data errors can be detected, while others still remain unnoticed or are evaluated as false-positives. Providing an overview of the current set of plausibility rules and their evaluation results, combined with available views in a visual analytics framework should help the user to refine these rules or to detect wrong rules or uncovered data errors. Finally, rules should not only be used to detect data errors, they should also be used to impute such errors.

2.2 Goals

This section gives an overview of the goals of this thesis, which are based on the problem characterization given in the previous section. To summarize, the goal of this thesis was to develop an extension for the visual analytics framework VISPLORE (see Section 3.3), which assists the user in recognizing data

quality issues, defining plausibility rules, getting a rule-based overview about the data-quality, semi-automatically correcting the quality issues and enabling an analysis of the data which is aware of the performed imputation.

2.2.1 Plausibility Rules

The user should be able to define plausibility rules, which can be used to detect data quality issues. For example, if data describes the consumption of electricity, it can be assumed that the values must not be negative, so that the user could define such a rule and each negative value would be automatically interpreted as an implausible value.

These rules depend on the concrete context of the data. Therefore, a content specialist should be able to create a set of rules for their specific type of data, which can be used by the data analyst who uses the same type of data but a different concrete dataset. Specifically, the goals can be summarized as follows:

- **Provide a user interface to define a set of plausibility rules**
The user interface should support the user in creating, editing and deleting these rules and it should be integrated into VISPLORE.
- **Store the plausibility rules optionally independent from the actual data**
The user should be able to export the current set of rules, or import another set of rules. This allows a content specialist to create templates, which can be imported and refined by the data analysts.
- **Provide an overview of the plausibility rules**
The user should get a quick overview of the defined rules, the different types of rules and which data attributes are described or rather tested by the rules.
- **Provide detailed information of violated plausibility rules**
The user should be able to determinate which rules were violated by which data entries, which rules have a similar pattern of violations, or which rules are violated by an abnormal percentage of data values.

2.2.2 Recognition of Data Quality Issues

The user should be able to detect data quality issues according to the plausibility rules and get an overview and insight of the structure of these issues. If the appearance of issues correlates with time, the user should be able to visually detect the pattern. The user should also be able to mark detected implausible values, so that those values are not only "marked" in the users head but also in a semantic way in VISPLORE. Specifically, the goals can be summarized as follows:

- **Provide an overview of the data quality according to the plausibility rules**
The user should be able to get an overview of the data quality, to find out the structures and characteristics of erroneous values. The overview should answer questions like: Are there any clusters of data errors, is there a correlation with the time axis or with any other data attribute?
- **Mark the erroneous values**
There should be an additional semantic layer, which defines for each value if it is erroneous or not. In current visual analytic frameworks, this information cannot be stored in a semantic layer, this information is only stored in the users head.

2.2.3 Imputation of Data Errors

The user should not only be able to detect and mark data errors. Instead, imputation of erroneous values should be possible within the same framework. To achieve this, plausibility rules do not only contain the constraint which is applied on values to evaluate them, they also define a specific imputation strategy. If such a data imputation is applied, the reason and used imputation method should be traceable for the corresponding data entries. Specifically, the goals can be summarized as follows:

- **Execute the plausibility rules to apply the defined imputation methods**
The plausibility rules should be executable by the user, which means that all values which are rated as not plausible by the executed rule, are modified according to the defined imputation strategy.
- **Provide methods to trigger error imputation manually**
Even if many data quality issues are detected by plausibility rules, some of the issues may still remain unnoticed using the defined rules. However, the user should be able to impute data values instantly using given imputation strategies, even if no plausibility rules rated the values as not plausible.
- **Traceability**
If data values are edited by executing rules, it should be traceable why a value was changed and which imputation strategy is responsible for the current value.

2.2.4 Integration into Existing Visual Analytics Framework VISPLORE

The result of this diploma thesis should not be a new standalone application. Instead it should result in a plugin for the existing visual analytics framework VISPLORE and should use the advantages and features provided by this tool. Furthermore, the plugin should be integrated in a way that it feels like a part of VISPLORE, both technically and regarding the user experience. Specifically, the goals can be summarized as follows:

- **A new view should provide an overview of erroneous data and the defined plausibility rules**
The overview of data errors and defined plausibility rules should be provided by a new view (see Section 3.3). Additionally, the view should provide detailed information about the rules and should support to create, edit, apply and delete rules.
- **Use or adapt existing views whenever possible**
The existing views and features of the framework should be used for all cases where it is possible, instead of creating new views or implementing own techniques which could be also done with the existing techniques.
- **Mark values as erroneous by selection**
The user should be able to use the available methods in VISPLORE to select some values and define them manually as implausible.
- **Define a plausibility rule by marking values as erroneous**
It should be possible to automatically infer a specific rule type and appropriate parameterization based on a selection of data and a hint for the cause of the implausibility.

Related Work

This chapter summarizes work related to this thesis. The discussion is structured into fundamental concepts of Visual Analytics, Data Editing and an introduction to the visual analytics application VISPLORE, which is used as a framework for this thesis.

3.1 Visual Analytics

Visual Analytics is a rather young research topic. It emerged in the last decade to combine the strengths of automatic data analysis with the human abilities as described in Section 1.2. This section summarizes fundamental concepts and state-of-the-art techniques which are relevant for this thesis.

3.1.1 Multiple Coordinated Views

The analysis of large data is a complex and challenging task. Different visualization techniques exist to support the exploration of data. As particular visualization techniques have different advantages and disadvantages, one specific view may be great for one specific task or subset of the data but inappropriate for another task or subset of the data.

Multiple Views is the general name to describe any instance where data is represented in multiple windows [Rob07][HHSW09]. It usually implies that different representations are used in the subsequent windows and that operations on the views are coordinated, thus Roberts suggests the longer term *Multiple Coordinated Views*:

"Multiple Coordinated Views is an exploratory visualization technique that enables users to explore their data. In fact, the overall premise for the technique is that users understand their data better if they interact with the presented information and view it through different representations." [Rob07]

According to Wang Baldonado et al. [WBWK00], multiple views should be considered when

- there is a diversity of attributes, models, user profiles, levels of abstractions, or genres
- different views bring out correlations and/or disparities
- multiple views would create manageable chunks and provide insight into the interaction among different dimensions

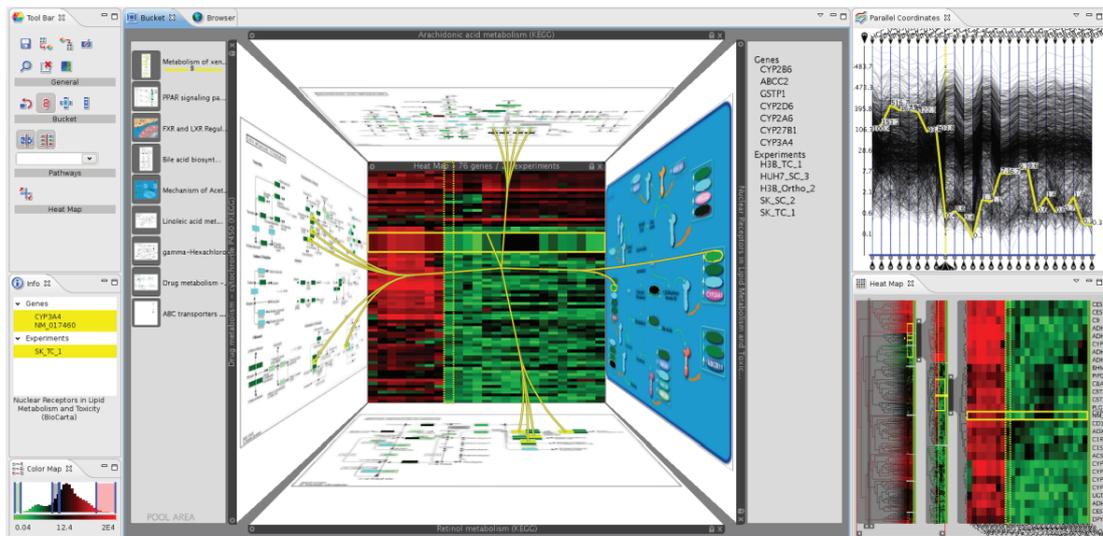


Figure 3.1: The visual analytics framework Caleydo [LSKS10] layouts multiple views in a 2.5D setting and connects related information using VisLinks [CC07] across views.

However, Wang Baldonado et al. advised that the computational overhead, display space overhead and necessary learning should be kept in mind and thus, multiple views should be used minimally.

For general multiple-view-environments, different strategies exist to manage and layout diverse types of views. Views can be organized using the provided widgets (windows, dynamic splitters) provided by the operating system. Also more complex organizations are possible, e.g., Caleydo places the views on the five visible inner faces of a cube (see Figure 3.1), so that the centered view has the focus and the surrounding views provide context information. Additionally, this 2.5D approach provides an intuitive way of drawing connections between related visual representations in different views.

As a conclusion, multiple coordinated views are a key technique in Visual Analytics. It is an approach that addresses different types of complexity in the data, which would be inappropriate or impossible to represent in a single view. Therefore, they are a key technique with respect to visual scalability [WBWK00].

3.1.2 Overview and Detail

Visualizing large data sets typically brings up the problem that it is not possible to show all details of the dataset in one image. One possibility to tackle this problem is filtering, which has the problem that it eliminates or ignores parts of the data. To overcome this limitation, approaches which provide focus on an area of interest and contextual information details were developed. These approaches can be grouped into different classes [KHG03][CKB08]:

- **Overview + Detail**

The overview information is shown spatially separated from the detail information. For example, such a technique was used in previous versions of Google Maps, as it can be seen in Figure 3.2.

- **Zooming**

Using this approach, users can interactively zoom into the areas of interest, but with temporal separation between the different levels of resolution.



Figure 3.2: Overview + Detail shown by the example of Google Maps. It shows a large detail map and a small overview map spatially separated (yellow arrow) [goo].

- **Distortion-oriented Focus+Context**

A more integrative approach is distortion-oriented focus+context, which provides a detailed area (focus) together with a contextual area within one visualization. Therefore, the visualization is distorted geometrically, so that the detailed area gets more image space while the image space for the contextual area is reduced. As an example, VISPLORE provides interactive zooming in some views, which implements this kind of distortion (see Figure 3.3). Another specialized example is the TableLens [RC94], which provides a different degree of detail for the focus and the context (see Figure 3.4).

- **In-place Focus+Context**

Different parts of the data (focus and context) are highlighted using different visual cues. These can be different colors or even different values of blur. For example, Kosara et al. [KMH01] use blur to mark parts as contextual information, while the sharp part is in focus (see Figure 3.5).

To conclude, there are different approaches and finding the best approach depends on the concrete application area and is task-dependent. However, it has been stated as a suitable and effective approach to achieve visual scalability by discriminating details from contextual information.

3.2 Data Editing

"[...] This process of improving the data quality by detecting and correcting errors encompasses a variety of procedures, both manual and automatic, that are referred to as statistical data editing." [dWPS11]

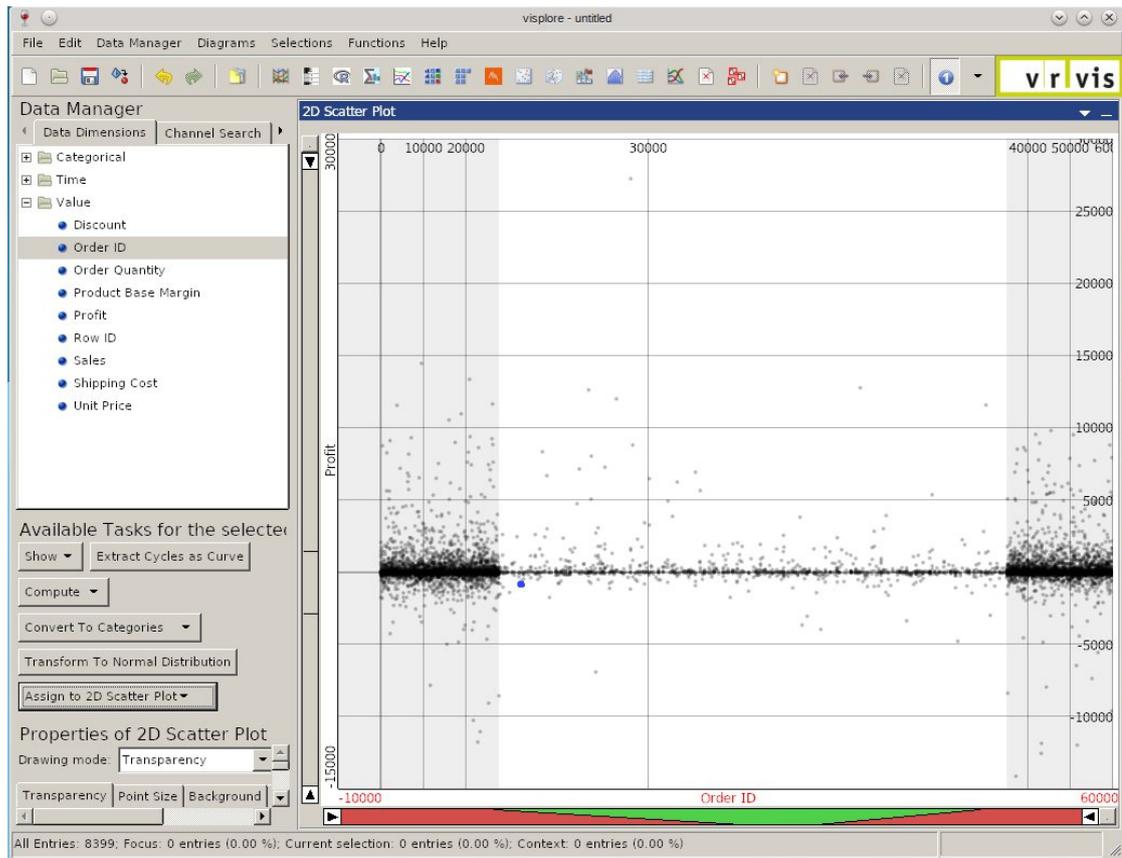


Figure 3.3: VISPLORE provides a data range slider with zooming ability. When the user interactively zooms in, the slider represents the zoomed area using green color and the contextual area using red color. Note the corresponding geometrically distorted visualization.

In general, data editing means the process of detecting and correcting data quality issues. It is a frequently researched topic and this section will give a short overview of recent and prior publications. Different approaches for data editing were proposed and while some approaches are meant to be general techniques, some of the publications are focused on rather specific application areas. This is caused by different error characteristics, for example, data errors in weather data are categorized and edited in a different way than data errors in survey data.

Rahm et al. [RD00] gave an overview of different data quality problems and data editing approaches. They focused their work on data warehouses [CD97][LVVJ03], which have to deal with many different data types and data sources. They provided a classification of data quality problems (see Figure 3.6) and distinguished between single-source / multi-source problems and schema-related / instance-related problems. While not shown in the picture, quality issues related to single-source data can also occur in multi-source cases, and schema-level problems are also reflected in the instances. As already mentioned in Section 1.3, Rahm et al. also described the overall process of data cleaning with several phases (*Data Analysis / Auditing, Workflow Definition, Verification, Transformation / Workflow Execution*) in their work.

Winkler and William [Win99] gave an overview of statistical data editing and current research problems back then. More recently, T. De Waal et al. [dWPS11] gave an overview of statistical data editing,

Order Priority	Region	Order Date	Ship Date	Discount	Profit
Low	Prarie	2011-01-12 00:00:00	2010-01-01 00:00:00	7.000000e-002	-160
Medium	West	2011-01-12 00:00:00	2011-02-12 00:00:00	0.02	605
Not Specified	West	2011-01-12 00:00:00	2011-02-12 00:00:00	0.02	294
High	Northwest Territories	2011-02-01 00:00:00	2011-03-01 00:00:00	0.01	-46
High	West	2011-02-01 00:00:00	2011-03-01 00:00:00	0	115
Critical	West	2011-02-01 00:00:00	2011-04-01 00:00:00	0.08	103
Critical	West	2011-02-01 00:00:00	2011-04-01 00:00:00	0.02	-98
Critical	West	2011-02-01 00:00:00	2011-02-01 00:00:00	0	539
High	West	2011-02-01 00:00:00	2011-04-01 00:00:00	0.01	1

Figure 3.4: The Table Lens provides different degrees of detail for the focus and the context. Screenshot of VISPLORE.

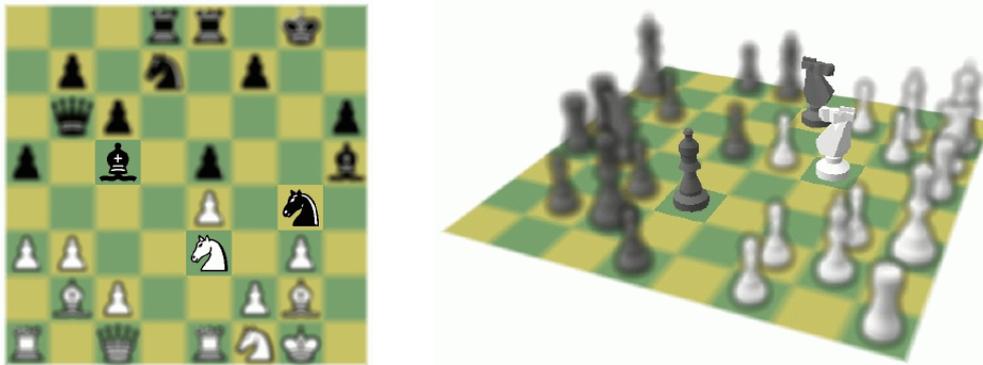


Figure 3.5: In-place focus+context as used by Kosara et al. [KMH01] to mark context and focus.

including anomaly detection, deductive correction, automatic or selective editing and imputation and an overview of practical applications. A selective editing approach using statistical indicators was done by J. Hoogland [HN06]. Detected data errors are either edited automatically using an application called SLICE 1 or manually by the user, depending on the impact on publication totals. To determinate this impact and the editing method, statistical indicators are used.

Another interesting approach to detect outliers was published by Hodge and Austin [HA04]. They were using statistical methods combined with machine learning algorithms, to classify data records automatically as outliers.

3.2.1 Anomaly Detection

"Anomalies are patterns in data that do not conform to a well defined notion of normal behavior." [CBK09]

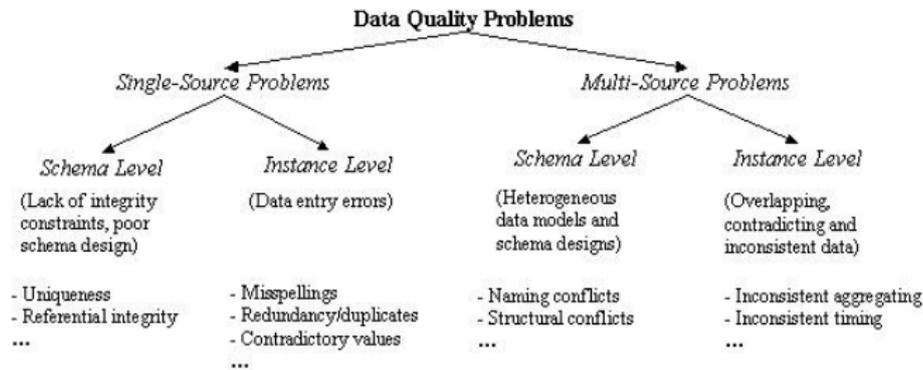


Figure 3.6: Classification of data quality problems from different data sources [RD00].

Another interesting work related to data editing was published by Chandola et al. [CBK09], giving a structured overview of data anomalies and the different techniques and their underlying approach to detect such anomalies. Detecting data anomalies is tightly coupled with data editing and thus, this section will shortly summarize the structured overview by Chandola et al. They categorized anomalies in three groups: *Point Anomalies*, *Contextual Anomalies*, and *Collective Anomalies*.

A value is termed as a point anomaly, if an individual data value can be considered as anomalous with respect to the rest of data. Figure 3.7 shows an example. It can be seen that some data values lie outside specific data ranges, thus they can be considered as anomalous.

Data values which are considered anomalous with respect to their context, but not anomalous if only the value itself is evaluated, are termed as contextual anomalies. To allow this contextual evaluation, the notion of a context is expected to be available in the data set. One common example are time series, which include the timestamp as contextual information. See Figure 3.8 for an illustration of a contextual anomaly.

A sequence of data values which is anomalous with respect to the entire data set while the values themselves are not anomalous, is called an collective anomaly. Figure 3.9 shows a human electrocardiogram output as an example. The low value by itself is not an anomaly, but the same low value for an abnormally long time can be stated as collective anomaly.

Moreover, they also presented a detailed list of anomaly detection techniques. The following techniques were listed:

- **Classification-based anomaly detection techniques**

These techniques use classification to learn a model from a set of labeled data instances (*training* phase). This model is then used to classify given data values into one of the classes (*testing* phase).

Neural networks, Bayesian networks, support vector machines or rule systems are examples for classification-based anomaly detection techniques. Classification-based techniques have the advantage that they can make use of powerful algorithms to distinguish between different classes. Furthermore, the testing phase is fast because it has only to be compared against the pre-computed model. However, they rely on the availability of accurate labels, which is often not possible. Additionally, these techniques cannot be used without modification if an anomaly score is wanted instead of a classification.

- **Nearest-neighbor-based anomaly detection techniques**

Nearest-neighbor-based anomaly detection techniques are based on the key assumption, that nor-

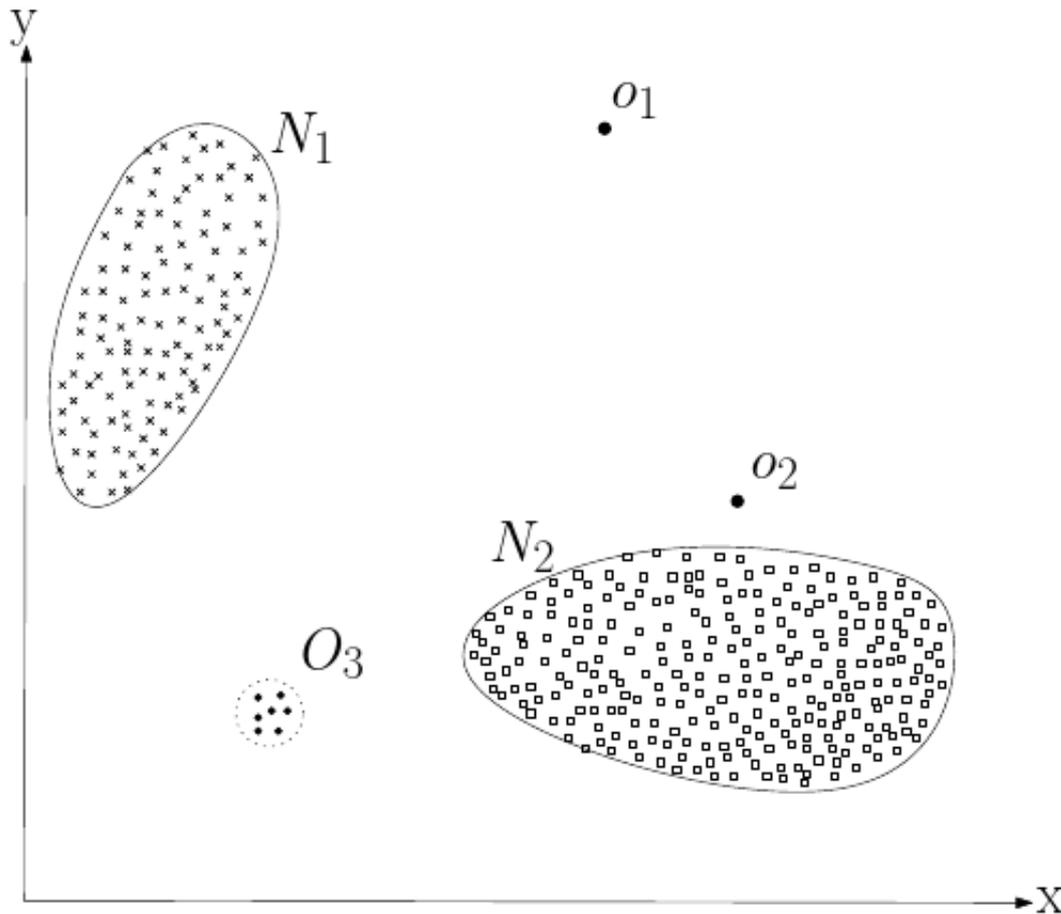


Figure 3.7: Point anomalies in a two-dimensional data set. The data has two normal regions N_1 and N_2 , since most observations lie in these two regions. Points o_1 , o_2 and points in region O_3 are anomalies [CBK09].

mal data values occur in dense neighborhoods, while anomalies occur far from their closest neighbors. These techniques can be broadly divided into two groups: Using the distance to k^{th} nearest neighbor or using the relative density as the anomaly score.

One of the key advantages of these techniques is that they are purely data driven and make no assumptions regarding the generative distribution of the data. However, if the data has normal instances that do not have enough close neighbors or if the data has clustered anomalies, the technique fails to label them correctly. Furthermore, the computational complexity is a significant challenge since it involves computing the distance of each data value towards many other values to determinate the nearest neighbors.

- **Clustering-based anomaly detection techniques**

Clustering based techniques group similar data values into clusters. The techniques itself can be grouped into three categories, which use different assumptions to rate data values as anomalous or as normal. The first category relies on the following assumption: *Normal data values belong*

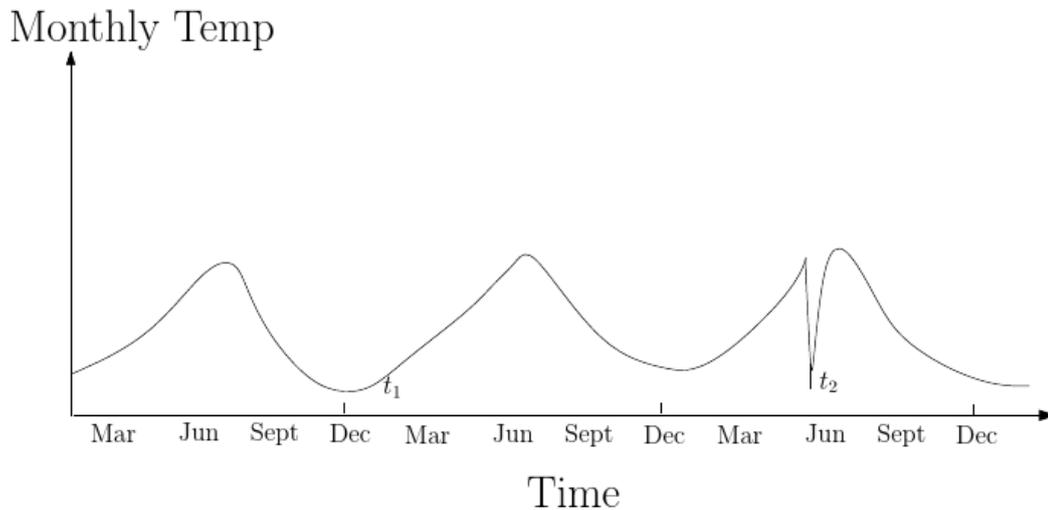


Figure 3.8: Showing a contextual anomaly. Note that the temperature at time t_1 is no anomaly while the same temperature at time t_2 must be considered an anomaly [CBK09].

to a cluster in the data, while anomalies do not belong to any cluster. These techniques have the disadvantage that they are not optimized to find anomalies, instead, they are optimized to find clusters. Hence, the second category of clustering based techniques relies on the following assumption: *Normal data values lie close to their closest cluster centroid while anomalies are far away from their closest cluster centroid.* But if anomalies in the data form clusters by themselves, they are still undetected. To address this issue, the third category of clustering based techniques relies on the following assumption: *Normal data values belong to large and dense clusters, while anomalies belong to either small or sparse clusters.*

Although the techniques seem similar to the neighbor based techniques, they have the advantage that the testing phase is much faster because values are only tested against a low number of clusters instead of all other data values.

- **Statistical anomaly detection techniques**

Statistical anomaly detection techniques use the underlying principle that an anomaly is an observation which is suspected of being partially or wholly irrelevant because it is not generated by the stochastic model assumed. Thus, they are based on the following key assumption: *Normal data instances occur in high probability regions of a stochastic model, while anomalies occur in the low probability regions of the stochastic models.* They fit a statistical model to the given data and then use a statistical inference test to decide if an unseen value belongs to this model or not. Statistical techniques have the advantage that, if the assumptions regarding the underlying data distribution hold true, statistically justifiable solutions are provided. Furthermore, the tested data values are not classified in "normal" and "anomaly", a score is calculated instead. However, it is necessary to rely on the assumption that the data is generated from a particular statistical distribution.

- **Information theoretic anomaly detection techniques**

Information theoretic anomaly detection techniques rely on the key assumption that *anomalies in the data induce irregularities in the information context of the data set.* They have the advantage that they do not make any assumptions about the underlying statistical distribution of the data and

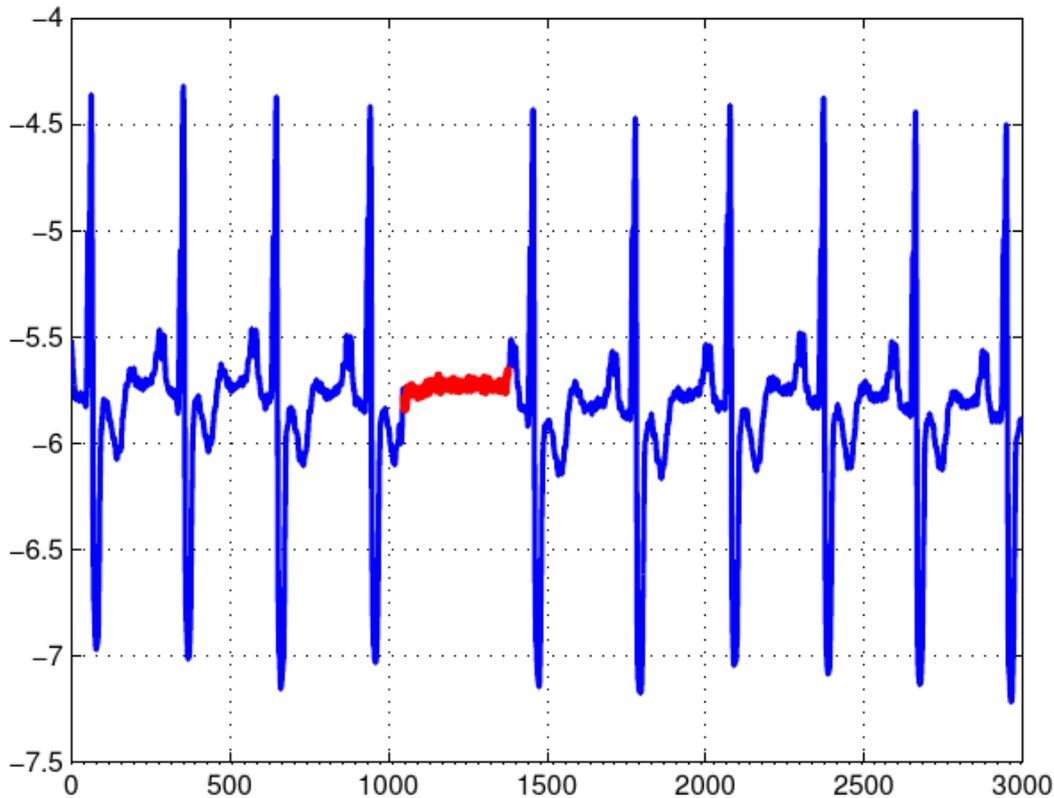


Figure 3.9: Collective anomaly corresponding to an *Atrial Premature Contraction* in a human electrocardiogram output [CBK09].

can operate in an unsupervised setting. However, the disadvantages are that the performance of such techniques is highly dependent on the choice of the information theoretic measure and often such measures can detect the presence of anomalies only when there are significantly large numbers of anomalies present.

- **Spectral anomaly detection techniques**

Spectral techniques try to determinate subspaces in which the anomalies can be identified easily. These techniques rely on the following key assumption: *Data can be embedded into a lower dimensional subspace in which normal instances and anomalies appear significantly different.* One famous technique is the *Principal Component Analysis (PCA)* [Jol02] for projecting data into a lower dimensional space. These techniques have the advantage that they can handle high dimensional data sets because they perform an automatic dimensionality reduction. Furthermore, they can be used in an unsupervised setting. However, they are only useful if the normal and anomalous values are separable in the lower dimensional embedding of the data, and they typically have high computational complexity.

3.2.2 Data Wrangling

"Data wrangling is the process of making data useful." [KHP⁺11]

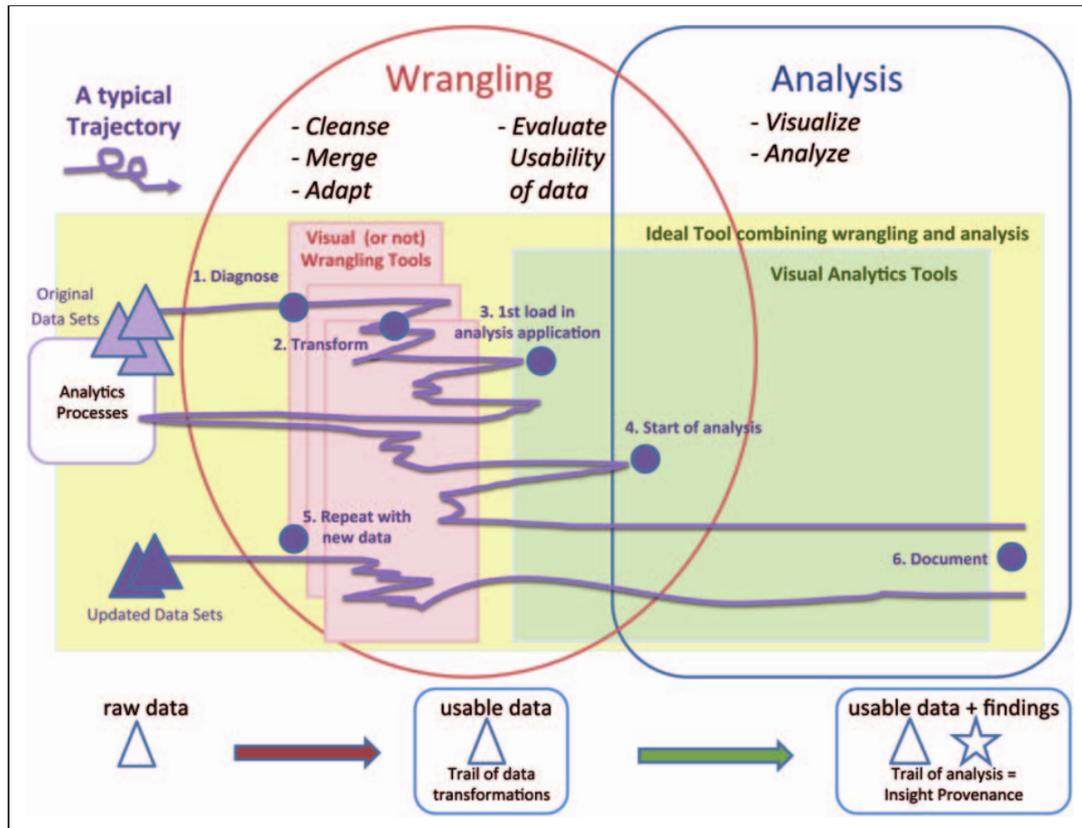


Figure 3.10: The iterative process of data wrangling and data analysis [KHP⁺11].

A related topic to data editing is data wrangling. Data wrangling describes the process of making data usable, to put it in a form that it can be analyzed. More generally, Kandel et al. [KHP⁺11] defined data wrangling as *a process of iterative data exploration and transformation that enables analysis*.

The process of data wrangling may also require a processing step of data editing to correct erroneous values. However, data wrangling is much more than data editing, it is the overall process to make data useful. Figure 3.10 illustrates this iterative process.

According to the definition of Kandel et al., data is *useable* if it can be processed and analyzed using analysis tools, it is *credible* if it suitably represents a phenomenon according to an analyst's assessment, and it is *useful* if it is useable, credible and responsive to one's inquiry.

3.2.3 Data Editing in Specific Application Areas

As already outlined, many publications considering data editing are focused on a specific application area. This section gives a short overview of these publications.

Van den Broeck et al. [VdBCEH05] published an overview of the process of data cleaning with focus on clinical epidemiological studies. They presented a three-stage process involving repeated cycles of screening, diagnosing and editing suspected data abnormalities. Fay and Robert [Fay99] used nearest neighbor imputation to edit demographic census data. A few years earlier, Bankier et al. [BHL97] proposed a "New Imputation Methodology (NIM)" to edit data of the 1996 Canadian Census. The NIM

is a two-step process which uses hot deck edit and imputation for the demographic variables. Hot deck imputation means that values are replaced by values which occur in the data.

Some publications are focused on application areas which handle time oriented data. Handling erroneous values in weather data was researched by Boissonnade et al. [BHW02]. Other related work was also done by Jin et al. [JSMR09], Xiao et al. [XGH06] and Galeano et al. [GPT06].

3.2.4 Interactive Data Editing

In the last decade, also interactive approaches and techniques for data editing were researched and published. Raman and Hellerstein [RH01] introduced an interactive data cleaning system called *Potter's Wheel*, which allows the user to clean the data by defining data transformations on a spreadsheet-like user interface. As soon as the cleaned data set is on the screen, the defined transformations are applied and the user can decide to undo previously defined transformations or add further actions.

Another interactive data cleaning application called *Wrangler* was proposed by Kandel et al. [KPHH11]. They use an underlying declarative transform language combined with a *graphical user interface (GUI)* where the user can define transform scripts for the data interactively. Applied transformations can be back-tracked by a history, which logs all executed transformations.

Kandel et al. have also developed a visual analytics tool to detect and analyze data quality issues, which is integrated with *Wrangler* and called *Profiler* [KPP⁺12]. It provides automatic view suggestions and scalable summary visualizations, which support brushing and linking to analyze detected anomalies. Compared to our approach, it does not rely on user-defined plausibility rules, instead, it uses data mining methods to automatically flag problematic data.

3.3 Used Framework: Visplore

3.3.1 Overview

As already mentioned in Section 2.2, the implementation of this diploma thesis has been done as a plugin for an existing visual analytics framework, called VISPLORE. VISPLORE is a software developed at the *VRVis Research Center*¹ and it is used as a framework for basic research projects as well as application-orientated research in cooperation with industrial project partners. This section will give a short overview of the key concepts, the strengths, major use cases and finally, some technical details of VISPLORE. Furthermore, it gives a short summary of two publications related to VISPLORE and this thesis: A multi-threading architecture with asynchronous event handling and the R integration.

Based on the background of the initial industrial project partner and the application field of engineering data, VISPLORE is tailored towards multivariate and high-dimensional data, which can be both categorical and continuous. Data columns are called *channels* and multiple channels can be grouped into *channel sets*. The framework is designed to handle large data sets, consisting of millions of entries and hundreds of dimensions. To visualize the data, VISPLORE supports more than 10 different visualization techniques (views), which have different advantages and disadvantages. Furthermore, VISPLORE supports four different data layers, which are described subsequently.

The following list briefly explains some of the provided views:

- **2D Scatterplot**

An old and well-known visualization technique for two-dimensional data, where each data record is aligned by two orthogonal axes and represented by a glyph [GTC01]. An example can be seen in Figure 3.11.

¹VRVis Zentrum für Virtual Reality und Visualisierung Forschungs-GmbH, Vienna

- **Parallel Coordinates**

Parallel Coordinates is a common technique to visualize multi-dimensional data. The data axes are drawn parallel and side-by-side, each data value is represented by a poly-line which has its vertices on the different axes [ID91]. An example can be seen in Figure 3.11.

- **Aggregate View**

Using the Aggregate View, the user can split the data into cells by assigning multiple categorical dimensions on the X and Y axis. Additionally, a continuous dimension and an optional aggregate (e.g., average, sum, min, max) function can be assigned to the X and Y axis, the color and size attribute. According to the concrete configuration, the view visualizes the results using bar charts, heatmaps, line graphs or scatterplots. See Figure 3.12 for an example.

- **Table Lens View**

The Table Lens [RC94] is a technique, which basically renders the data as a graphical table. Depending on the available vertical space for a horizontal row, values are either rendered as detailed textual information or visualized by horizontal bars. The available lens allows the user to zoom into the view and get more details, while other entries are mapped on less space. Figure 3.4 shows an exemplary Table Lens View.

- **Detail View**

The DetailView is a view which displays all entries within the current selection layer in a tabular format, where each data row represents a table row and each assigned data column is represented by a table column.

- **1D Rank By Feature View**

This view provides an overview of the assigned data dimensions by visualizing a small histogram combined with a box plot and offering a rich set of common statistical measures. Additionally, the view offers to rank the dimensions with respect to a selected statistical measure. Furthermore, the view includes a one-dimensional partition-based framework to build and validate regression models as published by Mühlbacher and Piringer [MP13].

- **2D Rank By Feature View**

This view is conceptually similar to the 1D rank by feature view, however, it provides a bivariate analysis rather than a univariate analysis. It provides statistical measures like the Pearson's correlation coefficient or Spearman's rank correlation coefficient for each pair of assigned data dimensions and displays a scatterplot matrix which contains each pair of assigned dimensions. The key idea of this view was published by Shneiderman et al. [SS04]. Similar to the 1D rank by feature view, this view also provides the proposed two-dimensional partition-based framework by Mühlbacher and Piringer [MP13].

The four different data layers which define arbitrary subsets of the whole data with respect to the data rows are as following:

- **Focus**

This layer contains all entries which are currently hovered by the user (e.g., points in the scatterplot or bars in the aggregation view).

- **Current Selection**

The current selection layer contains all data rows which match the current active query. A query can be defined by interactive brushing and combining the brushes using logical operations like AND ("refine"), OR ("extend") and SUBTRACT.

- **Context**

VISPLORE supports multiple queries using interactive brushing, but only one query can be used for the current selection layer at the same time. The context layer consists of data rows, which match one of the other queries (except the current active query). The current active query can be switched interactively in the user interface.

- **All Data Rows**

This layer represents all data rows which are loaded into the system.

If one or more layers are changed, the views are automatically updated. This enables an instant visual response and linked views. For illustration, in Figure 3.12, the user defines a query by interactively brushing in the scatterplot. Based on the resulting layer contents, the aggregation view visualizes the average profit separated by "in current selection" and "not in current selection".

VISPLORE is developed using the object orientated programming language *C++* [Str94] and uses *gtk+* [gtk] for the user interface and *OpenGL* [ope] for the realtime graphic rendering. It consists of a core system and a plugin interface, which is used to implement the different views and data connectors.

3.3.2 Multi Threading Architecture with Asynchronous Event Handling

For smooth and efficient exploration of unknown data sets, a tight feedback loop, that means a fast response in the visualization after a user interaction, is necessary. Data sets with several millions of entries, complicated calculations or demanding visualizations degrade the performance and have bad impact on that response time.

To provide visual feedback as fast as possible, VISPLORE uses a multi threading architecture as proposed by Piringer et al. [PTMB09], which helps to provide the following properties:

- low latency between interaction and visual feedback
- show as much visual feedback as possible as early as possible
- minimize the variation of the amount of shown details to provide a stable image

The key idea of this architecture is to subdivide the final render image into separate passes through the visualization pipeline (referred to as "layer"). The layers are processed by the visualization thread one after the other, while the event handling is done in a separate thread. Whenever a user interaction changes the visualization parameters, the visualization thread is stopped, affected layers are invalidated, and the visualization thread is restarted. Valid layers are re-used and only invalid layers are re-processed. Figure 3.13 gives an illustration of this technique.

3.3.3 Integration using R

Even though VISPLORE provides some statistical evaluations to analyze data, they are only a subset of available statistical methods. Furthermore, the statistical methods have to be implemented from scratch. The open source software R [Tea] is a statistical framework which provides an extendable scripting language, basic visualizations and libraries which implement many statistical methods and calculations. It is used in research communities and many publications are related to new statistical methods implemented in R (e.g., M. Templ [Tem08], Alfons et al. [ATF10], Templ et al. [THF11], Baddeley and Turner [BT05], Suzuki and Shimodaira [SS06]).

Kehrer et al. [KBFP12] published a general approach to combine the statistical framework R with visual analytics frameworks. VISPLORE provides a plugin which implements this approach and allows the on-demand access to all statistical methods and graphics provided by R. Figure 3.14 shows the key concept of the R integration. Part (a) is the iterative analysis workflow, which consists of the interactive

part in VISPLORE, and the computational part which is done in R. R scripts can access the current selections, and the scripts themselves are executed either on demand or whenever the selection information has changed. The results in R can either be transferred back to VISPLORE, or analyzed with the numerical and graphical abilities of R itself. In both cases, the results are immediately updated when the selection changes. In VISPLORE, two additional parts were added: an R object browser, which allows the inspection and configurable synchronization of objects in the R workspace, and an R console which provides the R command line interpreter.

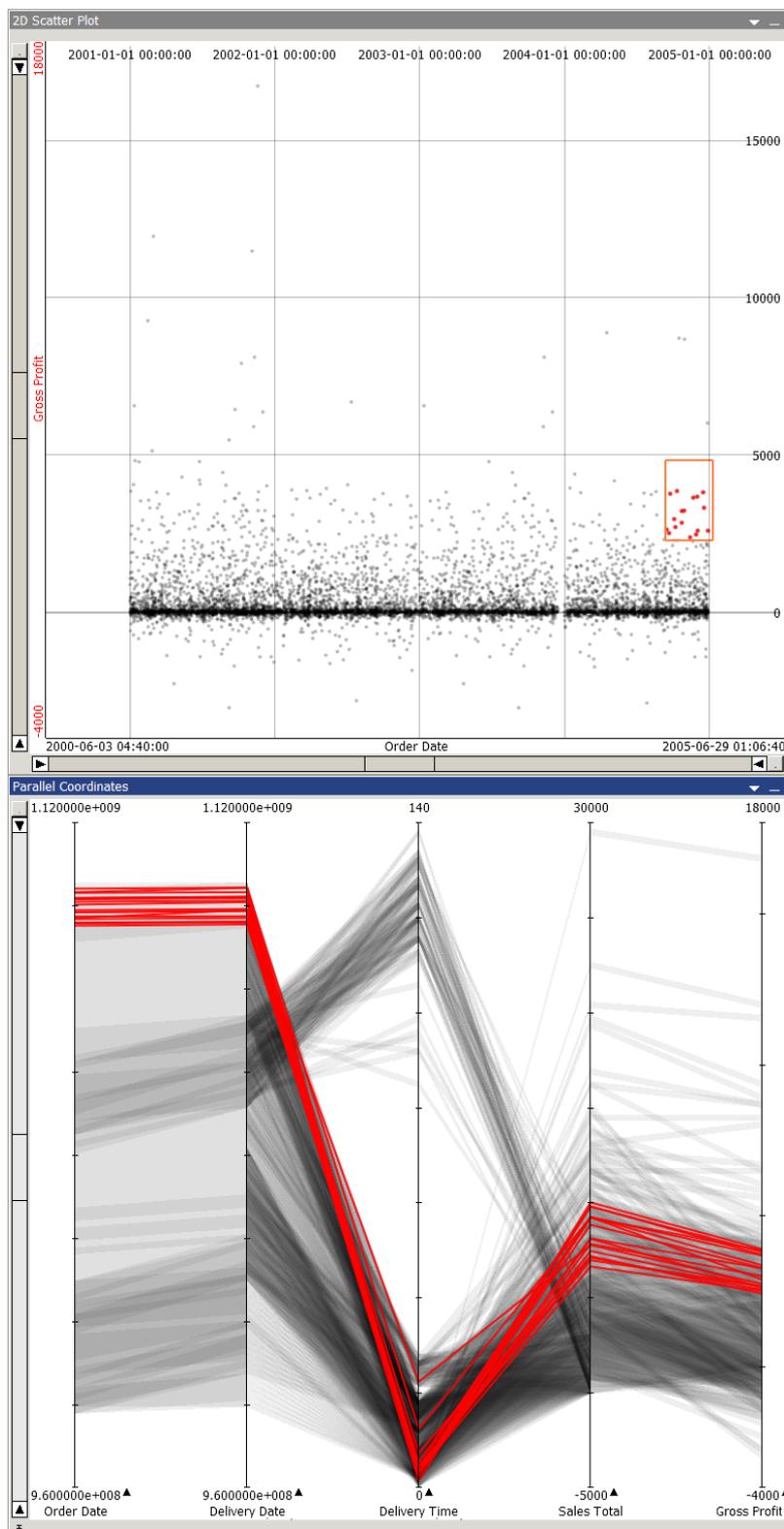


Figure 3.11: Sales data visualized using a 2D scatterplot view on the top and a parallel coordinates view on the bottom.

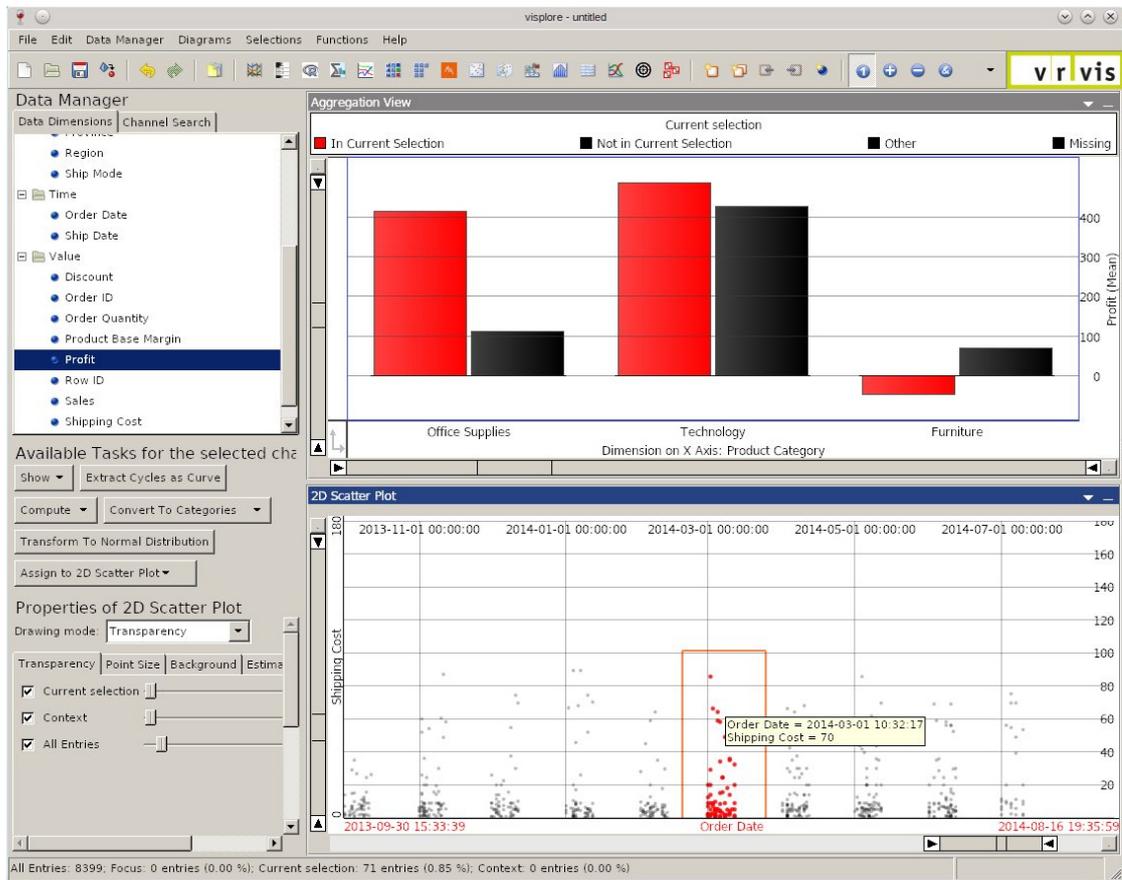


Figure 3.12: An example visualization scenario using VISPLORE. The user has selected some data points in a 2D scatterplot and a bar chart visualizes the average profit per product category for the selected (red bars) and not-selected (black bars) data points.

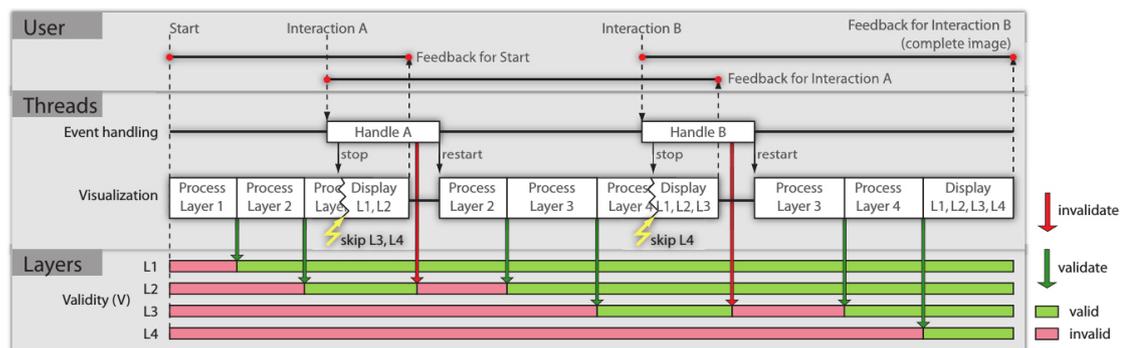


Figure 3.13: Showing the interaction between user, event handling thread and visualization thread. The visualization thread is told to stop after each user interaction and intermediate results are displayed. After updating the parameters, the visualization thread is restarted and re-uses as many layers as possible. L1-L4 represent different layers [PTMB09].

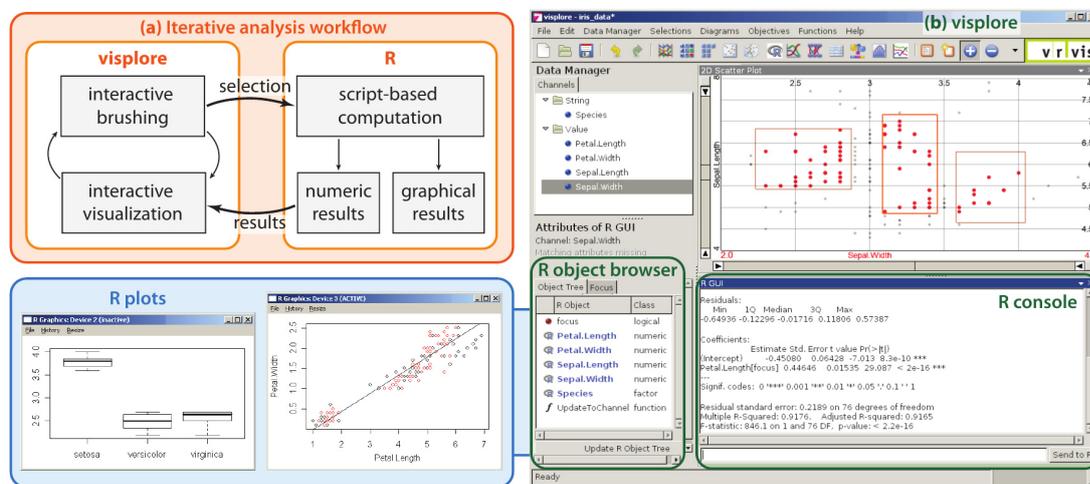


Figure 3.14: This picture shows the integration of R into VISPLORE. **(a)** describes the workflow concept, **(b)** shows VISPLORE and the R parts, which consist of an object browser and an R console [KBFP12]

Plausibility Rules

In this diploma thesis, plausibility rules are used as utility to define data constraints and pre-configured edit actions if these constraints are violated. Rule based approaches to detect quality issues and define transformations are well known in data editing and data warehousing [RD00][Win99][GK97][CD97]. The following sections describe the motivation to use such rules, the structure of rules we used and the corresponding limits.

4.1 Motivation

Real-life data is typically affected by different types of quality issues. There are various taxonomies of error types and it depends on the context of the data which types of errors may arise. A data value "2" can be a valid value if it describes a measured temperature in degrees Celsius, but if it describes the age of a criminal, it can be assumed that it is an erroneous value.

Although some data cleansing applications can detect some kind of data errors automatically [HA04], the available techniques are mostly focused on a specific application area with a well-known data background. To detect data errors in general cases, human perception is still an advantage. Using a general visual analytics framework like VISPLORE, we allow the user to use his human perception to find some reasonable constraints. Such constraints, either found using visualization or deduced from background knowledge, can be defined using plausibility rules. These rules can be automatically applied to new data (within the same context), and rate values as erroneous or rather plausible and not plausible.

Once erroneous values were detected, it follows that these values should be imputed or handled in a specific way. Similar to the error detection, it also depends on the context of the data, which strategy makes sense. To illustrate, an erroneous value describing a temperature in a time-dependent data set could be interpolated, whereas an interpolated value makes no sense in the case that the erroneous value describes the age of a criminal.

The main scope of this diploma thesis is the visual analytics part. We use a variant of rules to define data constraints and imputation strategies, which will be described in the next section.

4.2 Rule Structure

Our concept of plausibility rules consists of several parts.

1. **Rule**

A rule consists of a part which evaluates the data (data check) and a part which defines how data should be imputed (imputation strategy). The rule is always used to evaluate the data and it can optionally be used to impute the data using the predefined imputation strategy.

2. **Data checks**

Checks define how data should be evaluated. In general, they define constraints which can have parameters.

3. **Imputation strategies**

Imputation strategies define how data should be imputed. In general, they encapsulate specific imputation algorithms and can have parameters.

4. **Suggestion strategies**

Suggestion strategies define if and when rules should be created automatically for some data.

5. **Rule set**

A rule set is a set of data checks, imputation strategies, suggestion strategies, and plausibility rules.

The following section now describe these parts in detail:

4.3 Rules

A rule consists of a link to a *data check* and a link to an *imputation strategy*. It allows to check specific constraints on one data column and it provides the possibility to impute values which do not fulfill these constraints.

A data check is responsible for the decision whether a specific data value should be considered as plausible or as implausible. It is further described in Section 4.4. An imputation strategy is responsible to impute implausible values. Section 4.5 describes the imputation strategies in detail.

So a rule R can be formalized as:

$$R = (\text{column}, C, I) \quad (4.1)$$

where *column* is the reference to the data column, C is the used data check and I is the used imputation strategy.

The data checks as well as the imputation strategies are independent objects, which can be used by multiple rules. Hence, multiple rules can check different data columns for an identical constraint by using the same data check.

In general, rules are not restricted to one data column. However, in our system, each rule is bound to exactly one data column. This allows us to directly apply the imputation part of the rule. If a rule is bound to multiple data columns (e.g., "Constraint: $TotalCost = Quantity * Cost$ "), it is not clear which of the values is erroneous and should be imputed if the constraint is not fulfilled. This limitation does not make it impossible to define such constraints, but the rule (and thus, the imputation strategy!) must be explicitly defined for each column (the other columns can be used as input parameters), and each involved value would be rated as implausible if such a constraint is not fulfilled.

4.4 Data Check

A data check represents a specific data constraint and evaluates input values as plausible or implausible. Figure 4.1 shows a blackbox model of an exemplary data check. Formally, a *data check* C returns a plausibility vector p

$$p = (p_1, p_2, \dots, p_n) \quad (4.2)$$

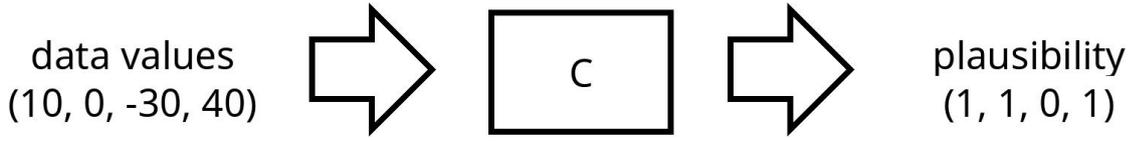


Figure 4.1: A data check C takes input values, has an internal blackbox evaluator function and outputs a plausibility vector. Additional context information is not shown here.

for an input vector v

$$v = (v_1, v_2, \dots, v_n) \quad (4.3)$$

which is actually the tested data column, the used data has n data rows.

If available, an additional input vector t

$$t = (t_1, t_2, \dots, t_n) \quad (4.4)$$

contains the corresponding timestamps, which is actually just another data column. This enables data checks to consider temporal context information. If the input vector t is given, all input and output vectors should be arranged in ascending order with respect to the timestamps in t to ensure that values which are consecutive in time are also consecutive in the data structures. The output values p_i are defined as:

$$p_i = \begin{cases} 1 & \text{if } v_i \text{ is plausible} \\ 0 & \text{if } v_i \text{ is not plausible} \end{cases} \quad (4.5)$$

In more detail, C is a data check with a specific internal plausibility function f_{check} and corresponding parameters P . For each v_i , this internal function is used to determinate the plausibility value. f_{check} could return non-binary plausibility values $p_i^* \in [0, 1]$ to rate data values as more or less plausible.

$$p_i^* = f_{check}(v_i, t_i, P) \quad (4.6)$$

If the output of C is used to decide which values should be imputed, it is necessary to have a boolean output. Therefore, we use a threshold function to get boolean plausibility rules:

$$p_i = \begin{cases} 1 & p_i^* \geq \text{threshold} \\ 0 & p_i^* < \text{threshold} \end{cases} \quad (4.7)$$

To summarize, a fully defined data check C consists of a plausibility function f_{check} and a parameter set P which contains all configurable parameters accepted by f_{check} . Additionally, a name to identify the data check within the system, is necessary. The fully defined data check is formally given as:

$$C = (\text{name}, f_{check}, P) \quad (4.8)$$

The parameter set P is explicitly separated from f_{check} to be able to re-use the same f_{check} with different parameters. For example, given a function f_{check} like

$$f_{check}(v, t, P) = \begin{cases} 1 & v \geq P_{\text{threshold}} \\ 0 & v < P_{\text{threshold}} \end{cases} \quad (4.9)$$

then different data checks could use this function with different parameters. One data check can test if values are not negative ($P_{\text{threshold}} = 0$) and another data check tests if a value is above a reasonable

threshold (e.g., if the values describe the size of a grown-up person in centimeters, $P_{threshold} = 140$ would be reasonable).

A specific data check can be used by multiple rules, and it is meant to describe one specific data constraint. If it turns out that a data check does not represent a specific data constraint in a satisfying way, only the data check must be reconfigured and all the rules are updated implicitly.

The concrete data checks which were implemented are described in Section 7.1.

4.5 Imputation Strategies

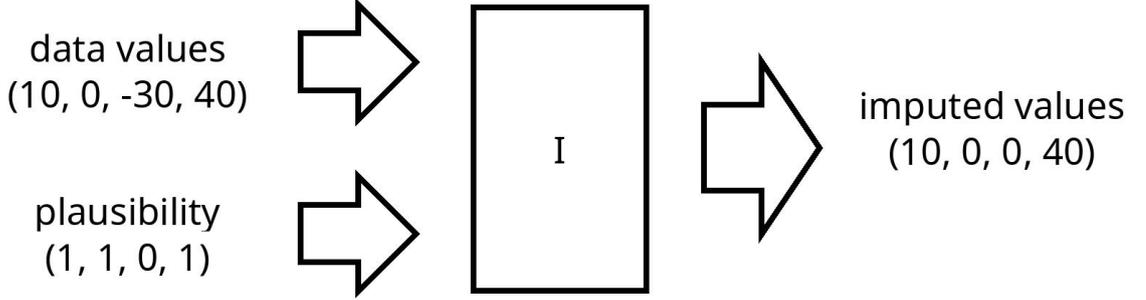


Figure 4.2: An imputation strategy I takes the original values and the plausibility evaluation as input, has an internal blackbox imputation function and outputs the imputed values. Additional context information is not shown here.

An imputation strategy I is responsible for the data imputation, as shown in Figure 4.2. More formally, I creates an output vector $v^{imputed}$, which is similar to the original data vector v , except for all values which are rated as implausible by an arbitrary data check. These values are replaced by applying the internal function $f_{imputation}$.

$$v^{imputed} = I(v, p, t) \quad (4.10)$$

$$v_i^{imputed} = \begin{cases} v_i & \text{if } p_i = 1 \\ f_{imputation}(v, t, p, i) & \text{if } p_i = 0 \end{cases} \quad (4.11)$$

where p is the plausibility vector of a previously used data check, v is the input vector containing the original data values respectively the processed data column, i is the index running from $1..n$ where n is the number of data rows and t is a vector containing the corresponding timestamps.

Similar to the data checks, each imputation strategy has a specific internal function $f_{imputation}$, which defines the concrete imputation algorithm, and a parameter set P which contains parameters used by $f_{imputation}$. This makes it possible that different imputation strategies employ the same algorithm, but use different parameters. A fully defined imputation strategy is defined as follows:

$$I = (name, f_{imputation}, P) \quad (4.12)$$

Similar to the data checks, the concrete imputation strategies which were implemented are described in Section 7.1.

4.6 Suggestion Strategies

As defined in Section 2.2, our framework should be able to provide rule suggestions, that means it should propose a specific plausibility rule on its own initiative. This feature should assist the user in the process of rule creation.

There may be different reasons why a specific rule should be suggested. If a data column contains missing entries, it might be reasonable to define a plausibility rule which highlights missing entries and provides an imputation strategy like interpolation. If we know the background of the data and that a data column should contain percentage values, then it is also reasonable to suggest two rules which constrain the value range between 0 and 100. Without knowing the semantics of the data, automatic suggestions and general approaches are difficult. Therefore, we decided to encapsulate the algorithm which decides if and which plausibility rules should be suggested. For this purpose, our implementation relies on *suggestion strategies*, which encapsulate the concrete decision algorithms and can be adjusted by the user or rather by the content specialist to tune the automatic rule suggestions.

Formally, a suggestion strategy S consists of the internal function $f_{\text{suggestion strategy}}$, corresponding parameters P and a name to identify the strategy:

$$S = (\text{name}, f_{\text{suggestion strategy}}, P) \quad (4.13)$$

While each plausibility rule uses exactly one data check and one imputation strategy, the usage of suggestion strategies differs from this scheme. When a user selects a data column and wants to create rules for this column, all existing and configured suggestion strategies are triggered. Each strategy has access to the current pool of available data checks and imputation strategies, runs its internal algorithm and creates a list of proposed plausibility rules. The data analyst then can accept, modify or discard these suggestions. The concrete implemented prototypes of such suggestion strategies are listed in Section 7.1.

4.7 Rule Set

Once defined, data checks and imputation strategies together with their adjusted parameters and the plausibility rules which employ them, should be reusable. For example, a data analyst may have different data sets with similar structure (e.g., one data set per year), and recreation of all necessary data checks, imputation strategies and plausibility rules would be an annoying time-consuming task. Therefore, a *rule set* is a set of configured data checks, imputation strategies, suggestion strategies, and plausibility rules, which can be exported and imported independently from the data set (using an XML format). Dependent on the concrete application area, it is also possible and reasonable that a rule set only contains configured data checks, imputation strategies, and suggestion strategies but not the plausibility rules. A content specialist may define such a rule set, and the actual plausibility rules are then suggested to the data analyst due to the configured suggestion strategies and further refined by the data analyst.

Overview and Management

This chapter describes the different techniques designed to give an overview of the created plausibility rules in different aspects.

5.1 Motivation

If a data analyst uses plausibility rules or another technique to rate the quality of data values, it is not only interesting *whether* data is dirty or not. Also the location and structure of the erroneous values is an important aspect to get an overview of violated constraints and to get an insight into the possible causes. The location of erroneous values can have different meanings in this case, which are written out in the following questions:

- **Violated plausibility rules**
Which plausibility rules evaluate in high rates of not plausible values? Are there any rules which evaluate to zero not plausible entries, or are there any rules which evaluate nearly all tested values as not plausible? Such rules are an indication that some constraints are wrong or wrongly defined, and the user should be able to find these rules quickly and review these rules in detail.
- **Data columns**
Are there any data columns with a high rate of not plausible values? To analyze failure structure and determinate the causes, the user should have quick access to data columns which have an atypical high number of values which are rated as not plausible. For example, if the data describes sensor measurement values, one specific sensor may be defect and output wrong values, which results in a data column with a high rate of not plausible values.
- **Time-dependent clusters of violated constraints**
Are there any time-dependent clusters of erroneous-values? For example, if a data set contains all planned and operated flights since 2005, an atypical low number of flights combined with an atypical high number of canceled flights would arise around the 15th of april 2010, when the volcano Eyjafjallajökull erupted. To detect such events, it should be possible to get an overview of time-dependent clusters of implausible values.
- **Check-based clusters of implausible values**
Are there any data checks which evaluate to high rates of implausible values? To review the configured data checks and their parameter configurations, it should be possible to get an overview of not

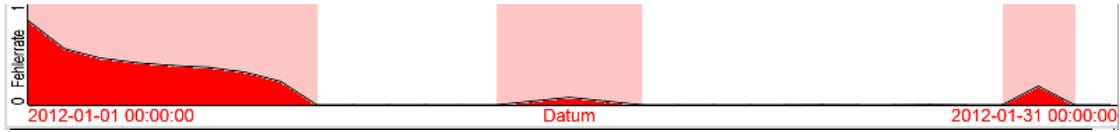


Figure 5.1: A mockup of the data-based overview in an early development stage. For each data row, mapped on the X axis, the rates of rejecting plausibility rules is mapped on the Y axis. Note that a rule works on a data column, while this overview works on data rows.

plausible values based on the data check, which was responsible for the rating. High rates of not plausible values may be caused by erroneous data, but it is also possible that wrongly configured data checks rate too many values as not plausible. An overview based on the used data checks should support the data analyst to answer this question.

To target these questions, we propose three linked overviews which provide different views on the data and the plausibility rules. The first overview is a *data-based overview*, which gives a fast insight into the overall data and rows-based localization of erroneous values. This overview is described further in Section 5.2. The second overview is a *group-based overview*, which provides different grouping mechanisms and a quick insight into the location of erroneous values for a selected group of rules. Section 5.3 describes this overview in detail. The third overview, as described in Section 5.4, lists all rules which match the current group filter and gives an overview of the evaluated error pattern for each rule independently.

5.2 Data-based Overview

The data-based overview is designed to get a quick overview of the failure structure with respect to the data rows. Therefore, it orders the data rows on the X axis and shows additional information about the failures on the Y axis. The ordering on the X axis can be defined by the user and is described later in this chapter.

Using the data rows as one dimension of a plot, the problem of spatial scalability arises. Typical data has a high number of data rows but only limited space is available on the screen. To accommodate this restriction, we use the following approach: If more data rows than available pixels in the drawing area exist, the data rows are binned into groups, and each group is at least one pixel wide.

5.2.1 Variants

On the way to our final data-based overview, we have designed different variants which differ in the form which information is plotted on the Y axis. In this subsection, the different variants and their potential problems, together with our final method, are described.

1. Variant 1: Mapping the failure rate f_1 for each data row on the Y axis

This was our initial approach and can be also seen in Figure 5.1, which shows one of our first mockups. In this case, f_1 is defined as follows:

$$f_1(\text{data row}) = \frac{|\text{failed_rules}(\text{data row})|}{|\text{rules}|} \quad (5.1)$$

To highlight data rows where at least one rule signals not plausible values, the area below the line which connects the plotted values is filled with dark red, while the area above is filled with a bright red.

It turns out that this approach has multiple problems: First, a high number of rules typically leads to a low failure rate, which is often not noticeable with this plotting method. To attenuate this problem, we introduced the already mentioned highlighted areas wherever $f_1(\text{data row}) > 0$ (can be also seen in the mockup). Additionally the failure rate is not that smooth as we rendered it in our mockup, which results in a rather noisy plot. Finally, an open question was the aggregation of different failure rates of multiple rules, if the rules are binned into groups as described above. The noisy plot and the problem with binned rules lead to a simplified plot, which is described in Variant 2.

2. Variant 2: Reducing to 1D, use color for different cases

The second approach is a simplified variant of the first, as it turns out that the first variant does not scale with a high number of plausibility rules. In this approach, the plot is reduced to the X axis, and the area above is either colored in its full height or it is empty. In addition, color can be linearly interpolated, if multiple data rows are horizontally mapped on an area with 1 pixel width. To avoid that bins with a small number of not plausible values remain unnoticed because of the linear color interpolation, the interpolation is defined as follows:

$$colorintensity = \begin{cases} 0.2 + 0.8 * f_2(bin) & \text{if } f_2(bin) > 0 \\ 0 & \text{if } f_2(bin) = 0 \end{cases} \quad (5.2)$$

where f_2 is the failure rate for each bin and is defined as follows:

$$f_2(bin) = \frac{\sum_{r=1}^{|\text{data rows in bin}|} failure\ rate(data_row_r)}{|\text{data rows in bin}|} \quad (5.3)$$

The aim of this overview is to give a fast insight into the location of errors regarding the data rows, not to visualize the exact rate of not plausible values. Therefore it is arguable to introduce a step function between 0.0 and 0.2 for color intensity. It helps to identify the areas where data rows contain not plausible values, even if there are many data rows aggregated into one bin.

Furthermore, we have used two different colors, which represent the following two cases:

- **Red:** One plausibility rule marks entries on the current data row as implausible
- **Blue:** Two or more plausibility rules mark entries on the current data row as implausible

We have chosen these two colors because they are separated in the used RGB color space representation. This means, if one data row is marked by two rules as not plausible and another data row is marked by exactly one rule and both rows are binned into one group, the resulting RGB representation keeps both details. The blue part represents the percentage of rows with more than one marked values, while the red part represents the rows with exactly one marked value. More formally, the color is calculated as:

$$color(bin) = \frac{m \cdot rgb(255, 0, 0) + n \cdot rgb(0, 255, 0)}{N} \quad (5.4)$$

$$N = n + m \quad (5.5)$$

where m is the number of data rows rated as not plausible by exactly one rule, n is the number of data rows marked as not plausible by multiple rules and N is the number of data rows grouped into the current bin.

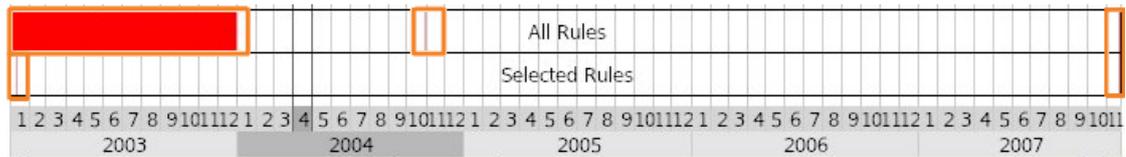


Figure 5.2: The data-based overview, showing implausible values detected by all rules and by a selected subset. For clarification, the implausible values are highlighted by orange rectangles in this screenshot. The data rows are sorted and grouped by the values of a data column which represents the timestamp. It can be clearly seen that most of the not plausible values happened in 2003.

3. Final variant: Using 1D and only one color

The final variant is similar to the described variant 2, but only one color is used. While using a prototype of variant 2, we recognized that the visual distinction between blue and red tones is more difficult than expected in those cases, where many data rows are grouped into bins. Additionally, it has turned out that for a quick overview of the data, it is not relevant whether some rows are found to be erroneous by one or two rules. The important aspect seems to be whether rows are found to be erroneous or not, and the details can be inspected with the support of other views.

Therefore, in the final variant, data rows which are found to be erroneous by one or more rules, are visualized by a red color, which is also interpolated because of the grouping into bins, if necessary. Figure 5.2 shows the final overview in use.

5.2.2 Partitioning and Ordering

The used data format (see Table 1.1) does not provide an additional semantic which defines the order of the available data rows, so rows are sorted by their implicit order as given in the loaded data set. Dependent on the context of the data, there may be a column which actually describes an order of the data, e.g., spatial or temporal. For illustration, if there is data which includes sensor measurements and one data column contains for each row the point of time when the values were recorded, then this column can be interpreted as logical order of the data.

To use the values of a data column to determinate the order of the data rows, the user can define a "order column". Although we are focused on time-oriented data in this thesis, the user may select any quantitative or temporal data column and set it as order column.

Additionally, data columns containing categorical data can be used as *partitioners*. A partitioner splits the data rows into subsets, which are used to separate them visually in the data-based overview. Within the subsets, the data rows are still sorted by the configured order. It is possible to define multiple partitioners, the resulting subsets are intersected accordingly and displayed as hierarchical groups. In addition to explicit categorical data columns, partitioners may also be defined implicitly as the evaluation result of plausibility rules. This subdivides the data rows into the two groups "Rated as plausible by Rule x " and "Rates as implausible by Rule x ". For example, consider the small data set of Table 5.1: The user has set the column Time to define the ascending order of the rows and he has set two partitioners, column A as first partitioner and column C as second partitioner. Based on these partitioners, the following subsets, containing the corresponding data row indices, exist:

- **A**
 $a_0\{0, 3, 5\}$, $a_1\{1, 6\}$ and $a_2\{2, 4\}$
- **C**
 $c_0\{0, 1, 3, 5\}$ and $c_1\{2, 4, 6\}$

Row Index	A	B	C	Time	Measurement Data 1	...
0	a0	b0	c0	00:09	18	...
1	a1	b0	c0	00:08	12	...
2	a2	b1	c1	00:07	15	...
3	a0	b2	c0	00:06	11	...
4	a2	b1	c1	00:05	11	...
5	a0	b1	c0	00:04	10	...
6	a1	b2	c1	00:03	11	...

Table 5.1: Example data set to illustrate the use of partitioners

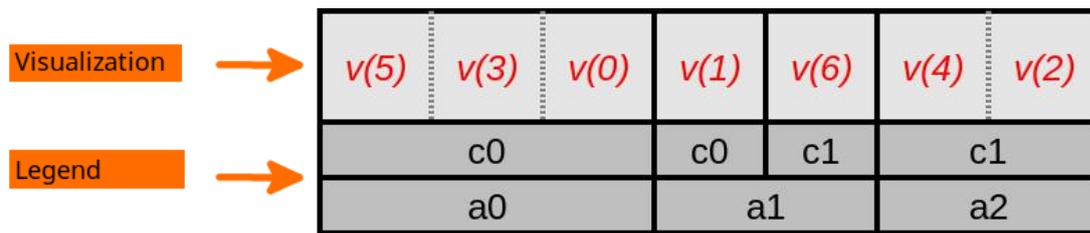


Figure 5.3: Illustration of the visualization using partitioners. In this example based on table 5.1, the columns A and C are used as partitioners. The partitioners divide the data rows into groups which are separated visually. As defined by the user, the rows are ordered by the column Time within each group. $v(r)$ returns the visualization color for the data row r .

These subsets are intersected:

- $a0 \cap c0 = \{0, 3, 5\}$
- $a0 \cap c1 = \{\}$
- $a1 \cap c0 = \{1\}$
- $a1 \cap c1 = \{6\}$
- $a2 \cap c0 = \{\}$
- $a2 \cap c1 = \{2, 4\}$

Based on these subsets, the visualization is done. The ordering of the subsets can be optionally configured by the user, but for simplicity, let us assume that the user uses the default ordering. Within the intersected subsets, the order of the rows is given by the set order column, which is the column Time in this concrete case. Figure 5.3 illustrates the visualization and grouping of data rows. A real visualization using Year and Month as partitioners can be seen in Figure 5.2.

5.3 Group-based Overview

The created rules can be seen from different points of view. They have implicit properties like the relative number of values which were rated as not plausible or explicit properties which can be assigned by the user like a custom tag.

These properties can be used to group rules and get additional insight into the failure structure of the data. We have designed a group-based overview which provides an overview of the data quality and failure structure by grouping and aggregating the results of the rules using these properties. We have identified the following properties:

1. Custom Tags

Compared to the other properties, which are implicit properties, tags are an explicit property and can be assigned by the user. Custom tags give the user the ability to group rules manually. Using the rule-based overview, which is described later, he can select rules and assign custom tags to them. Tags can be assigned to multiple rules at once, furthermore they can be modified and removed. Within the group-based overview, each used tag represents a group and all rules which have a specific tag assigned belong to the corresponding group.

This tagging system allows the user to manually organize rules. For example, if a subset of the data describes measurement values while another subset describes forecasts and thus corresponding plausibility rules exist, he can tag the rules for the measurement values with the tag "measurement" and the rules for the forecast values with the tag "forecasts". Using this system, he adds two user-defined groups "measurement" and "forecast" to the group-based overview and can evaluate those groups separately.

2. Frequency Classification

Using the frequency of implausible values, the user can quickly find plausibility rules which evaluate all values as plausible, or plausibility rules which evaluate nearly all values as implausible. This property is split into the following groups:

- a) all values plausible ($ratio = 0$)
- b) nearly all values plausible ($0 < ratio \leq 0.01$)
- c) few values not plausible ($0.01 < ratio \leq 0.05$)
- d) some values not plausible ($0.05 < ratio \leq 0.25$)
- e) many values not plausible ($0.25 < ratio \leq 0.90$)
- f) nearly all or all values not plausible ($0.90 < ratio \leq 1.0$)

$ratio \in [0, 1]$ is the rate of implausible values.

The asymmetric definition of the groups is motivated by two reasons: If a rule rates 90% or 95% of the values as implausible makes not really a difference for the data analyst, the data column seems to have a bad data quality anyway. Additionally, if a plausibility rule rates nearly all or all values as not plausible, is more likely that the rule is wrongly configured.

3. Pattern Clusters

Data quality issues are not limited to single data entries. Depending on their actual cause, they may affect multiple entries in a data row or multiple data rows. For example, a power blackout in a measurement station will affect multiple sensors and thus, produce not plausible values on multiple data columns on specific data rows. To find quality issues with a similar pattern of implausible values (with respect to the affected data rows), we have implemented a property which represents clusters of plausibility rules which rate similar data rows as not plausible. The algorithm to achieve this works as follows:

- (i) For each rule, the similarity to all other rules is computed using the hamming distance h , which is defined as:

$$h(rule_1, rule_2) = \sum_{n=1}^{\text{number of data rows}} f(rule_1, row_n) \neq f(rule_2, row_n) \quad (5.6)$$

where f is the result of a plausibility rule for a given data row. So if two plausibility rules rate exactly the same data rows as not plausible, the hamming distance is $h = 0$. If one rule rates all data rows as not plausible which are rated as plausible by another rule and vice versa, the hamming distance is $h = \text{number of data rows}$. For each rule, all other rules which have a hamming distance below a specific threshold $h_{\text{threshold}}$ are grouped into a cluster. In our work, we have identified by experimentation that $h_{\text{threshold}} = 0.10 * \text{number of data rows}$ seems to be a suitable threshold.

- (ii) The resulting clusters are sorted descending by their number of corresponding rules.
- (iii) The sorted clusters are processed from large groups to small groups and all rules within the current cluster are removed from other clusters. This is done to get disjoint clusters and whenever a rule belongs to multiple clusters, the greater cluster is preferred compared to smaller clusters.

4. Used Data Check

As described in Chapter 4, data checks can be used by multiple rules. To get a fast overview of all rules which use the same data check, each data check creates an own group.

5. Column Hierarchy

In some data sets that we used for evaluation, data columns have nested names which can be interpreted as hierarchy. For example, data containing weather measurements may have columns like ("Vienna/Temperature", "Vienna/Wind Speed", "Salzburg/Temperature", "Salzburg/Wind Speed"). Using these four columns as an example, there would be two resulting groups: "Vienna" and "Salzburg". Plausibility rules which test "Salzburg/Temperature" and "Salzburg/Wind Speed" would belong to the group "Salzburg" and the rules which test "Vienna/Temperature" and "Vienna/Wind Speed" belong to the group "Vienna".

6. Column Name

The column name is the last part of the full data column name or the full data column name without the hierarchy. For ("Vienna/Temperature", "Vienna/Wind Speed", "Salzburg/Temperature", "Salzburg/Wind Speed") the column names would be ("Temperature", "Wind Speed", "Temperature", "Wind Speed"). As an example and using the given data columns, this property helps the user to inspect and review all rules which evaluate data columns containing the temperature, regardless of which city.

Each of these properties defines different groups of rules. These groups can be used to refine which rules should be listed in the rule-based overview. Each group which contains rules is represented by a button below a headline button which labels the current property (see Figure 5.4).

To improve scalability, these properties have different priorities, which correlate to the order of the list above ("Column Name" has the lowest priority, "Custom Tags" has the highest priority). Selecting a group acts like a filter for all plausibility rules, only those which are included in the selected groups are shown in the rule-based overview. Furthermore, in the displayed list of groups, the filter is also applied to groups which have a lower priority than the selected group (respectively their corresponding properties). The user can select multiple groups of the same property, in this case, all rules which belong to one of these groups are shown in the rule-based overview. If multiple groups of different properties are selected, only rules which belong to all these groups, are shown.

The priorities and ordering of the different properties have been chosen to allow the user to go "from the big to the small" when using our sample data sets. That means that filtering groups with a high priority filters a large number of rules, while filtering groups with low priority filters less rules. Although suitable priorities may differ based on the used data set, they are fixed in the current version. Modifiable priorities or at least a deeper analysis of the predefined priorities should be considered for future work.

Additionally, each button which represents a group provides different data about the corresponding rules:

- **Relative number of rules in group**

A gray bar on the upper half of the button shows the ratio of rules which are not filtered and belong to the group. If all rules which are not filtered belong to this group, the ratio is 1.0 and the gray bar is drawn in full width.

- **Absolute number of rules in group**

Hovering a button displays a tooltip which shows the absolute number of rules which match the current filter and the hovered group.

- **Rate of implausible values**

A red bar on the lower half of the button shows the rate of values which are evaluated to be not plausible by rules which belong to this group and the current filter. The calculation of this ratio is done by using the set of data rows which is evaluated as not plausible by any of the corresponding plausibility rules, or more formally $\frac{|not\ plausible\ data\ rows|}{|data\ rows|}$.

In Figure 5.4, the implemented group-based overview can be seen as part of the rule manager view. In Figure 5.5, a group is selected and thus, only rules which belong to this group are shown in the rule-based overview.

5.4 Rule-based Overview

The rule-based overview lists all rules which meet the current filter based on the selection in the group-based overview. Before we finally decided to use a similar visualization method as for the data-based overview, the design process involved several stages:

1. **Using a tabular layout**

The first approach was a tabular layout. Each row represents a plausibility rule and the columns contain a visual representation of the evaluation result and different properties of the rules like the used data check, tested data column, number of implausible values, and some others. The visual representation of the evaluation result is done in the same way as it is done for multiple rules in the data-based overview. Figure 5.6 shows a screenshot of this early approach.

Using the prototype, it turned out that this approach has some problems. The columns which provide detailed information about the properties of the rule (used data check, tested data column, number of implausible values, ...) are using too much of the horizontal space, so that the visual representation of the plausibility evaluation gets too less space. It would be preferable if the visual representation uses more of the horizontal space. Additionally, even if the visual mapping of data rows to screen space is the same as it is used in the data-based overview, it differs in width. We found that this may be confusing and lacks visual comparability between data-based overview and rule-based overview, which lead us to the following approach:

2. **Using full-width visual representation plus additional information**

The second approach abandons the tabular layout and arranges the cells with additional rule details above the visualization of the evaluation result. This leads to a full-width visual representation of the evaluation result, which has the advantage that it uses the same width as the data-based overview.

Figure 5.7 shows a screenshot of this approach. However, this approach lacks overview if there is a large number of plausibility rules, because each plausibility rule now uses much more vertical space.

3. Using full-width visual representation, details on demand

The final variant is similar to the second variant, but moves all additional rule details into the tooltips which are displayed as soon as the user moves the mouse cursor over a displayed plausibility rule. This approach uses less vertical space but allows a full-width visual representation for each rule, and provides additional details on demand.

However, the evaluation result of the rules are actually always shown "in-detail", which means that the evaluation result is shown for all data rows. Considering the visual analytics concept "Overview first, details on demand", there is room for further improvement, e.g., using aggregated visualizations and showing the full evaluation result only on demand. See Chapter 9.3 for further thoughts on future work. The final variant as implemented in this thesis is shown in Figure 7.2.

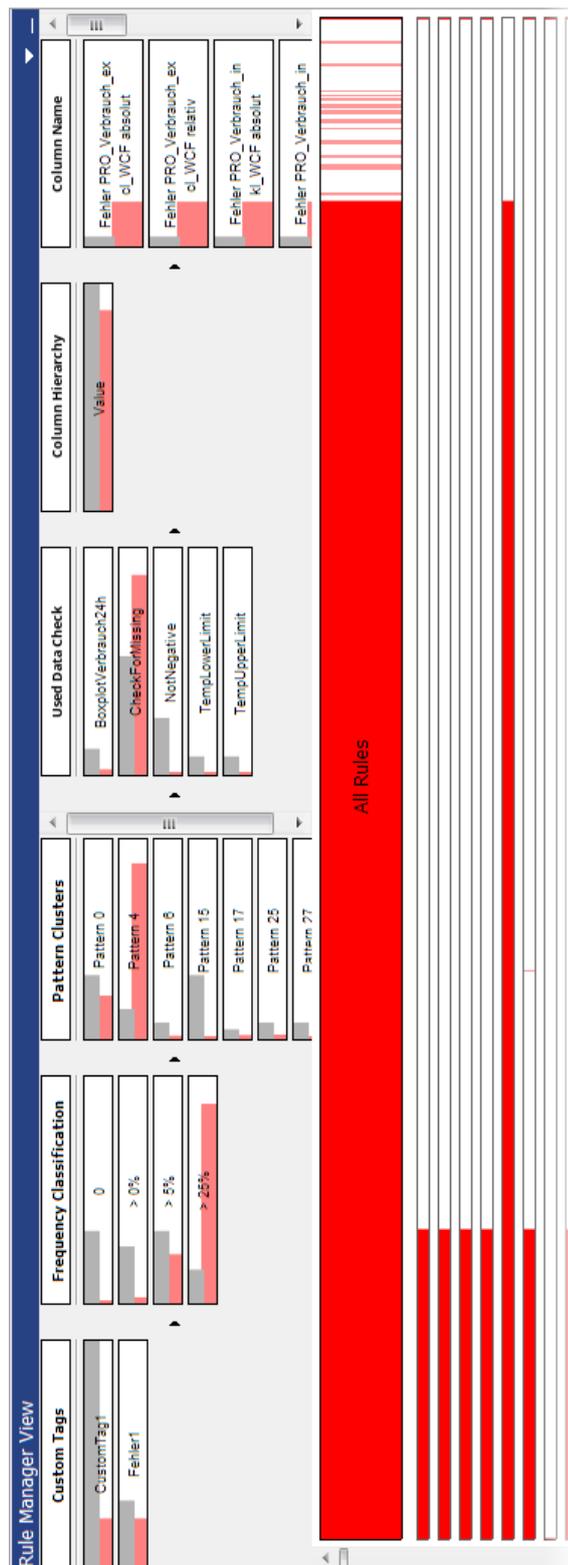


Figure 5.4: A screenshot of the Rule Manager View, including the group-based overview. The different properties and groups can be seen in the upper part of the view. On the upper half of each button, a gray bar represents the percentage of rules which belong to this group (e.g., all rules belong to the group "CustomTag1", while only a few rules belong to the group "Pattern 17"). On the lower half of each button, the percentage of implausible data rows, as evaluated by the corresponding rules, is shown (e.g., nearly all data rows are rated as implausible by at least one of the rules which use the data check "CheckForMissing").

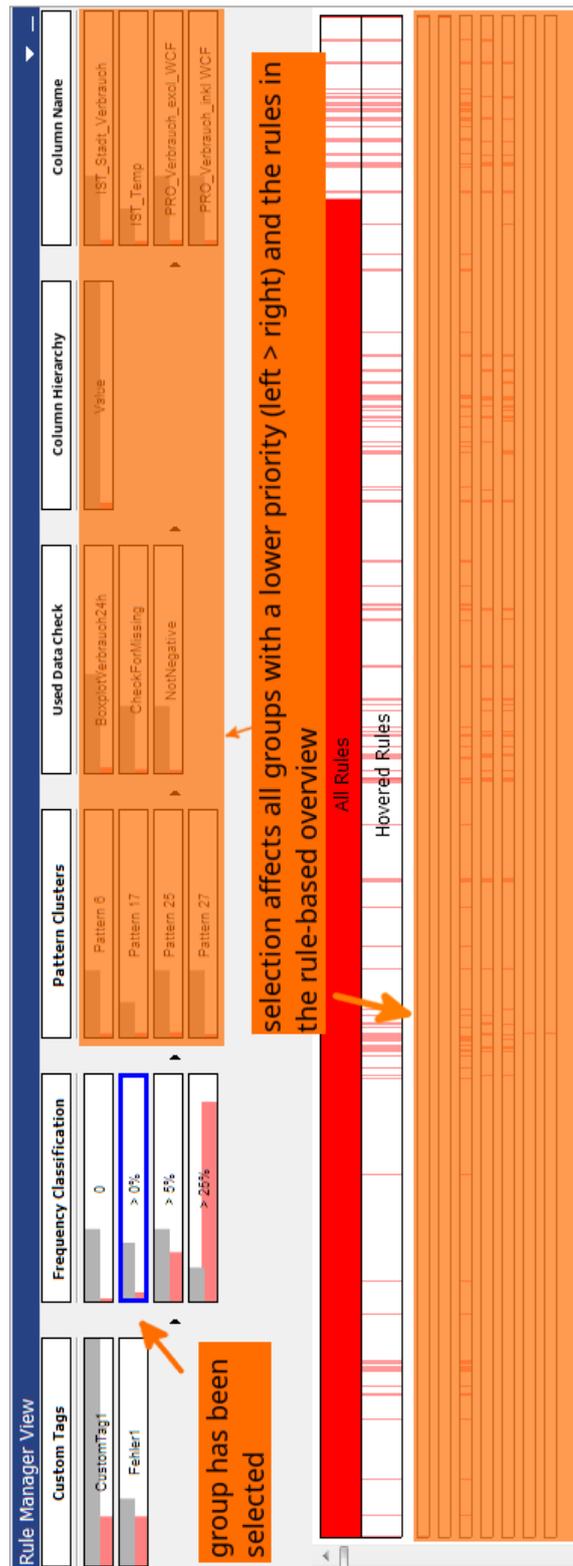


Figure 5.5: A screenshot of the Rule Manager View, including the group-based overview. The group ">0%" has been selected and thus, groups with a lower priority are filtered. The shown plausibility rules in the rule-based overview, which is located below the group-based overview, are also filtered accordingly.

Tested data column	Visual representation	failed values
Vienna/Temperature		2
Vienna/Gas Consumption		9765
Salzburg/Temperature		8837

Figure 5.6: A screenshot of the first prototype of the rule-based overview, showing three different plausibility rules. The first prototype used a tabular layout and placed the visual representations of the plausibility evaluations in a column, and additional rule details (tested data column, number of implausible values) in further columns.

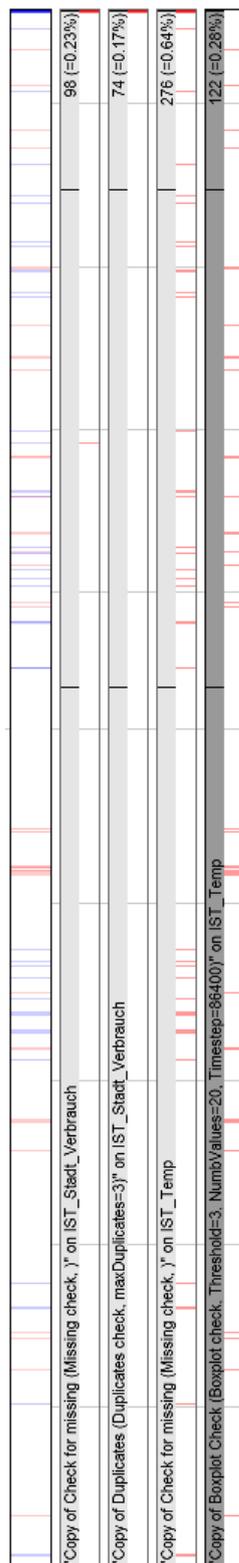


Figure 5.7: The second prototype of the rule-based overview, together with the data-based overview on top. The visual representation of the plausibility evaluation is similar and the user can compare the location of implausible values. Additional rule details (used data check, tested data column, number of implausible entries) are now displayed for each rule above the visual representation, using vertical instead of horizontal space.

Imputation

This chapter is going to describe how data values are imputed using the plausibility rules. Furthermore, it describes which additional meta data is created when a plausibility rule is applied to the tested data column and the motivation for these meta data.

6.1 Motivation

Data editing does not only consist of detecting implausible values, it also includes steps to impute these values in a way which is reasonable for a given purpose. There is no common editing algorithm which is reasonable for all situations. It rather depends on the data quality issues itself and the purpose of the data (e.g., modelling, reporting, ...).

As described in Chapter 4, our notion of plausibility rules includes an imputation strategy in addition to the strategy for detecting data quality issues. If data values were modified by any of these rules, one of the next questions for the user may be why and how. Hence, if these rules are applied to the data column, it should be traceable which values were modified, why these values were modified, and on which imputation strategy this new value is based on.

6.2 Applying Plausibility Rules

A plausibility rule as described in Chapter 4 consists of a data check of the tested data column, and an imputation strategy of presumably implausible values. In contrast to the evaluation of the tested data, which is done as soon as rules are created and every time when the content of a data column changes, the imputation must be explicitly triggered by the user. The user has the possibility or responsibility to review the evaluation of the plausibility rules, before the changes are applied. The user can trigger the imputation either for all rules or for a subset of the rules which is selected in the rule view (see Section 5.4).

6.3 Created Meta Data

If data values are imputed by plausibility rules, it should be traceable for the user why some values were changed and how they were changed. Therefore, if data values are imputed, additional data columns

containing meta information about the applied imputations are created or modified automatically. The additional data columns can be used and read by existing tools like other data columns.

We have identified the following questions, which arise as soon as a user employs a potentially edited data column:

- Are there any modified values?
- What are the original values?
- Which plausibility check was initially responsible for rating a value as not plausible?
- Which imputation was responsible for the current value?
- If values were changed more than once: How did the values change over the imputation passes?
This history can also be of interest for the user if imputation artifacts arise. Such artifacts would arise if one plausibility rule X imputes a value so that it seems to be implausible for plausibility rule Y afterwards, which then imputes the value again. Using a full history of imputation steps, such imputation artifacts can be detected.

To be able to answer these questions, applying a plausibility rule does not only impute a data column, it also leads to the following new or updated data columns:

- *\$ColumnName/Original values*
Contains the original values of the data column *\$OriginalName*. These values are always the original values of a data column, no matter how often values of the data column were edited.
- *\$ColumnName/First applied data check*
Each entry contains the name of the data check, which was the first data check to evaluate the corresponding original value as implausible. If the original value is plausible and not yet imputed, this entry contains a missing value.
- *\$ColumnName/Last applied imputation strategy*
Each entry contains the name of the imputation strategy, which is responsible for the current value in the corresponding data column. If the original value is plausible and not yet imputed, this entry contains a missing value.
- *\$ColumnName/History of changes*
Each entry contains a string which describes the history of changes of the corresponding value in the data column. If the corresponding value has not been imputed yet, this string is empty, else, for illustration, it may look like:
"99.0 -> MISSING (OutlierCheck failed, executed SetToMissing); MISSING -> 12.3 (IsNotMissingCheck failed, executed SetToInterpolation)".

These additional meta information can be inspected using existing tools, e.g., the detail view in VIS-PLORE. Figure 7.7 shows an example.

Integration into VISPLORE

In this chapter, we will describe the integration into VISPLORE. This includes the plausibility rules, the overview and management frontend and finally the possibilities to apply these rules and impute data.

The first section describes some prototyped types of data checks, imputation and suggestion strategies with respect to the internal functions. The following section explains how users can create plausibility rules in VISPLORE, complemented by the subsequent section which describes the provided view to manage the rules and get an overview. Finally, the last section informs about the implemented possibilities how to inspect imputed data and compare original and modified values.

7.1 Implemented Prototypes

To evaluate the concept of plausibility rules in VISPLORE, we implemented the subsequent set of concrete types of data checks, imputation and suggestion strategies.

7.1.1 Implemented Data Checks

- **Threshold**

The threshold check tests all values against a specific threshold. The compare operator and the threshold can be set as parameters. This check can be used for range constraints, e.g., wind speed values are only plausible if they are > 0 .

- **IsNotMissing**

In Visplore, data entries can also contain missing values, similar to NULL values in some popular database systems. This check rates the missing values as implausible.

- **Duplicates**

This data check was implemented to detect consecutive data entries which have identical values. These can be implausible in a specific context. For example, if temperature values are gathered from sensors and have at least a minor jitter, arising duplicates can indicate an erroneous preprocessing step of the data or technical errors.

This check has a parameter $P_{max\ duplicates}$ which describes the maximum number of succeeding identical values which is rated as plausible. Evaluating a data column C with N values, for each

value v_i ($i = 1..N$), the plausibility value p_i is determined as follows:

$$p_i = \begin{cases} 0 & \text{if } v_{(i-P_{max\ duplicates})} \equiv v_{(i-P_{max\ duplicates}+1)} \equiv \dots \equiv v_i \\ 1 & \text{otherwise} \end{cases} \quad (7.1)$$

- **Outlier**

The check type Outlier is a statistical prototype and moreover the only prototype, which makes use of the available time column as described in Section 4.4. The evaluation is loosely based on the decision which points should be drawn as outliers when drawing a graphical boxplot.

This check also considers temporal information and was implemented to have a prototype which is able to detect outliers like a low gas consumption on Monday at 03:00, while the consumption is much higher on all other days of the week at 03:00. It has the following parameters:

- $P_{num\ values}$
describes the number of other values in the local neighborhood and must be even.
- $P_{timestep\ in\ seconds} \in \mathbb{N}$ is the considered timestep. For example, if ($P_{timestep\ in\ seconds} = 3600$), which would be an hour, the considered neighborhood for a value $v_{03:05}$, which was measured at 03:05, would be $[\dots, v_{01:05}, v_{02:05}, v_{04:05}, v_{05:05}, \dots]$.
- $P_{outlier\ threshold} \in \mathbb{R}$ which is a multiplier to the interquartile range, which is used to determine outliers. Equation 7.2 describes the influence of this parameter formally.

Evaluating a data column C with N values, for each value v_i with the corresponding timestamp t_i and $i = 1..N$, the check builds a set of all other values $v_j, j = 1..N$ which fulfils $t_j = t_i + x * P_{timestep\ in\ seconds}, x \in \mathbb{G} \wedge x \in [-\frac{P_{num\ values}}{2}, \frac{P_{num\ values}}{2}]$. For the resulting set, the 1st Quartile $c1$, the 3rd Quartile $c3$ and the interquartile range iqr are calculated, which are used to determine the plausibility p_i for v_i as follows:

$$p_i = \begin{cases} 0 & \text{if } v_i < (c1 - iqr * P_{outlier\ threshold}) \\ 0 & \text{if } v_i > (c3 + iqr * P_{outlier\ threshold}) \\ 1 & \text{otherwise} \end{cases} \quad (7.2)$$

- **Residual Threshold**

VISPLORE supports different types of regression models for predicting a quantitatively dependent variable Y as a function of one or more continues or categorical independent variables X_1 to X_n . Regression models are frequently used for model-based anomaly detection. The idea is to consider data values as anomalies that deviate from the model by more than a given threshold. This type of check has two parameters:

- $P_{function}$
which is the function.
- $P_{residual\ threshold}$
which defines the residual threshold

Evaluating a data column C with N values, for each value v_i ($i = 1..N$), the plausibility value p_i is determined as follows:

$$p_i = \begin{cases} 1 & \text{if } |P_{function}(X_i, i) - v_i| \leq P_{residual\ threshold} \\ 0 & \text{otherwise} \end{cases} \quad (7.3)$$

- **Selection**

The key idea of the selection data check is to let the user define subsets of the data as plausible or implausible by using these interactive selection methods provided by VISPLORE. There are multiple possibilities to interactively select a subset of the data in VISPLORE. For example, in a 2D scatterplot, selection components can be defined by brushing rectangles and linked using logical operations (AND, OR, SUBTRACT). Furthermore, it is a key feature to link these selection components across different views.

Caused by the limitation of our implementation which allows only univariate data checks, the actual data check does not evaluate the selection by itself, instead it uses the binary result of a selection for its internal evaluation function. The binary result of the selection, which is "selected" or "not selected" for each data row, is directly used as plausibility output. This may be confusing, e.g., if a selection is based on the data columns A and B and defined as $(10 < A < 20 \text{ AND } 5 < B < 30)$, a selection data check can be used to evaluate data column C, however, the values of the data column C are irrelevant for the evaluation result. In Chapter 9.3, possible future work considering multivariate data checks is discussed. Using multivariate data checks, a clean implementation of a selection data check would be possible.

- **R script**

The idea behind this check type is that the user can implement a data check algorithm on his own if a wanted type of data check is not implemented yet. Using an R script as data check, the user can define arbitrary data checks based on the R language, which is supported by VISPLORE as described in Section 3.3.3.

Compared to the other implemented checks, which have a hard-coded evaluation function, checks based on R scripts have a worse performance and additional parameters are not allowed. This is related to the user interface, which will be described in Section 7.2. It was initially not designed to administrate scriptable data checks. Scriptable data checks have the advantage that they allow the use of all statistical methods which are already implemented in the corresponding script language, and R has a great variety of implemented statistical methods [KBFP12]. Hence, improvements considering the handling of scriptable data checks are interesting enhancements and therefore discussed in Chapter 9.3.

Technically, using this data check it is possible to define an arbitrary R script which gets the input vectors *value_channel* and *time_channel* to calculate a result vector *plausibility_channel*. The input vectors are automatically converted and transferred from VISPLORE to the R workspace, and the output vector is automatically converted from the R workspace to a data column in VISPLORE. This type of data check accepts a parameter P_{script} that contains a string, which defines the actual R script. To illustrate, if a data check which rates all negative values to implausible should be created, P_{script} can be set as follows:

$$P_{script} = \text{"plausibility_channel} < -(value_channel >= 0)\text{"} \quad (7.4)$$

7.1.2 Implemented Imputation Strategies

- **SetToValue**

The SetToValue imputation replaces all not plausible values with a constant value. To configure this constant value, the check type provides the parameter P_{value} .

- **SetToMissing**

Similar to the SetToValue imputation, all not plausible values are replaced by a constant value. But using this imputation, the value is always set to MISSING.

- **Interpolation**

Another implemented imputation strategy is the linear interpolation. For example, if temperature values contain implausible outliers which may be caused by technical defects of the sensor, a plausible imputation strategy may be to interpolate the outliers by using the plausible values in the local neighborhood.

Technically, this imputation strategy replaces implausible values by using their plausible predecessor and successor values to calculate an interpolated value. The following listing describes the used interpolation algorithm in detail:

```

1 double GetInterpolatedValueForIndex(int iIndex)
2 {
3     int iIndexValidValuePrior;
4     int iIndexValidValueAfter;
5     bool bFoundValidValuePrior = false;
6     bool bFoundValidValueAfter = false;
7     double dInterpolatedValue;
8
9     // Search previous plausible value in local neighborhood
10    for (int iTempIndex = iIndex - 1;
11         (iIndex >= 0) && (bFoundValidValuePrior == false);
12         iIndex --)
13    {
14        if (IsValuePlausible(iTempIndex))
15        {
16            iIndexValidValuePrior = iTempIndex;
17            bFoundValidValuePrior = true;
18        }
19    }
20
21    // Search next plausible value in local neighborhood
22    for (int iTempIndex = iIndex + 1;
23         (iIndex < iNumValues) && (bFoundValidValueAfter == false);
24         iIndex ++)
25    {
26        if (IsValuePlausible(iTempIndex))
27        {
28            iIndexValidValueAfter = iTempIndex;
29            bFoundValidValueAfter = true;
30        }
31    }
32
33    if (bFoundValidValuePrior && bFoundValidValueAfter)
34    {
35        double dValuePrior = GetValue(iIndexValidValuePrior);
36        double dValueAfter = GetValue(iIndexValidValueAfter);
37
38        int iInterpolatedSteps = (iIndexValidValueAfter - iIndex) +
39                                (iIndex - iIndexValidValuePrior);
40
41        double dFactorPrev = (iIndexValidValueAfter - iIndex) / iInterpolatedSteps;
42        double dFactorAfter = (iIndex - iIndexValidValuePrior) / iInterpolatedSteps;
43
44        dInterpolatedValue = dFactorPrev * dValuePrior + dFactorAfter * dValueAfter;
45    }
46
47    else if (bFoundValidValuePrior && !bFoundValidValueAfter)
48    {
49        dInterpolatedValue = GetValue(iIndexValidValuePrior);
50    }
51    else if (!bFoundValidValuePrior && bFoundValidValueAfter)

```

```

52 {
53   dInterpolatedValue = GetValue(iIndexValidValueAfter);
54 }
55 else
56 {
57   // no valid value was found in the whole
58   // data vector
59
60   dInterpolatedValue = 0.0;
61 }
62
63 }

```

Listing 7.1: Pseudocode describing the interpolation algorithm

- **R script**

Similar to the implemented checks, there is also an imputation strategy which wraps an arbitrary R script. The motivation for this script wrapper is the possibility for the user to implement his own imputation strategy. Additionally, the use of R as script language provides a huge number of already implemented and well tested statistical methods (see Section 3.3.3).

Compared to the other imputations, which use a hard-coded and compiled imputation algorithm, the script based imputation has a worse evaluation performance. Furthermore, the implemented GUI to manage the used imputation strategies was not designed to allow arbitrary imputation scripts. However, using script based imputation strategies is a powerful feature as it enables the use of any scriptable imputation strategy and removes the need of concrete pre-defined and implemented imputation strategies. In Chapter 9.3, possible improvements and future work on this feature are discussed.

Technically, the script based imputation works similar to the R-based data check as described in Section 7.1.1. The vectors *plausibility_channel*, *values_channel* and *time_channel* are automatically transferred to the R workspace before the script is executed, and after the execution the output vector *modified_channel* is automatically transferred back from the R workspace to the VISPLORE environment. To define the R script, the imputation strategy expects the parameter P_{script} . For example, to define an imputation strategy which replaces all implausible values by 0, P_{script} can be set as follows:

$$P_{script} = "modified_channel < -value_channel; \\ modified_channel[plausibility_channel == FALSE] < -0" \quad (7.5)$$

7.1.3 Implemented Suggestion Strategies

We have also implemented one prototype for a suggestion strategy as described in Section 4.6. The key idea behind such suggestion strategies is that some plausibility rules are automatically suggested by VISPLORE. Whenever the user selects some data columns and wants to create some plausibility rules, the configured suggestion strategies are executed and all matching plausibility rules are suggested.

We have implemented only one type of suggestion strategy. The implemented strategy is based on a pattern matching algorithm and motivated by some properties of the data which we used for evaluation. We have noticed that, e.g., for a data column "Temperature", we have created checks named "TemperatureLowerLimit" and "TemperatureUpperLimit". To provide rule suggestions based on these names, we have implemented the following suggestion strategy:

A plausibility rule which tests a data column $C \in selected_columns$ using check strategy $S \in available_check_strategies$ is suggested, if C matches P_{column_name} , S matches P_{check_name} and the

ratio of values rated as implausible using this rule would result in a value between P_{min_range} and P_{max_range} . More technically, the strategy accepts multiple parameters:

- Name of data column as string P_{column_name}
The name of the data column, which must be matched to be a valid suggested rule. Wildcards can be used.
- Name of data check as string P_{check_name}
The name of the data check, which must be matched to be a valid suggested rule. Wildcards can be used.
- The name of the imputation strategy $P_{default_imputation_name}$
The parameter $P_{default_imputation_name}$ defines the imputation strategy which should be set by default for rules created by this suggestion strategy.
- Minimum rate P_{min_range} of not plausible values as value $\in [0, 1]$
If a rule evaluates less values as implausible, it is not suggested.
- Maximum rate P_{max_range} of not plausible values as value $\in [0, 1]$
If a rule evaluates more values as implausible, it is not suggested.

The idea behind P_{min_range} and P_{max_range} is that data may contain data quality issues, but, for example, if a rule evaluates 90% of the data values as implausible, then it may be an implausible plausibility rule instead of implausible values.

For example, to suggest a plausibility rule which tests for missing values using the data check *MissingCheck* for all data columns, the following parameters must be set: $P_{column_name} = "*" , P_{check_name} = "MissingCheck" , P_{min_range} = 0.0, P_{max_range} = 1.0$.

7.1.4 Default Rule Set

Our implementation of plausibility rules requires the definition of some data checks and imputation strategies before any plausibility rules can be used or imputations can be executed. For convenience, we have defined a default rule set. This rule set is loaded automatically whenever a new session is created in VISPLORE and is meant to provide basic data checks and especially imputation strategies, which then can be used for instant imputation as described in Section 7.3.3. It includes the following data checks and imputation strategies:

- Data Check: CheckForMissing
- Data Check: IsNotNegative
- Imputation Strategy: SetToMissing
- Imputation Strategy: Interpolate

7.2 Configuring Available Data Checks, Imputation and Suggestion Strategies

To configure the available data checks and imputation strategies, we have integrated a graphical user interface into VISPLORE. This interface, as it can be seen in Figure 7.1, consists of two main parts:

- **Create, copy and delete data checks, imputation and suggestion strategies**

As seen in Figure 7.1a, the user can create new objects either by clicking on the "Add"-button and choosing an available type of the data check or imputation strategy, or by selecting an existing object and clicking on the "Copy"-button which creates an object of the same type and with the same parameters. Furthermore, an existing data check or imputation strategy which is currently not used by a plausibility rule, can be deleted by clicking on "Delete".

- **Configure parameterizations**

For each created object, the name and available parameters can be configured (see Figure 7.1b).

The configured data check and imputation strategies then can be used to define plausibility rules. Unlike those, the suggestion strategies are used in a different context. They are used as soon as a data column is selected and the user wants to create rules for this data column. The resulting suggestions are proposed to the user, who can refine these suggestions and add additional rules, as further described in the following section.

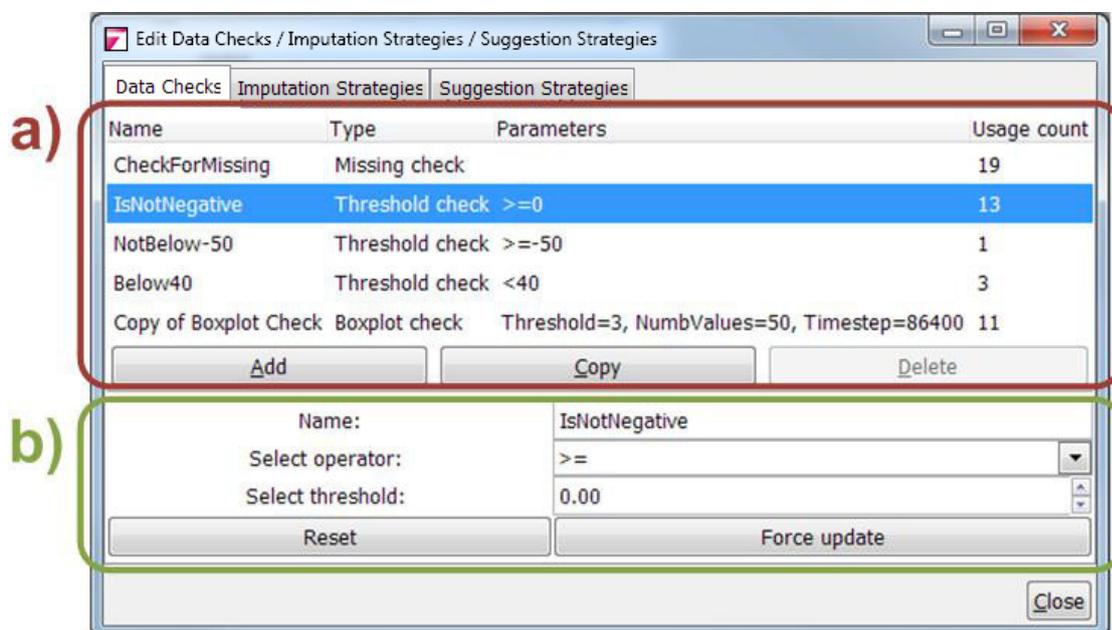


Figure 7.1: The configuration of data checks, imputation and suggestion strategies. **a)** lists the current configured data checks together with their names, types, parameters and usage counts. New data checks can be added or copied and unused data checks can be deleted. In area **b)**, the parameters of a data check can be changed by the user.

7.3 Creating Rules

To create plausibility rules in VISPLORE, the user has different possibilities. He can create rules manually by selecting data columns and define rules for them by choosing a responsible data check and a responsible imputation strategy. Furthermore, he can use the feature selection provided by VISPLORE to create rules based on the current selection and a hint by the user (e.g., "the selected entries are outliers"). In addition, he can impute entries immediately. The following sections describe these possibilities in detail.

7.3.1 Create Rules Manually

The user can select one or more data columns and press the provided button "create rule", which opens an interface to define plausibility rules for the selected data columns. Depending on the configured suggestion strategies, some rules are automatically suggested (see Figure 8.1).

In addition to the provided suggestions, he can add plausibility rules by choosing a responsible data check together with a responsible imputation strategy. The plausibility rule is created immediately and the result of the evaluation can be inspected in the view (see Section 5 and Section 7.4 for details).

7.3.2 Create Rules Automatically by Entry Selection and User Hint

We have also implemented a way to define rules using a basic artificial intelligence. The underlying idea is that in some cases, the user himself detects implausible values and wants the system to find entries, which are implausible in a similar way. To address this use case, we have implemented the feature as following:

The user can use the current feature selection in VISPLORE and press a button "Mark entries as suspicious". This opens a dialog which asks the user to give a hint why those values are suspicious. Using this information, a rule is created which rates the selected values and maybe also other values as implausible. To achieve this, the given hint is used to determinate the type of the needed data check.

We have implemented a prototyped intelligence for two types of data checks, which uses the selected entries as input and tries to find a matching parameterization of the data check, so that the plausibility evaluation rates at least the selected values, and maybe others, as implausible. These data checks are as follows:

- **Duplicates**

For this data check, the parameterization is derived by an evaluation of the selection. For example, if there are the values [4, 2, 3, 3, 3, 3] and the user selects the last two values, the data check is parameterized so that 3 duplicates are plausible but 4 duplicates are rated as implausible.

- **Outliers**

For this data check, the parameterization is done by a brute-force algorithm until the evaluation result matches the selection.

The key idea of this feature is to show the possibility that rules can be derived from a selection. However, the implemented algorithms which try to create the corresponding data checks are very basic and just for demonstration purposes. Possible future work considering automatic rule derivation is discussed in Section 9.3.

7.3.3 Instant Data Imputation

Another use case we have considered is that the user detects implausible values while analyzing the data, which he wants to correct immediately, without the need to define a rule first. To address this use case, we have implemented a feature called "Impute entries". Applied on some selected entries, those entries are instantly imputed by the imputation strategy as chosen by the user. Internally, this feature creates a temporary rule which evaluates the selected entries as not plausible and uses the selected imputation strategy. The rule is applied and deleted afterwards.

7.4 The Rule Manager View

As described in Chapter 5, we have designed three different overviews to get a summary of the created plausibility rules, with the possibility to go from the summary to the detail. In VISPLORE, those overviews are implemented together in a new view, as shown in Figure 7.2.

The view does not only provide insight into the currently configured set of plausibility rules, it is also the central point where existing plausibility rules can be manipulated. For this reason, the view provides the following actions:

- **Open view to inspect not plausible values in detail**

An important possibility to go from the overview to the detail is a detailed inspection of a plausibility evaluation. While the visualization of a rule in the view provides only an overview, the detailed inspection allows the user to analyze the actual data values underlying the plausibility check in the context of the other data. As illustrated in Figure 7.3, a view showing the data column underlying the check as time series is opened.

Moreover, the current selection is not static but is automatically updated as soon as the evaluation result changes. This gives the user the possibility to edit parameters of a used check strategy with an instant visual response concerning which entries are now rated as not plausible.

- **Apply rules** (execute the imputation)

A main aspect when using plausibility rules, is the possibility to impute values which are rated as not plausible. To trigger that imputation, selected rules can be "executed", which means that for each rule, the responsible imputation strategy (see Section 4.5) is called to impute the implausible values.

- **Change parameters of used data checks / imputation strategies for rules**

The user can also edit parameters of a rule respectively of the used data check or imputation strategy. If a data check or imputation strategy is used by other rules as well, changing its parameters would also affect the other rules. Hence, if the user wants to edit the parameters of a data check or imputation strategy, the user is asked if he wants to edit the data check or imputation strategy and affect all other rules or if he wants to clone it and edit the parameters of the clone, which is then only used by the current rule. Either way, a graphical user interface as seen in 7.1, is opened.

- **Replace used data checks / imputation strategies for rules**

The data check which is used by a rule can be replaced through a submenu. In this submenu, all available data checks are listed, and a simple click replaces the current check. The same holds for the imputation strategies, which can be replaced by selecting a new strategy from a submenu.

- **Add and remove custom tags**

As described in Section 5.3, rules can be grouped by using custom tags. The user can add custom tags to the selected rules using the available popup menu and assigned tags can also be removed.

- **Clone existing rules**

If only a single rule is selected, the popup menu provides the button to clone the rule. The user then can select multiple other data columns as target and then the rule is cloned for each data column.

- **Remove rules**

By this option, all selected rules are removed. If data values were already edited by using these rules, the edited values stay unchanged.

- **Use as partitioner**

As described in Section 5.2.2, rules can be also used as partitioner for the data overview and rule-based overview. To use a rule as partitioner, an option is available in the popup menu.

- **Showing detail information on demand**

As described in Section 5.4, the rule-based overview is designed to provide an overview of the configured plausibility rules. While the view itself provides the overview, detail information is provided as soon as the user hovers a plausibility rule. A tooltip displays the detail information for this plausibility rule, which contains the underlying data column, the used data check, the defined imputation strategy and the relative and absolute number of implausible values.

- **Drill down on partitioner group**

If partitioners (see Section 5.2.2) are used, the user can drill down on a group to get a better insight into the distribution of implausible entries within this group. Drilling-down can be activated by right-clicking on a group and hides all other groups of the corresponding partitioner. If multiple partitioners are used, the nested groups are not affected but on the upper layers all groups which did not contain the selected group are hidden. To undo this action and display all groups of the partitioners, the user can middle-click anywhere in the partitioners legend. Figure 7.4 illustrates the steps of drilling down on a partitioner.

7.5 Inspecting Imputed Data

When data is edited by applying a plausibility rule, an important aspect for usability is the possibility to trace the applied changes. To provide the necessary data which is needed to trace these changes, additional meta columns are created or updated when a rule is applied, as described in Section 6.3. In VISPLORE, these data columns can be organized hierarchically, so that a data column can have a parent column and child columns. These meta columns are created as child columns of the original data column. If a data column E is edited by executing the imputation strategy, the original values are stored in those meta columns.

VISPLORE provides different techniques to deal with modified data values. First, if an edited data column is mapped to a view, an additional widget, which is shown in Figure 7.5, shows the ratio of valid, imputed and missing values. All values which are original values and not missing, count as valid. All values which are imputed and not missing after the imputation, count as imputed. Values which are missing or set to be missing by the imputation, count as missing. This widget thus provides a quick information to which degree a certain data column has been affected by imputation. Hovering the visual representation of valid, imputed, or missing entries in the widget immediately highlights the respective data values in all views using VISPLORE's ad-hoc selection called *Focus* (see Section 3.3.1).

The second technique to inspect imputed data is provided as soon as the user selects a modified data column in the data manager. In this case, VISPLORE offers shortcuts to specific view parameterizations, which are designed to provide a detailed comparison of original and imputed data. These shortcuts are:

- **Compare original data column with modified data column**

This shortcut opens a visual representation, which is designed to compare the original data column with the modified data column. The data column which is set to define the order of the entries is mapped on the X axis, while the original values are mapped on the Y axis in one color, together with the modified values in another color. Figure 7.6 shows an example.

- **Show modification history**

Using this shortcut, a tabular data representation shows the modification history together with the current values and the original values. The modification history contains every modification applied on a value and provides the user advanced insight to determinate why values were replaced and which objects were responsible for that.

Because the meta data about data imputation is stored in data columns the user may employ all available views and features of VISPLORE to analyze these data columns like he would do with other data.

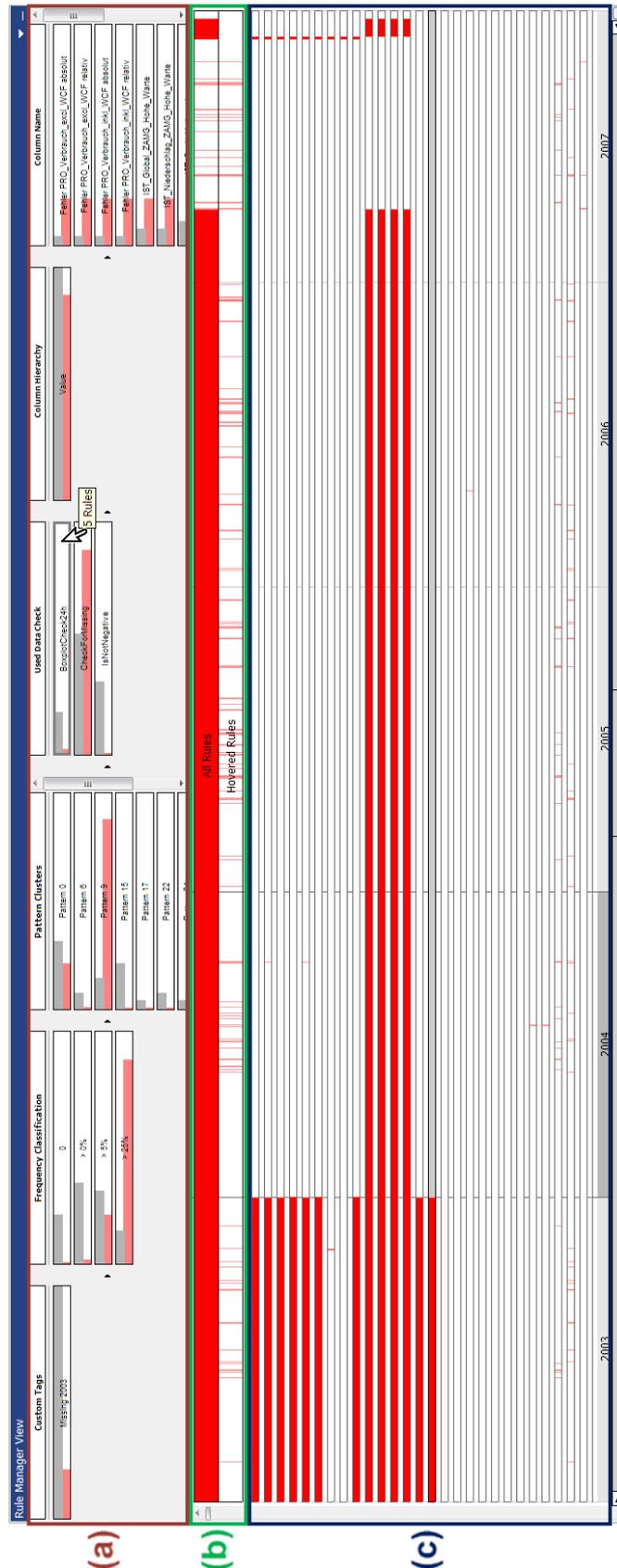


Figure 7.2: This screenshot shows the rule manager view with many plausibility rules loaded. **(a)** The group-based overview allows the user to select subsets of rules. **(b)** The data-based overview displays all not plausible entries with respect to all rules and all currently selected rules or all rules within a group, if a group is currently hovered. **(c)** The rule-based overview shows implausible entries separately for each rule.

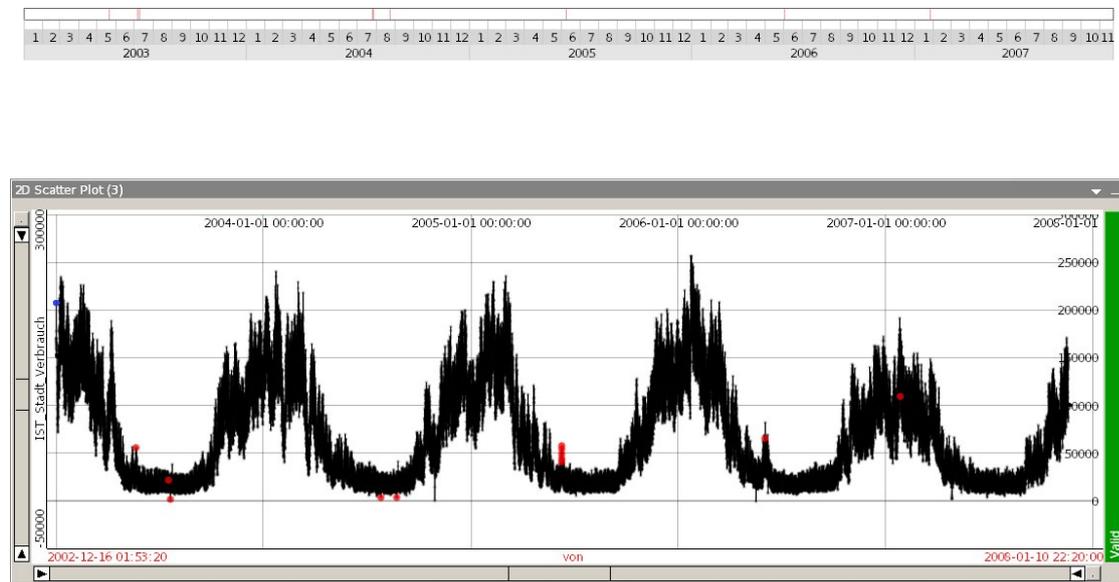


Figure 7.3: This image shows a rule in the rule-based overview and the detailed visualization in a scatter-plot.

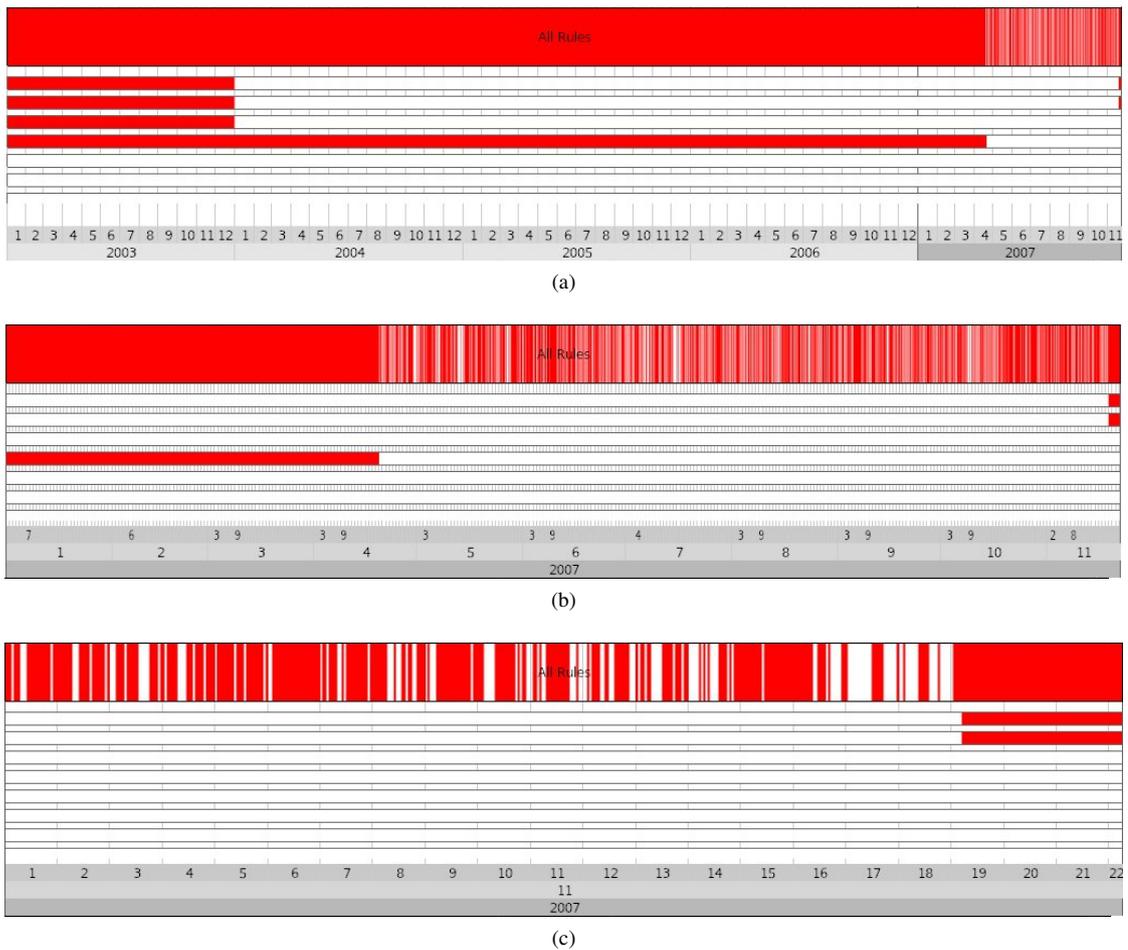


Figure 7.4: Drilling down on partitioner groups. (a) All groups are shown. (b) The user right-clicks on the year 2007 and all other years are hidden. (c) The user right-clicks on the month 11 and all other months are hidden.

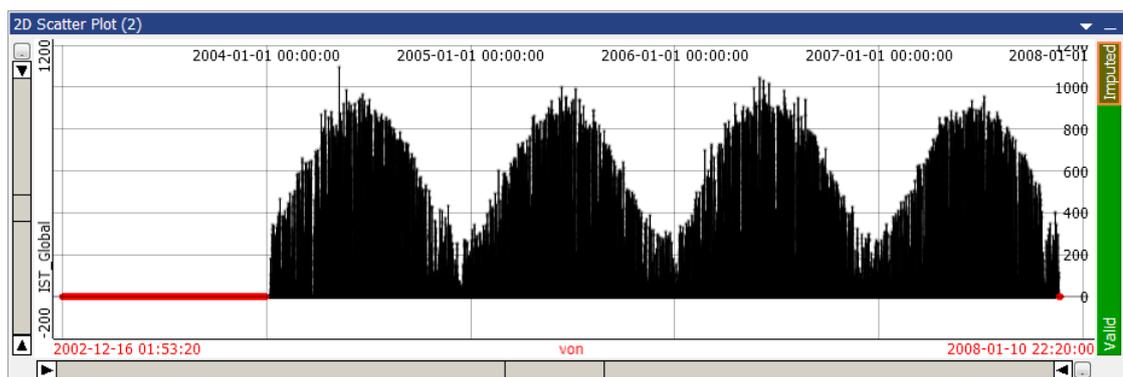


Figure 7.5: When data is edited by applying a plausibility rule, an additional widget which can be seen on the right side, provides fast interactive information about the ratio of missing, imputed and valid values.

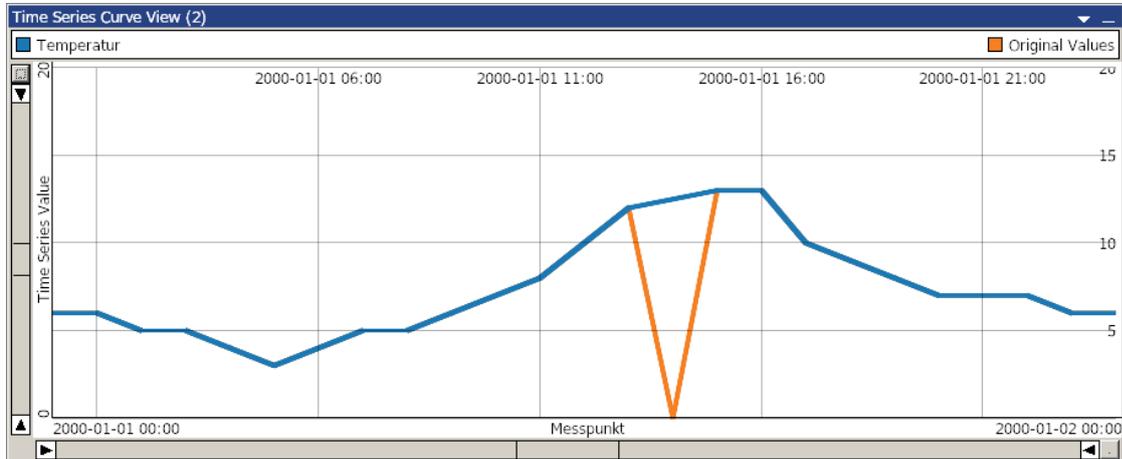


Figure 7.6: In this view parameterization, original values can be compared with modified values. The modified values are drawn in blue while the original values are shown in orange.

Mark	Messpunkt (Uhr)	Temperatur (°C)	Original Values	Modification History
<input type="checkbox"/>	2000-01-01 07:00:00	5	5	
<input type="checkbox"/>	2000-01-01 08:00:00	5	5	
<input type="checkbox"/>	2000-01-01 09:00:00	6	6	
<input type="checkbox"/>	2000-01-01 10:00:00	7	7	
<input type="checkbox"/>	2000-01-01 11:00:00	8	8	
<input type="checkbox"/>	2000-01-01 12:00:00	10	10	
<input type="checkbox"/>	2000-01-01 13:00:00	12	12	
<input type="checkbox"/>	2000-01-01 14:00:00	12	0	0 -> 12 ("TestName" failed, executed "LinearInterpolation")
<input type="checkbox"/>	2000-01-01 15:00:00	13	13	
<input type="checkbox"/>	2000-01-01 16:00:00	13	13	
<input type="checkbox"/>	2000-01-01 17:00:00	10	10	
<input type="checkbox"/>	2000-01-01 18:00:00	9	9	
<input type="checkbox"/>	2000-01-01 19:00:00	8	8	

Figure 7.7: The detailed modification history can be inspected using a tabular view as shown above.

Case Study

In this section, a real world data set is used to demonstrate two use cases how our developed framework can be employed to detect and impute data quality issues.

The used data set deals with gas consumption and is described in detail in section 8.1. Section 8.2.1 describes a use case which assumes users of two different types, a content specialist and a data analyst. Section 8.2.2 then describes a use case without plausibility rules.

8.1 The Data

The data set contains time-series describing the natural gas consumption of a large city together with additional data attributes like meteorological and calendric aspects. It is a real-world data set from a European city which consists of 42870 data rows and includes data from 01-01-2003 until 22-11-2007. We were restricted to anonymize it for this thesis, so we have removed the name of the city from all data attributes.

The included data attributes can be grouped into different groups:

1. **Temporal information**
2. **Measurements**
3. **Predictions**

The data set also includes predictions for the measured attributes, which are based on Multivariate Adaptive Regression Splines (MARS) [Fri91].

4. **Derived attributes**

In more detail, the following data attributes exist:

1. **Temporal information**

- **von** (YYYY-MM-DD HH:MM, e.g., 2007-12-31 08:00)
Describes the starting time for the corresponding data row.
- **bis** (YYYY-MM-DD HH:MM, e.g., 2007-12-31 09:00)
Describes the end time for the corresponding data row.

- **Feiertag** (categorical values ["ja", "nein"])
Describes if the corresponding day was a public holiday ("ja") or not ("nein").
- **Wochentag** (categorical values ["Montag", "Dienstag", "Mittwoch", "Donnerstag", "Freitag", "Samstag", "Sonntag"])
Describes the day of the week for the corresponding timestamps.

2. Measurements

- **IST_Global**
Describes the measured global irradiance.
- **IST_Niederschlag**
Describes the measured precipitation in millimeters per hour.
- **IST_Stadt_Verbrauch**
This data column describes the total gas consumption of the city.
- **IST_Temp**
This data column describes the temperature in degree Celsius.
- **IST_Windgeschw**
This data column describes the measured wind speed in km/h.

3. Predictions

- **PROG_Global**
This data column describes a forecast for the global irradiance.
- **PROG_Niederschlag**
This data column describes a forecast for the precipitation.
- **PROG_Temp**
This data column contains a forecast for the measured temperature.
- **PROG_Windgeschw**
This data column contains a forecast for the measured wind speed.
- **PRO_Verbrauch_excl_WCF**
This data column contains a forecast for the gas consumption without considering the wind chill factor, which is a perceived decrease in air temperature on exposed skin due to the wind.
- **PRO_Verbrauch_inkl_WCF**
This data column contains a forecast for the gas consumption, including the effects of the wind chill factor, which is a perceived decrease in air temperature on exposed skin due to the wind.

4. Derived attributes

- **Fehler PRO_Verbrauch_excl_WCF absolut**
The absolute error between the predicted value for the global gas consumption (without considering the wind chill factor) and the measured global gas consumption.
- **Fehler PRO_Verbrauch_excl_WCF relativ**
The relative error between the predicted value for the global gas consumption (without considering the wind chill factor) and the measured global gas consumption.
- **Fehler PRO_Verbrauch_inkl_WCF absolut**
The absolute error between the predicted value for the global gas consumption (considering the wind chill factor) and the measured global gas consumption.

- **Fehler PRO_Verbrauch_inkl_WCF relativ**

The relative error between the predicted value for the global gas consumption (considering the wind chill factor) and the measured global gas consumption.

8.2 Use Case 1: Content Specialist + Data Analyst

In this section, we want to demonstrate a use case as it would be done by two different experts, a content specialist and a data analyst. The content specialist knows constraints and properties of the data set and prepares the defined data checks and imputation strategies, which then are used and refined by the data analyst to analyze the data set. The data set can be a similar set (e.g., the same data but for different years), but in this concrete case, the same data set is used.

8.2.1 Step 1: Content Specialist Creates Data Checks and Imputation Strategies

The first step is the creation and configuration of data checks, imputation strategies and suggestion strategies, which is done by the content specialist. While the data checks and imputation strategies are created to be used by the plausibility rules, the suggestion strategies define which plausibility rules are suggested by VISPLORE when the user wants to create rules. To create the right data checks and imputation strategies, the content specialist analyses the data and inspects the value ranges, the context and typical restrictions.

In this use case, we act as the content specialist and create data checks based on the following analysis of the data attributes:

- **IST_Global**

Based on the definition of this attribute, we can assume that all values are greater or equal to 0. Therefore, we create an instance of a threshold data check named "NotNegative", which tests each value against ≥ 0 . Furthermore, similarities should be cyclic within 24 hours, that means that the measured global horizontal irradiation at, e.g., 2005-01-01 00:00 is typically similar to the measured global horizontal irradiation at 2005-01-02 00:00, but may differ heavily from the measured value at 2005-01-01 12:00. Therefore, we create an instance of an outlier data check named "Outlier_24h" to determinate not plausible outliers based on an evaluation of values at the same time of each day (e.g., the data check compares 2005-02-02 00:00 with 2005-01-31 00:00 and 2005-02-03 00:00).

- **IST_Niederschlag**

For the precipitation, we can assume that the values are greater or equal to 0. A corresponding data check already exists, so we do not have to create a new one. Additionally, based on the contextual information of the content specialist, which knows that the yearly average of the city is around 500-700 mm, an upper limit of 500 mm (per hour) can be safely assumed. So we create a new instance of a threshold check called "Niederschlag < 500 mm", which is parameterized accordingly.

- **IST_Stadt_Verbrauch**

For the global gas consumption, we can also safely assume that the values are not negative. A matching data check already exists. Furthermore, we can assume that there are no large differences in gas consumption within one hour, so we can create an instance of an outlier check, which uses time-based predecessors and successors to calculate the quartiles and does an outlier-detection based on these quartiles. The concrete parametrization (number of predecessors and successors, outlier-threshold) is based on a rough guess (according to Section 1.5) and can be fine-tuned later. Additionally, the gas consumption can be seen as cyclic like the global irradiance and a matching instance of an outlier data check already exists ("Outlier_24h").

- **IST_Temp**
Based on the previous extreme values for the temperature of the city ($-26.3^{\circ}C$ and $38.3^{\circ}C$), we can assume two limits: The temperature is typically above $-27^{\circ}C$ and below $40^{\circ}C$. Therefore we create two instances of threshold data checks ("TempLowerLimit", "TempUpperLimit"), which are parametrized accordingly. Based on the cyclic (daily) behaviour of the temperature, we can also use the already created outlier check "Outlier_24h".
- **IST_Windgeschw**
For the wind speed values, we can assume that they are greater or equal to 0. A matching instance of a threshold check already exists.
- **PROG_Global**
PROG_Niederschlag
PROG_Temp
PROG_Windgeschw
PRO_Verbrauch_excl_WCF
PRO_Verbrauch_inkl_WCF
See the corresponding measurement columns IST_Global, IST_Niederschlag, IST_Temp, IST_Windgeschw, and IST_Stadt_Verbrauch.
- **Fehler PRO_Verbrauch_excl_WCF absolut**
Fehler PRO_Verbrauch_excl_WCF relativ
Fehler PRO_Verbrauch_inkl_WCF absolut
Fehler PRO_Verbrauch_inkl_WCF relativ
These data attributes actually depend on other attributes. Therefore, we do not create specific data checks for this attributes, however, we can check for missing values.

To conclude, the analysis and preparation step leads to the following data checks, which were instantiated and configured by the content specialist:

- **CheckForMissing**
Can be used for all data attributes to detect missing values.
- **NotNegative** (threshold check, ≥ 0) Can be used for attributes like wind speed and temperature, which cannot be negative i
- **Niederschlag_<500mm** (threshold check, < 500)
- **TempLowerLimit** (threshold check, > -27)
- **TempUpperLimit** (threshold check, < 40)
- **Outlier_24h** (outlier check, timestep=24h)
- **Outlier_1h** (outlier check, timestep=1h)

To provide some simple imputation methods, we also instantiate the following imputation strategies:

- **"SetTo0"** (SetToValue imputation strategy, value=0)
This imputation strategy sets all not plausible values to 0.
- **"SetToMissing"** (SetToMissing imputation strategy)
This imputation strategy sets all not plausible values to *MISSING*.

- **"Interpolation"** (Interpolate imputation strategy)
This imputation strategy interpolates all implausible values with respect to the nearest predecessor and nearest successor.

To suggest some rules automatically, the content specialist also defines some instances of suggestion strategies:

- **MissingTrigger**
This suggestion strategy is created so that a plausibility rule which tests for missing data values is automatically suggested for each data column.
- **TempTrigger**
This suggestion strategy is created so that for each data column which represents temperature values, plausibility rules to test the defined constraints (data checks "TempLowerLimit" and "TempUpperLimit") are automatically suggested.
- **NiederschlagTrigger**
This suggestion strategy is created so that for each data column which represents precipitation values, a plausibility rule which tests the defined constraints (data check "Niederschlag < 500 mm") are automatically suggested.

Listing 8.1 shows the resulting rule set as XML file, which is given to the data analyst. This rule set does not contain any defined plausibility rules - it only includes predefined data checks and imputation strategies. In this use case, the plausibility rules are then created by the data analyst in the next step.

Listing 8.1: The exported rule set as predefined by the content specialist

```

1 <ruleset>
2 <checks>
3   <missing_check name="s:CheckForMissing" />
4   <threshold CompareOperator="i:2" Threshold="f:0.000000" name="s:NotNegative" />
5   <threshold CompareOperator="i:1" Threshold="f:-27.000000" name="s:TempLowerLimit" />
6   <threshold CompareOperator="i:-1" Threshold="f:40.000000" name="s:TempUpperLimit" />
7   <outlier_check NumUsedValues="i:8" Timestep="i:86400" UseTimestep="b:t"
8     WhiskerFactor="d:2.000000" name="s:Outlier_24h" />
9   <outlier_check NumUsedValues="i:4" Timestep="i:3600" UseTimestep="b:t"
10    WhiskerFactor="d:2.000000" name="s:Outlier_1h" />
11   <threshold CompareOperator="i:-1" Threshold="f:500.000000"
12     name="s:Niederschlag <500" />
13 </checks>
14 <correctors>
15   <interpolation_corrector name="s:Interpolation" />
16   <set_to_missing Mode="i:1" Value="d:0.000000" name="s:SetTo0" />
17   <set_to_missing Mode="i:0" name="s:SetToMissing" />
18 </correctors>
19 <suggestions>
20   <fault_range channel="s:*" check="s:CheckForMissing" corrector="s:Interpolation"
21     hint="s:MissingTrigger" max_le="f:1.000000" min_gt="f:0.000000" />
22   <fault_range channel="s:*Temp*" check="s:*Temp*" corrector="s:" hint="s:TempTrigger"
23     max_le="f:1.000000" min_ge="f:0.000000" />
24   <fault_range channel="s:*Niederschlag*" check="s:Niederschlag*" corrector="s:"
25     hint="s:NiederschlagTrigger" max_le="f:1.000000" min_ge="f:0.000000" />
26 </suggestions>
27 </ruleset>

```

Data Column	Created Rules
IST_Global	CheckForMissing -> Interpolation
IST_Niederschlag	CheckForMissing -> Interpolation Niederschlag < 500 mm -> Interpolation
IST_Stadt_Verbrauch	CheckForMissing -> Interpolation
IST_Temp	CheckForMissing -> Interpolation TempLowerLimit -> Interpolation TempUpperLimit -> Interpolation
IST_Windgeschw	CheckForMissing -> Interpolation
PROG_Global	CheckForMissing -> Interpolation
PROG_Niederschlag	CheckForMissing -> Interpolation Niederschlag < 500 mm -> Interpolation
PROG_Temp	CheckForMissing -> Interpolation TempLowerLimit -> Interpolation TempUpperLimit -> Interpolation
PROG_Windgeschw	CheckForMissing -> Interpolation
PRO_Verbrauch_excl_WCF	CheckForMissing -> Interpolation
PRO_Verbrauch_inkl_WCF	CheckForMissing -> Interpolation
Fehler PRO_Verbrauch_excl_WCF absolut	CheckForMissing -> Interpolation
Fehler PRO_Verbrauch_excl_WCF relativ	CheckForMissing -> Interpolation
Fehler PRO_Verbrauch_inkl_WCF absolut	CheckForMissing -> Interpolation
Fehler PRO_Verbrauch_inkl_WCF relativ	CheckForMissing -> Interpolation

Table 8.1: The suggested set of plausibility rules.

8.2.2 Step 2: Data Analyst Creates Plausibility Rules

After the preparation step, the data analyst can use the rule set to analyze the same data or data with a similar structure (e.g., the same data set but for different years). However, in this use case, the data analyst simply uses the same data set.

As above, we act as the data analyst. We import the data set to VISPLORE and load the prepared rule set from an XML file (see Listing 8.1). Afterwards, we have to define which data column contains the time information to define the ordering. We use the data column "von" as time column. Then we select all continuous data columns from the data manager, and click on "Create Rule". The rule creation dialog shows up and already contains some suggested rules (see Figure 8.1). Table 8.1 shows all rules which are automatically suggested by VISPLORE, based on the configured data checks and suggestion strategies. We now add additional rules, to cover the constraints which are not handled by the suggested rules. We click on "Add" (see Figure 8.1) and select the wanted data checks and imputation strategies, based on the meaning of the data attributes as discussed in Section 8.2.1. Table 8.2 shows the final set of plausibility rules which are specified for the mentioned data columns. All together, we now have 35 defined rules.

8.2.3 Step 3: Data Analyst Uses Plausibility Rules

We now want to evaluate the rules and get an overview of the quality issues in the loaded data set. We use the provided overviews to get insight in the structure of implausible values.

On the data-based overview (see Figure 8.2a), we can see that many entries are rated as not plausible, especially until 04-2007. Using the group-based overview (see Figure 8.2b), we then notice that the data check "CheckForMissing" is responsible for many entries rated as not plausible, so we first select this group to analyze the rules which use this data check. Now, only plausibility rules which

Data Column	Created Rules
IST_Global	CheckForMissing -> Interpolation Outlier_24h -> Interpolation NotNegative -> SetTo0
IST_Niederschlag	CheckForMissing -> Interpolation Niederschlag < 500 mm -> Interpolation NotNegative -> SetTo0
IST_Stadt_Verbrauch	CheckForMissing -> Interpolation NotNegative -> SetTo0 Outlier_24h -> Interpolation
IST_Temp	CheckForMissing -> Interpolation TempLowerLimit -> Interpolation TempUpperLimit -> Interpolation
IST_Windgeschw	CheckForMissing -> Interpolation NotNegative -> SetTo0
PROG_Global	CheckForMissing -> Interpolation Outlier_24h -> Interpolation NotNegative -> SetTo0
PROG_Niederschlag	CheckForMissing -> Interpolation Niederschlag < 500 mm -> Interpolation NotNegative -> SetTo0
PROG_Temp	CheckForMissing -> Interpolation TempLowerLimit -> Interpolation TempUpperLimit -> Interpolation
PROG_Windgeschw	CheckForMissing -> Interpolation NotNegative -> SetTo0
PRO_Verbrauch_excl_WCF	CheckForMissing -> Interpolation Outlier_24h -> Interpolation NotNegative -> SetTo0
PRO_Verbrauch_inkl_WCF	CheckForMissing -> Interpolation Outlier_24h -> Interpolation NotNegative -> SetTo0
Fehler PRO_Verbrauch_excl_WCF absolut	CheckForMissing -> Interpolation
Fehler PRO_Verbrauch_excl_WCF relativ	CheckForMissing -> Interpolation
Fehler PRO_Verbrauch_inkl_WCF absolut	CheckForMissing -> Interpolation
Fehler PRO_Verbrauch_inkl_WCF relativ	CheckForMissing -> Interpolation

Table 8.2: The final set of plausibility rules.

use the `CheckForMissing` to evaluate values, are shown in the rule-based overview. In the rule-based overview, we can hover the shown rules to get detailed information (see Figure 8.3). We recognize that the measurements in the data columns "IST_Global", "IST_Niederschlag", "IST_Windgeschwindigkeit", "PRO_Verbrauch_excl_WCF" and "PRO_Verbrauch_inkl_WCP" are only available since 2004-01-01 00:00. Hence, values for the derived attributes "Fehler_PRO_Verbrauch_incl_WCP_relativ", "Fehler_PRO_Verbrauch_excl_WCP_absolut", "Fehler_PRO_Verbrauch_excl_WCP_relativ" and "Fehler_PRO_Verbrauch_incl_WCP_absolut" are naturally also missing for the year 2003.

While inspecting the rules, we identify additional implausible values in 10-2004 (see Figure 8.4). We use the possibility to zoom into the time categorization by right-clicking on the year 2004 followed by a right-click on the month 10. As can be seen in Figure 8.5, we further zoom into 2004-10-29 and determine that there are missing values in "Fehler_PRO_Verbrauch_incl_WCP_relativ" and "Fehler_PRO_Verbrauch_excl_WCP_absolut" on 2004-10-29 at 03:00. Because these attributes are derived attributes, we notice the implausible values but decide that we do not impute them, and continue with the analysis.

Using the middle mouse button, we reset the time axis to inspect the whole time range. We hover the available groups to see similar failure patterns and find out that rules which rate entries at 2004-01-01 00:00 as not plausible, rate also entries at 11-2007 as not plausible. We activate the group "Pattern 0" to analyze these rules in detail. We now see nine rules, which belong to the same pattern-cluster, as shown in Figure 8.6. However, their evaluation results are not totally identical and we recognize that seven rules rate values at 11-2007 as implausible. Using the tooltip which appears while hovering the rules (e.g., see Figure 8.3), we identify the affected data columns as "Fehler_PRO_Verbrauch_excl_WCF_absolut", "Fehler_PRO_Verbrauch_excl_WCF_relativ", "Fehler_PRO_Verbrauch_inkl_WCF_absolut", "Fehler_PRO_Verbrauch_inkl_WCF_relative", "IST_Global", "IST_Niederschlag" and "IST_Windgeschwindigkeit". Although these found values are rated as implausible by the defined plausibility rules, as a data analyst with background information, we know that values for this time period are missing because the corresponding sensors started measurements at 01-2004. Therefore, we do not apply any imputation strategies.

To proceed with the analysis, we clear the group selection to see all rules again. We decide that we want to analyze rules which have a small number of implausible values. To do this, we use the group "Frequency Classification > 0%" to show only rules which rate more than 0% as not plausible (and, according to the definition of the group as described in Section 5.3, not more than 1%). As shown in Figure 8.7, we realize that only rules using the "Outlier_24h", "CheckForMissing", "Niederschlag<500" and "NotNegative" data checks match the current selection. We have already analyzed the rules using "CheckForMissing", so we additionally select the groups "Outlier_24h", "Niederschlag<500" and "NotNegative" to see only the corresponding rules. We now see five rules in the rule-based overview and open the detailed representation for the first rule by double clicking on it. As it can be seen in Figure 8.8, this opens a 2D scatterplot with connected data points and we see some selected data points (red) which are rated as implausible. In the context of the other values, these values are significant outliers and additionally, they severely distort the scaling of the Y axis, so we decide that we want to impute these values. Using the rule-based overview, we select the corresponding rule, right-click and execute "Execute Rule" from the pop-up menu. The values are replaced by an interpolation of the local neighbours (with respect to the timestamp). The result can be seen in Figure 8.9. We continue with the analysis using the rule-based overview and recognize that a rule which tests the data column "IST_Stadt_Verbrauch" rates many values as implausible. We open the detailed representation. The visual representation gives no hint why these high number of values are rated as implausible and we conclude, that the used data check to detect outliers is just too strict and should be fine-tuned. We choose "Edit check parameters" from the popup menu. The edited data check is also used by some other plausibility rules, so VISPLORE asks the user if he really wants to edit that data check or if he wants to copy it and use the copy instead, which would not affect other rules. We create a copy of the data check and edit the parameters, while the result

of the evaluation is continuously updated in the detailed representation as well as in the overviews. As soon as we have found a reasonable parameterization, we close the editing window and execute the rule, which means that the implausible values are imputed using the configured imputation strategy.

Now we have different possibilities to analyze the changes made. We can select the data column in the data manager and use the provided "Compare original with modified" action. It opens a time series curve view, where one curve represents the original values and a second curve represents the modified values, as can be seen in Figure 8.10. We can also use the provided widget which shows up as soon as values are modified, which can be seen in Figure 8.11.

8.3 Use Case 2: Immediate Imputation of Implausible Values

In this section, we want to demonstrate a short use case without the use of plausibility rules. The user has loaded the data and analyzes it using the multiple views of VISPLORE. When inspecting the data column "IST_Niederschlag", he recognizes some outliers, which affect the scaling of the Y axis (see Figure 8.12(a)). To remove the outliers, he selects them using VISPLORE's brushing feature and calls "Impute selected values" from the main menu. A dialog shows up to choose the imputation strategy, which now only includes those strategies which are already defined in the default rule set (as described in Chapter 7). The user selects "SetToMissing", executes the imputation and inspects the modified data column (see Figure 8.12(b)).

8.4 Summary

To conclude, we have shown two exemplary use cases of our framework, using real world data with quality issues. We have shown the preparation step to create specific data checks and plausibility rules together with some imputation strategies. The framework can be used to detect quality issues and analyze them in detail. Missing or anomalous values can be imputed and the modified data can be compared with the original data as well.

However, the framework has some room for improvements. Especially the preparation step for the use of plausibility rules and some limitations regarding detectable quality issues should be discussed. In the following section, the results and limitations of the framework are discussed in detail together with some thoughts on possible future work.

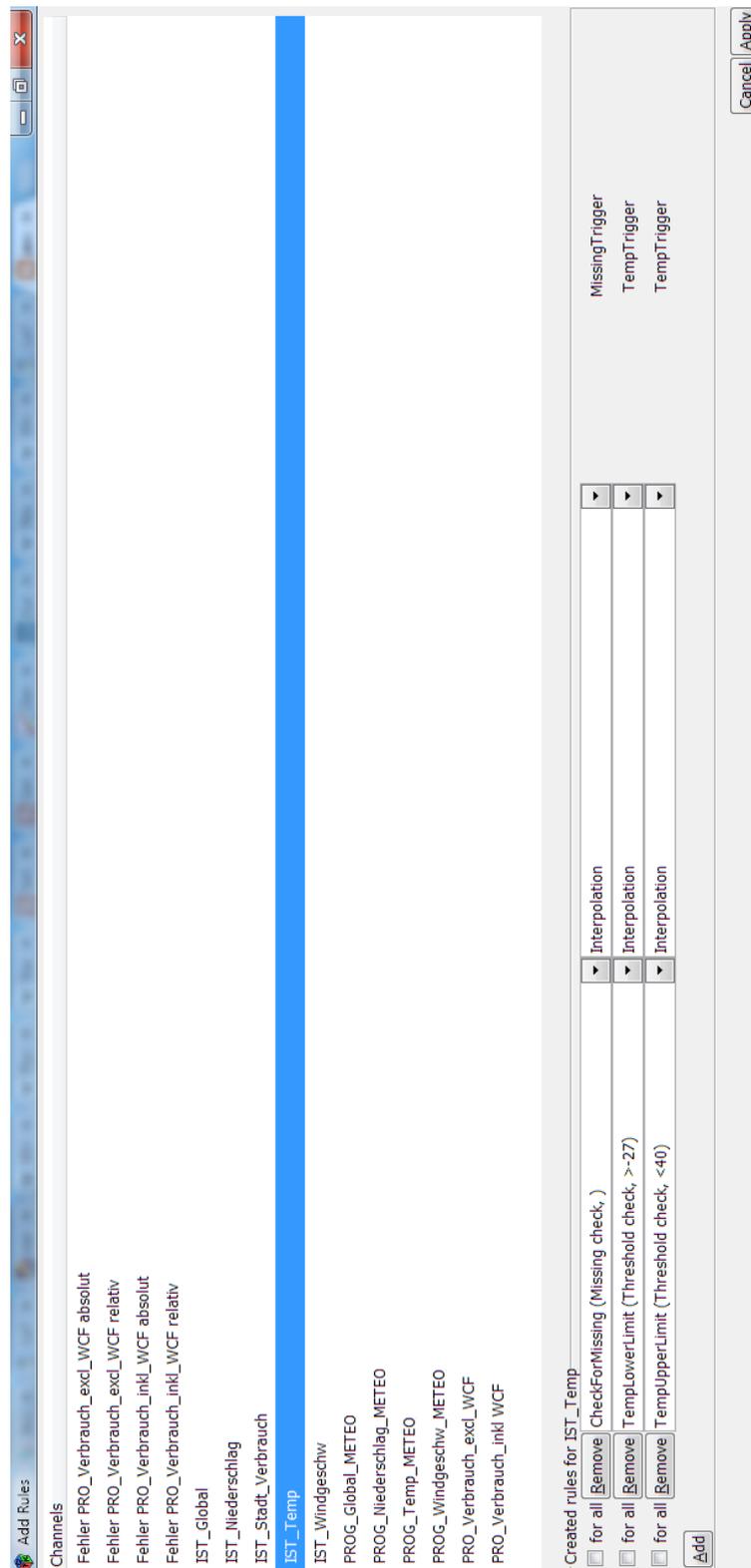


Figure 8.1: Creating new rules for the selected data columns. Note that the shown rules are automatically suggested by VISPLORE.

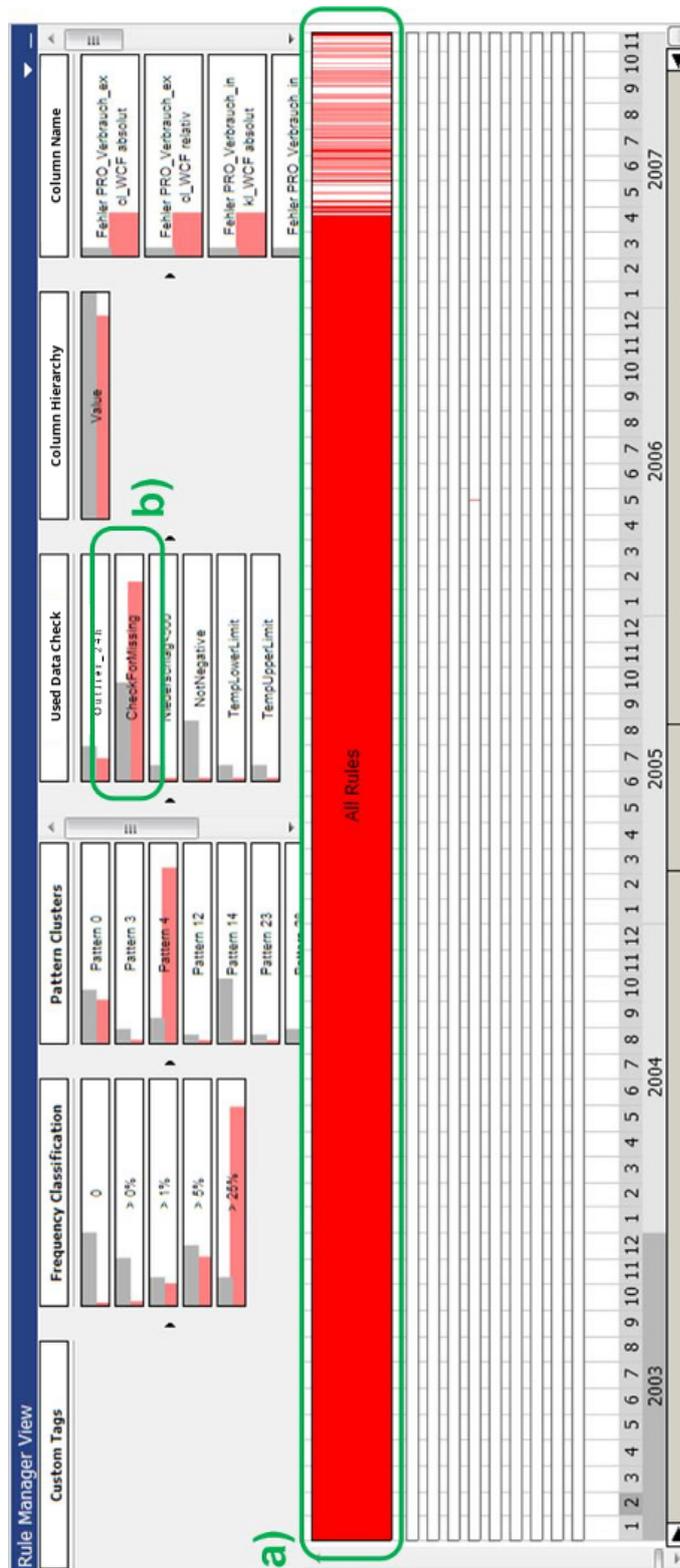


Figure 8.2: (a) Many not plausible entries appear until 04-2007. (b) The data check "CheckForMissing" is responsible for many values evaluated to not plausible.

The screenshot shows the 'Rule Manager View' interface. At the top, there are several panels: 'Custom Tags' (with 'CustomTag1' and 'Fehler1'), 'Frequency Classification' (with '0', '> 0%', '> 5%', and '> 25%'), 'Pattern Clusters' (with 'Pattern 0' through 'Pattern 27'), 'Used Data Check' (with 'Outlier_24h', 'CheckForMissing', 'NotNegative', 'TempLowerLimit', and 'TempUpperLimit'), 'Column Hierarchy' (with 'Value'), and 'Column Name' (with 'Fehler PRO_Verbrauch_inkl_WCF absolut', 'Fehler PRO_Verbrauch_inkl_WCF relativ', 'Fehler PRO_Verbrauch_inkl_WCF absolut', and 'Fehler PRO_Verbrauch_inkl_WCF absolut').

The main area displays a grid of rules. A tooltip is shown over a rule, providing the following information:

- Tested column: Fehler PRO_Verbrauch_inkl_WCF absolut
- Failed entries: 8834 (Hovered category: 744 of 744)
- Data check: CheckForMissing
- Parameters: no parameters
- Imputation strategy: Interpolation
- Parameters: no parameters
- Created: 27.11.2013 11:46
- Created by suggestion strategy: MissingTrigger
- Last modified: 27.11.2013 11:46
- Custom tags: CustomTag1
- Last executed: -
- Year of von as string: 2003
- Month of von as string: 12

The bottom of the interface shows a timeline from 2003 to 2007, with columns for each year and month.

Figure 8.3: We have now selected the Group "CheckForMissing" and see the corresponding rules. Hovering the rules shows us detail information and we recognize that the column "Fehler_PRO_Verbrauch_inkl_WCF absolut" has missing values for 2003.

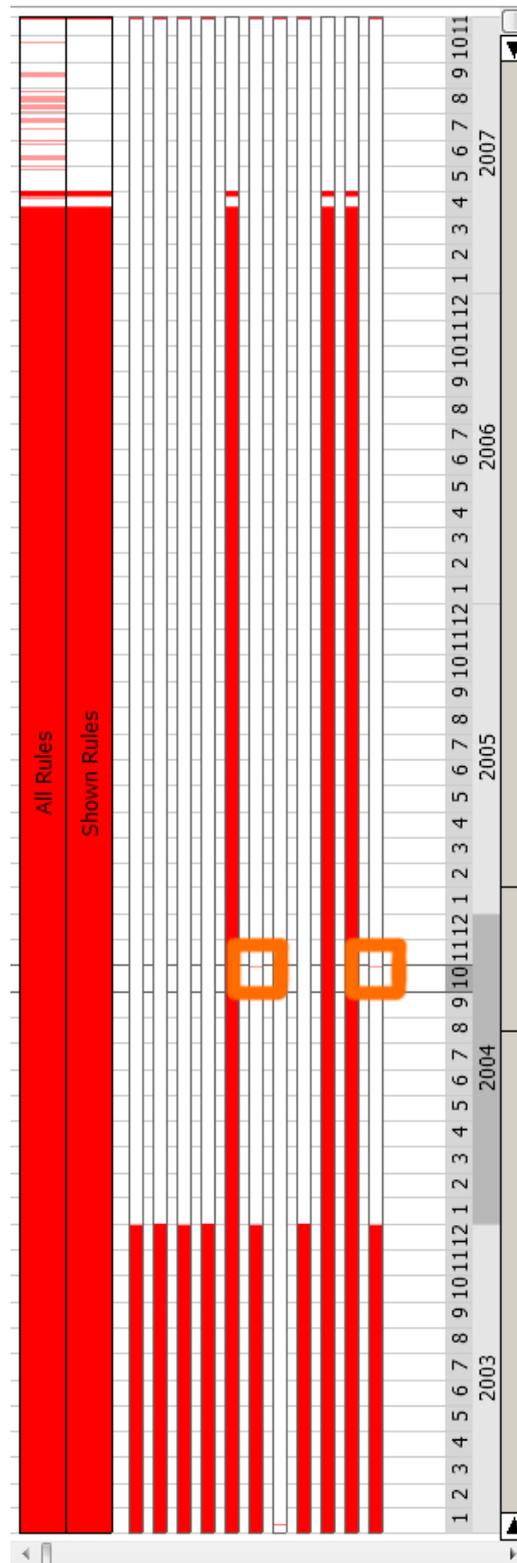


Figure 8.4: Using the rule-based overview, we identify additional implausible values around 10-2004 (highlighted by orange boxes).

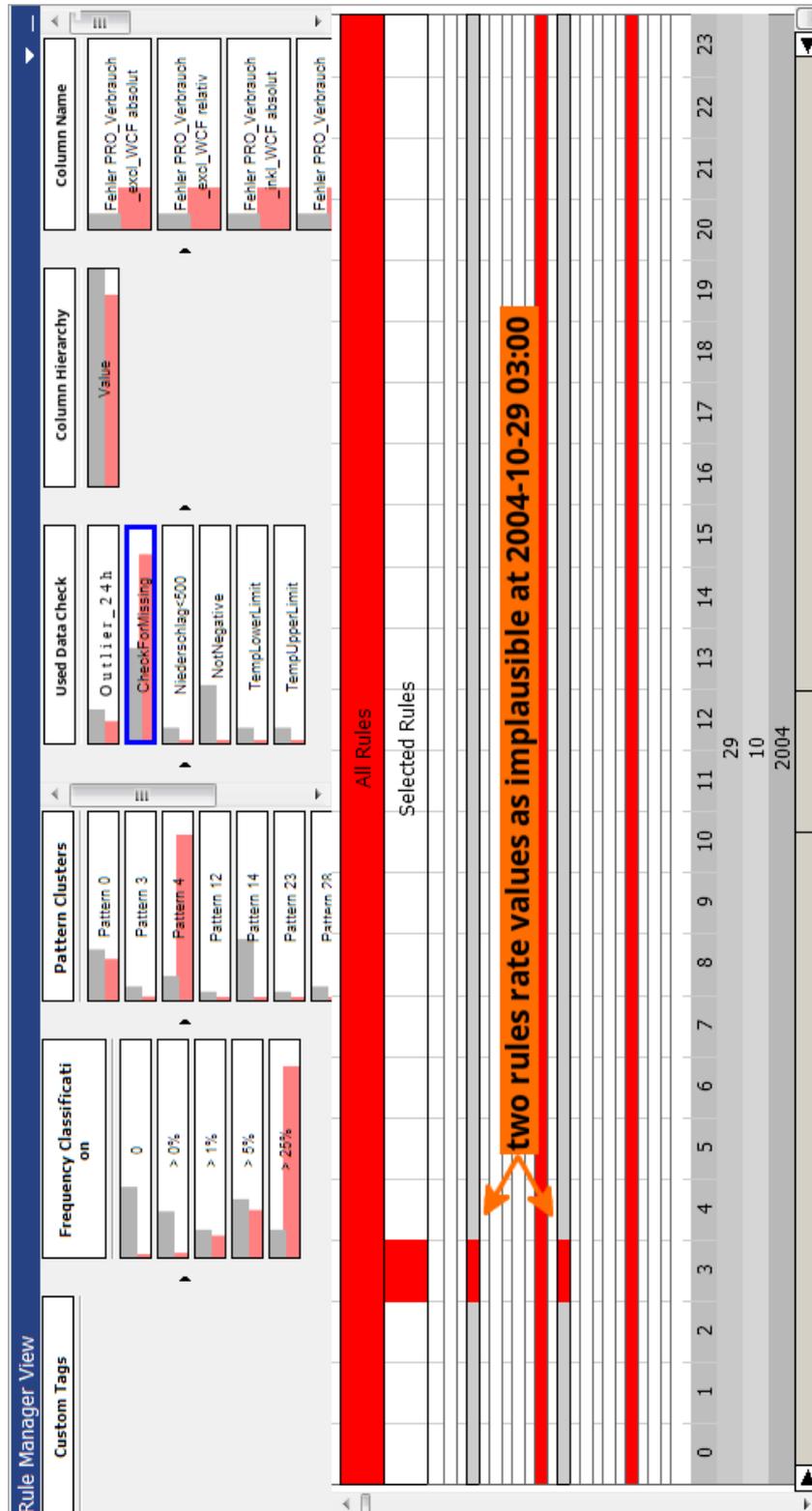


Figure 8.5: Using the interaction methods, we drill down on the time axis and identify that two rules rate values as not plausible at 2004-10-29 03:00. Using the tooltip, which is not shown here, we can identify the affected data columns as Fehler_Pro_Verbrauch_inkl_WCP_relative and Fehler_Pro_Verbrauch_excl_WCP_absolut.

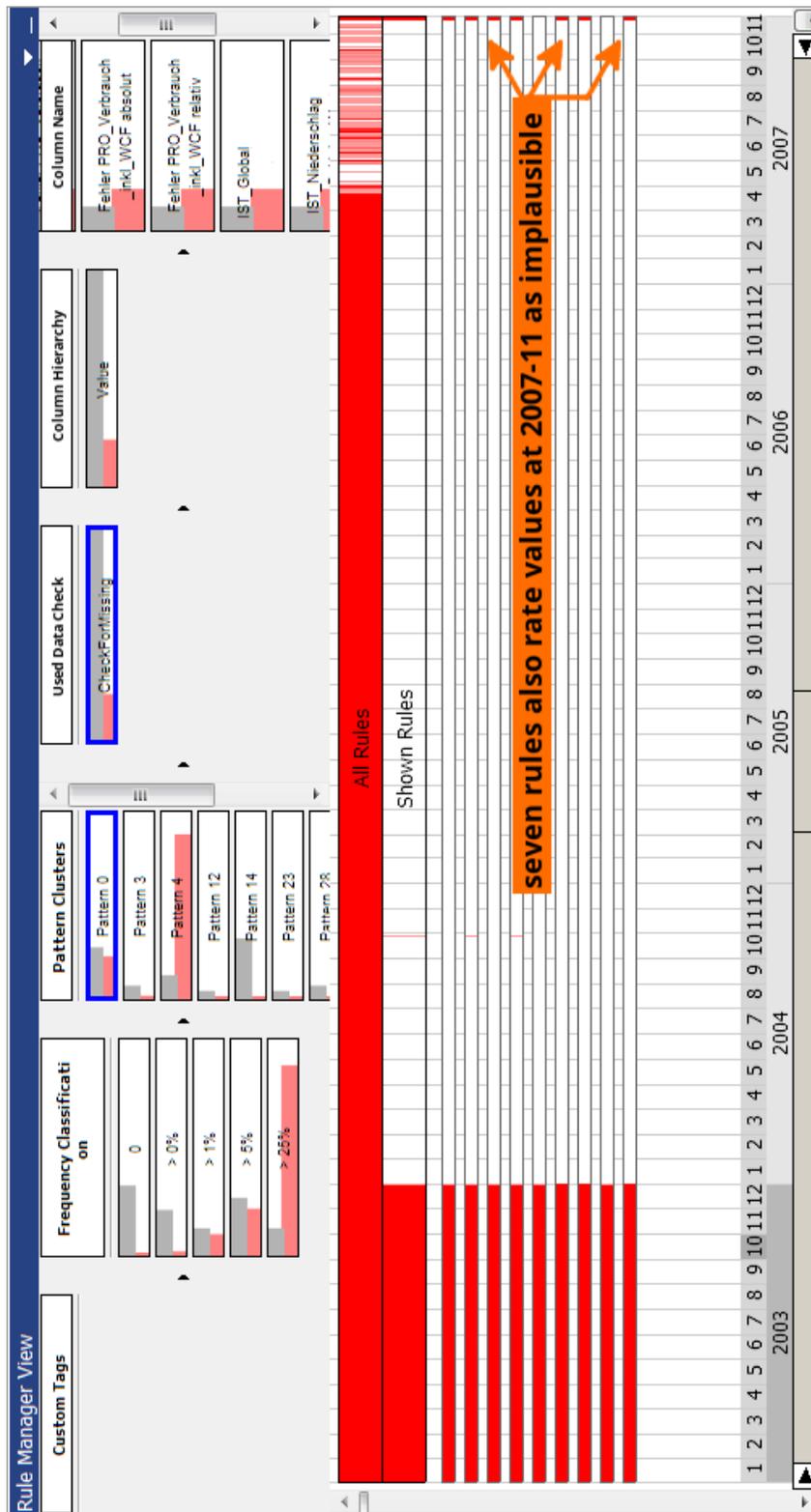


Figure 8.6: We select the group "Pattern 0" which represents a cluster of rules with a similar structure of not plausible values, as it then can be seen in the rule-based overview, which lists nine rules. We can also see, that seven of the nine rules also rate values at 11-2007 as implausible.

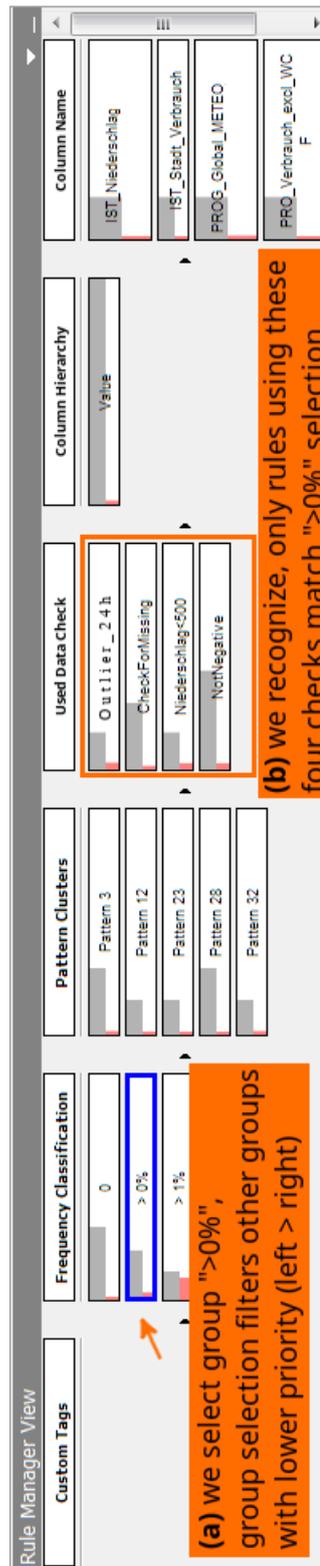


Figure 8.7: Using the group-based overview, we recognize that four data checks are used by rules which rate more than 0 but less than 1% entries as implausible.

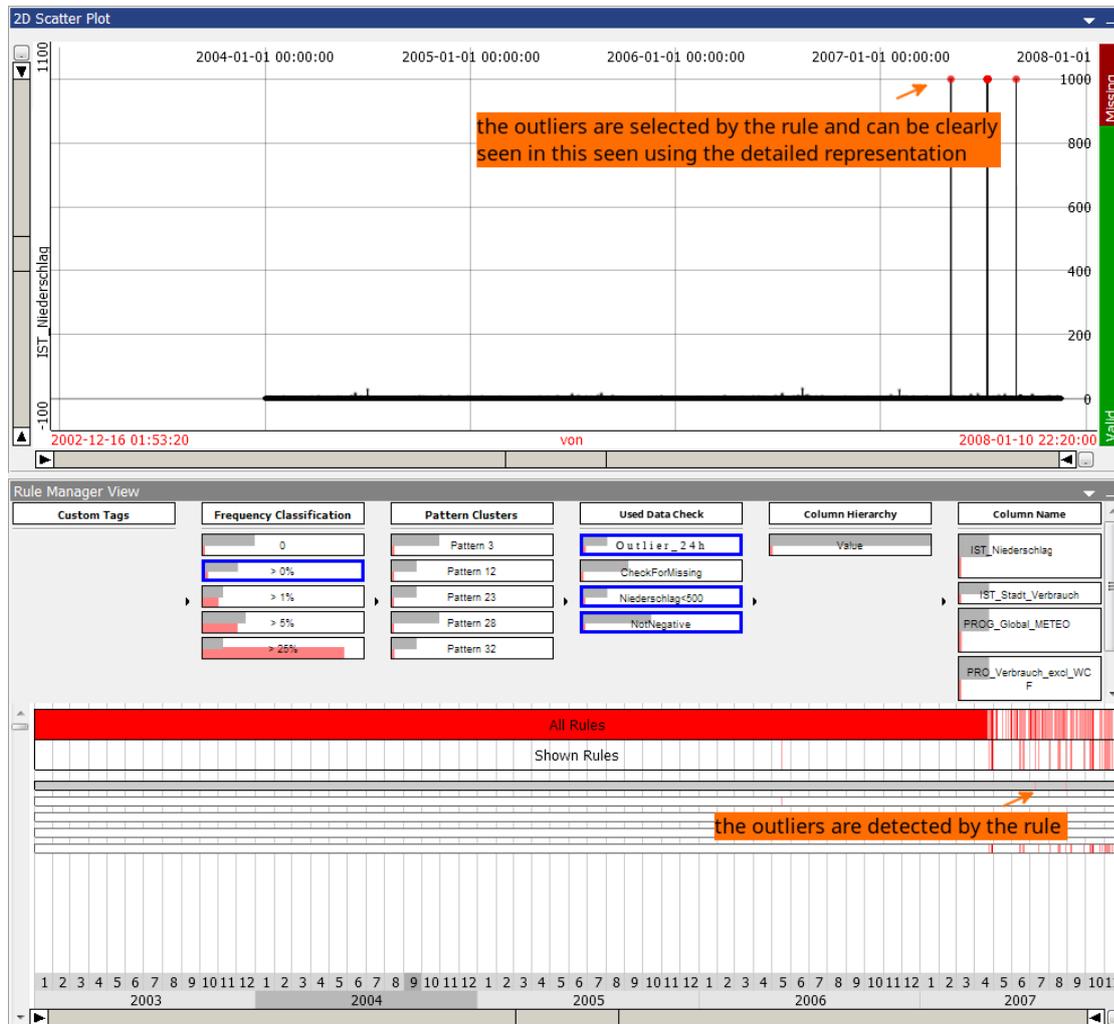


Figure 8.8: IST_Niederschlag contains outliers. They can be clearly seen in this visualization and are also detected by the rule (red highlighted).

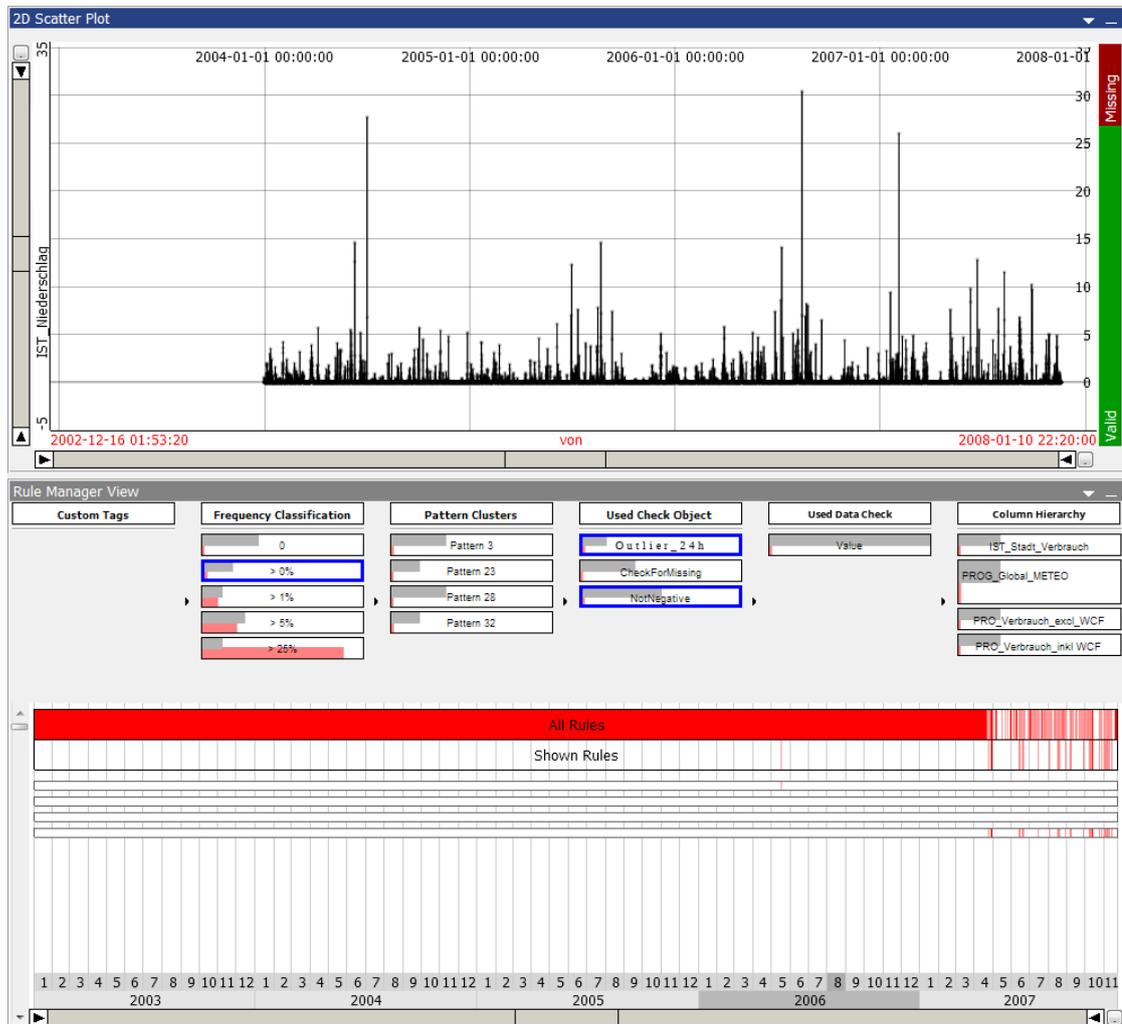


Figure 8.9: Applying the rule imputes the outliers using the set imputation strategy. In this case, the points are replaced by interpolated values. Furthermore, the outliers affected the scaling of the Y axis, which made it difficult to distinguish other values.

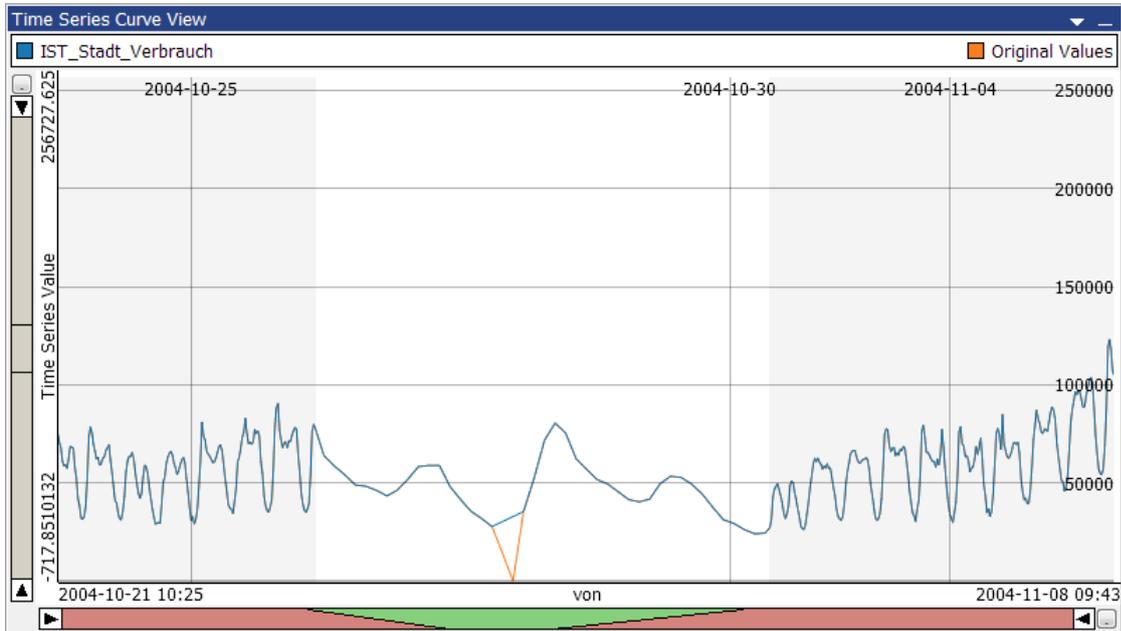


Figure 8.10: The original values (orange) can be easily compared with the modified values (blue).

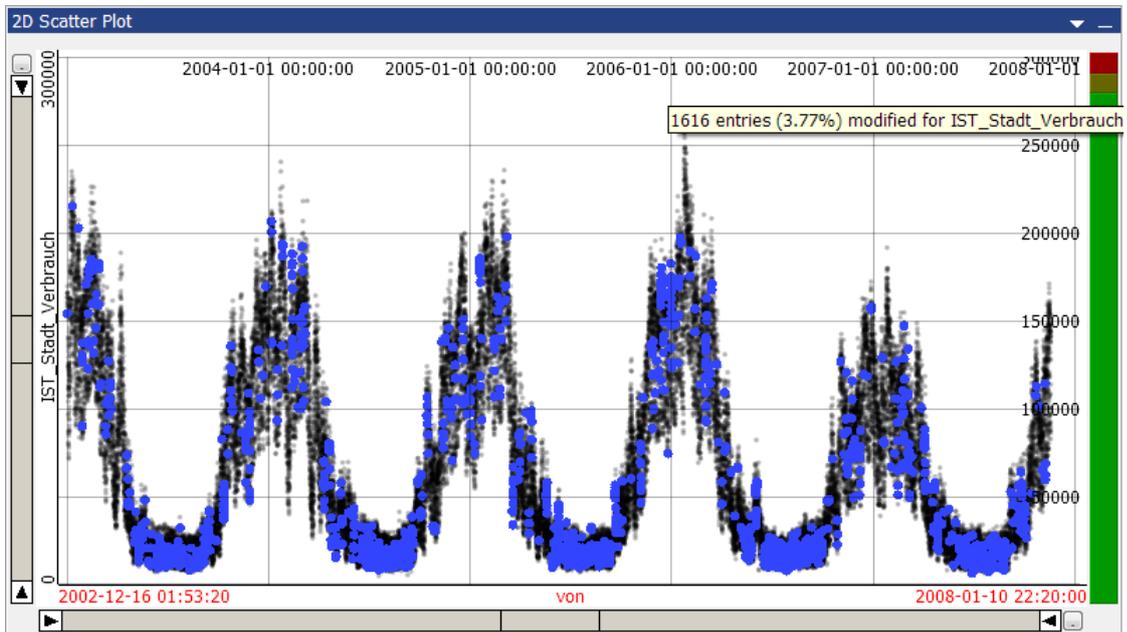
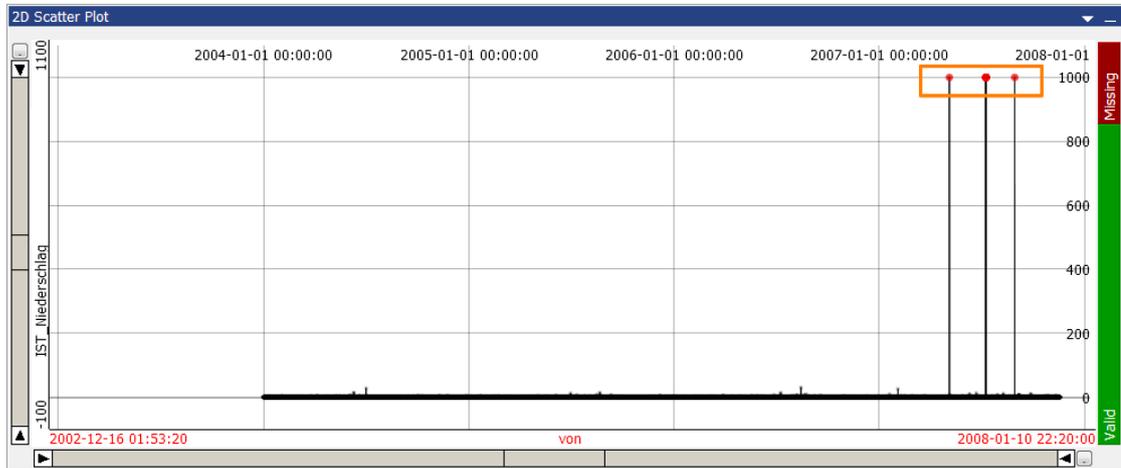
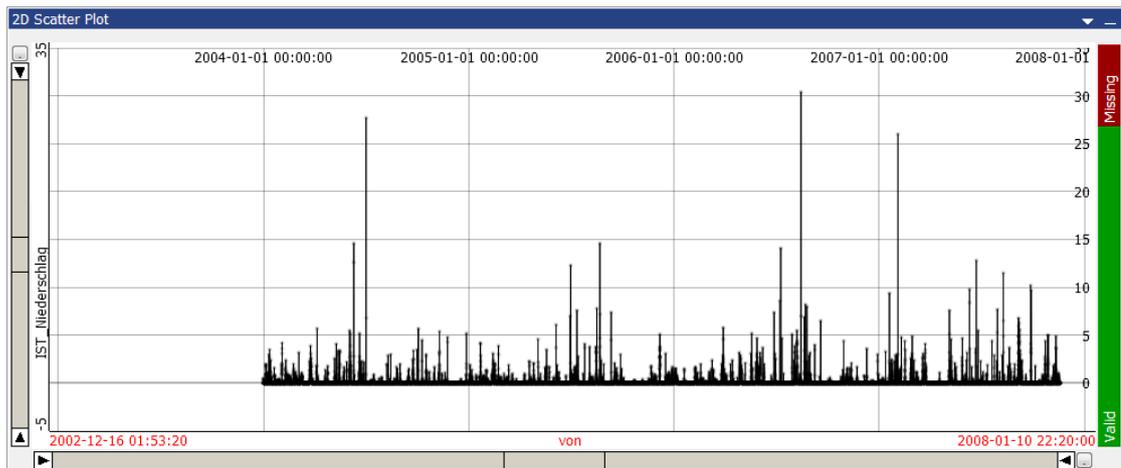


Figure 8.11: The widget provides a fast overview of the ratio between original entries (green), modified entries (dark yellow) and missing entries (red). Hovering the areas shows detailed information (tooltip) and highlights the corresponding entries. In this screenshot, the dark yellow area is hovered by the user.



(a)



(b)

Figure 8.12: Instant imputation of implausible values. **(a)** shows the data including the outliers. They affect the scaling of the Y axis, so that other values are difficult to distinguish. The user selects them by brushing a rectangle and calls "Impute selected values" from the main menu. **(b)** shows the data after the instant imputation, which set those values to MISSING.

Discussion and Future Work

In this section, we want to discuss the results of this diploma thesis. We have defined goals and we want to review how these goals are achieved. Furthermore, we want to discuss the limitations and potential future work in this section.

9.1 Achieved Goals

We have created a framework which allows the user to define plausibility rules for data, different overviews to provide insight into the existing plausibility rules and their evaluation results and methods to modify data and review the changes afterwards.

Our solution provides different techniques to the user to achieve the outlined goals, as described in Section 2.2. In detail, the goals are tackled by the following possible interaction techniques:

- **Provide an overview of the data and the structure of not plausible values**
The user can evaluate the results of the plausibility rules in different overviews. He can use the data-based overview to get an overview of the structures related to the data itself, he can add category-based partitioners to the overview axis to review the error locations with respect to the distribution in different categories.
- **Mark the erroneous values**
As soon as plausibility rules are defined, values which are rated as erroneous respectively not plausible are highlighted in the provided rule manager view. Additionally, if plausibility rules are selected, the current selection is set to all data rows which contain not plausible values with respect to the selected rules. An application wide accentuation of not plausible values is then implicitly done by linking all views.
- **Provide a user interface to define a set of plausibility rules**
We provide a new view for VISPLORE, which does not only provide three overviews to get an overview of the rules, it provides also user interactions to create new rules, edit rules, delete rules or manage the different objects which are needed to define plausibility rules.
- **Store the plausibility rules optionally independent from the actual data**
The rule set containing the defined data checks, imputation strategies, suggestion strategies and actually plausibility rules which use these objects, can be exported to an XML file and stored

independent from the other data. Even when different data is loaded into VISPLORE, an XML file containing the rule set can be imported and applied to the current data.

- **Provide an overview of the plausibility rules**

The new view provides three different overviews, of which two overviews are focused on the plausibility rules. The group-based overview provides a fast overview of the number of rules and the number of not plausible values for groups of rules. The rule-based overview lists the rules one by one and provides a fast overview of the evaluation result of each rule.

- **Provide detailed information of violated plausibility rules**

In addition to the quick overview provided by the rule-based overview, detailed information is shown while a rule is hovered. Furthermore, when the user double clicks on a rule, a 2D scatterplot with the corresponding data column is opened and all values which are rated as not plausible are highlighted.

- **Provide methods to impute erroneous values instantly**

If the user detects not plausible values while analyzing the data with VISPLORE, he does not need to define a plausibility rule to impute these entries. Instead, he can select the entries by using the available selection methods and call "Impute selected entries", which shows up a dialog to select the imputation strategy. The selected values are immediately imputed with respect to the chosen strategy.

- **Provide methods to impute values evaluated as not plausible by plausibility rules**

The plausibility rules consist of two parts, one part is responsible for the evaluation of values and the other part is responsible for the imputation of values which are evaluated as not plausible. The user can apply or execute selected plausibility rules to trigger the imputation part, while the evaluation is continuously done and the results are visualized in the new view.

- **Traceability of modified values**

Whenever values are modified by using plausibility rules or instant imputation, additional meta columns are updated or created if necessary. These meta columns contain the original values, the name of the first data check which rated the values as not plausible, the name of the imputation strategy which is responsible for the current values and a full textual modification history. These columns can be analyzed with the existing views in VISPLORE and provide a full traceability.

- **Implement a new view to provide an overview of erroneous data and plausibility rules**

We have implemented a new view which consists of three parts, a data-based overview, a group-based overview and a rule-based overview. The view and its parts are designed to get an overview of the erroneous data, different group of rules and the plausibility rules.

- **Use or adapt existing views whenever possible**

The implemented framework to create, edit and use plausibility rules uses existing views in many cases. For a detailed rule inspection, a 2D scatterplot view is opened and parameterized. To compare the modified data column with the original data column, a time series view is used. Furthermore, the detailed modification history can be reviewed using a tabular data representation.

- **Mark values as erroneous by selection**

VISPLORE's existing functionality for selecting data subsets via brushing can directly be used to define data entries as erroneous/implausible.

- **Define a plausibility rule by selecting erroneous values**

The user can employ the current selection to mark suspicious values. After giving a hint (e.g.,

outliers, duplicates), a corresponding data check tries to find a matching parameterization which evaluates the selected values as not plausible.

9.2 Limitations

However, although many of the goals are achieved, our framework also lacks of some shortcomings. The proposed data checks are univariate. This means that they can only test one data column. With the current implementation of plausibility rules, it is not possible to test multivariate restrictions, like the example in Equation 9.1.

$$(FehlerPrognoseVerbrauch) = (PrognoseVerbrauch) - (Verbrauch) \quad (9.1)$$

Another restriction is the fixed number of one imputation strategy per plausibility rule. For some cases it may be useful to have multiple imputation actions which are triggered in sequence. If the data check evaluates too many values as not plausible, it may also be useful to modify the plausibility vector manually before the imputation is applied, for example by using existing selection techniques.

In general, the design of plausibility rules, consisting of links to independent data checks and imputation strategies, which define the concrete check algorithm or imputation, should definitely be revised. Although this design has been chosen to allow multiple rules doing the same check or imputation on different data columns, while using the finished implementation, we have learned that this concept is not as user-friendly as we have thought while designing the concept. Also, the need for configuring checks and imputations before plausibility rules can be defined, complicates the use of the framework.

The designed view provides an overview of the plausibility rules and their structures, but there are also some aspects for improvement. The group-based overview was designed to enhance the scalability, so that a high number of rules can be inspected. Indeed, using the finished implementation showed that it does not scale to a non-trivial number of plausibility rules (e.g., hundreds). In this case, an aggregated approach could be necessary and should be considered. The provided possibilities to partition the shown time axis using categories gives an additional insight which categories correlate with a high number of not plausible values, but the problem here is that the user has to know which categories he want to inspect. An automatic approach which tells the user, which categories have a high correlation with not plausible values would be useful and should be considered in future work.

To summarize, we have designed a solution to provide interactive data editing in a visual analytic framework. Our solution uses plausibility rules to evaluate values as plausible or not plausible and to define how implausible values should be imputed. It provides instant imputation to impute values and it writes additional meta data when values are imputed. Existing views and shortcuts can be used to inspect these meta data, so that original and modified values can be compared and the modification history can be reviewed. Indeed, for each of the provided techniques and approaches there is some future work and additional research can improve the usefulness, usability, and scalability.

9.3 Future Work

There are many possibilities for future work. In this section, we want briefly describe some of the ideas.

First, the creation of rules is still a time-consuming step. The definition of data checks and imputation strategies as a necessary preprocessing step is not that intuitive and a more intuitive semi-automatic mechanism to define these rules would be an interesting feature. Furthermore, in this thesis the data checks and imputation strategies were restricted to be univariate. Future work could discuss the consequences of multivariate checks, the necessary modifications to visualize the rule results and facilitate the use of multivariate checks.

Considering the imputation, the possibility to apply a manual or at least a selective imputation, that means a imputation which is only applied on a selectable subset of implausible values, should be considered. It would allow the user to refine the set of imputed entries before the imputation is applied, which would be helpful if the data check rated too many entries as implausible.

In this thesis, we have also presented a data check and an imputation strategy which is based on arbitrary R scripts, however, the actual usage of them was complicated because the user interface was not designed for scriptable data checks and imputation strategies. Improving the support and user experience using such scriptable data checks and imputation strategies could be an interesting point for future work.

Based on the presented framework, an important starting point for future work is scalability. The current approach does not scale for a large number of rules and developing an approach which tackles this problem would be a valuable contribution.

Moreover, the rule view always shows all implausible values for a specific rule. This could be potentially improved using an aggregated representation, which would give an overview first, and details are provided on demand.

A point to improve the group-based overview would be the defined priorities for the different group types, which should be configurable by the user, as it depends on the concrete data which priorities make sense.

An interesting topic for future work could be the integration of machine-learning algorithms to detect quality issues based on semi-automatic algorithms instead of predefined rules. Using visual analytics to interactively define a set of labels for machine-learning algorithms which then derive models to classify general data as plausible or implausible could be a interesting approach. Similar to that, future work can also consider improvements regarding the semi-automatic rule creation as described in Chapter 7, which creates rules based on user selection and an additional hint.

Furthermore, an evaluation by real users, also considering the deployment and adaption, is an important point for future work.

Summary and Conclusion

In this thesis, we have presented an approach to detect and handle data quality issues using a visual analytics framework. The approach is based on plausibility rules, which are used to define specific constraints for data attributes and to define imputation strategies for implausible values.

We have implemented three overviews, a data-based overview, a group-based overview, and a rule-based overview to provide insight into the structure of the found data quality issues. Furthermore, the results of the plausibility rules can be inspected in detail. It is also possible to use already existing views and features of VISPLORE.

The plausibility rules can be used to impute implausible values using a predefined imputation strategy. When values are modified, meta-data are created to be able to trace the modifications. We have implemented shortcuts to compare original and modified values or track the full modification history in a tabular view. Furthermore, we have modified an interactive widget which originally showed the ratio between missing and valid data, so that it now shows the ratio between missing, imputed and valid data.

Moreover, we have shown two exemplary use cases for our developed framework using a real-world data set to demonstrate how data quality issues can be identified and imputed using our implemented framework. Finally, we have discussed the limitations and possible future work.

To conclude, we have presented an approach which allows the user to interactively analyze data quality issues and to impute implausible values using an existing visual analytics tool. However, our approach has some limitations, which were discussed together with different ideas to extend and improve the framework.

List of Figures

1.1	In 1844, many people suffered from a deadly disease. This visualization helped to discover a contaminated water pump as the cause for the deaths. The bars represent the number of deaths in geographic areas, and it can be seen that the deaths are clustered around the encircled pump [Gil58].	3
1.2	Volume rendering of a human head, which was scanned with magnetic resonance imaging. The three visualizations represent the same scan, but three different parameter settings for the visualization method. Screenshot taken from an own student project.	4
1.3	An example for flow visualization, showing a specific weather situation. The data set contains a 2D grid, each point in the grid has the direction and the velocity of the wind as additional information. Streamlines [JL97] are used to visualize the flow of the wind. Screenshot taken from an own student project.	8
1.4	Information Visualization: Interactive comparison of multiple biological tree structures. Approach published by Bremm et al. [BvLH ⁺ 11], screenshot taken from an own student project.	9
1.5	High-level visualization taxonomy, illustrated by examples. Design models are classified based on whether they are discrete or continuous and by how much the algorithm designer chooses display attributes (spatialization, timing, colour, and transparency). Examples show different constraints on spatialization [TM04].	10
1.6	The abilities of a machine differ from the human abilities. Visual Analytics tries to combine the best of both [KAF ⁺ 08].	10
1.7	The Visual Data Exploration process. It is characterised by interaction between data, visualisation, models of the data, and the user to discover knowledge [KKEM10].	11
3.1	The visual analytics framework Caleydo [LSKS10] layouts multiple views in a 2.5D setting and connects related information using VisLinks [CC07] across views.	18
3.2	Overview + Detail shown by the example of Google Maps. It shows a large detail map and a small overview map spatially separated (yellow arrow) [goo].	19
3.3	VISPLORE provides a data range slider with zooming ability. When the user interactively zooms in, the slider represents the zoomed area using green color and the contextual area using red color. Note the corresponding geometrically distorted visualization.	20
3.4	The Table Lens provides different degrees of detail for the focus and the context. Screenshot of VISPLORE.	21
3.5	In-place focus+context as used by Kosara et al. [KMH01] to mark context and focus.	21
3.6	Classification of data quality problems from different data sources [RD00].	22
3.7	Point anomalies in a two-dimensional data set. The data has two normal regions N_1 and N_2 , since most observations lie in these two regions. Points o_1 , o_2 and points in region O_3 are anomalies [CBK09].	23
3.8	Showing a contextual anomaly. Note that the temperature at time t_1 is no anomaly while the same temperature at time t_2 must be considered an anomaly [CBK09].	24

3.9	Collective anomaly corresponding to an <i>Atrial Premature Contraction</i> in a human electrocardiogram output [CBK09].	25
3.10	The iterative process of data wrangling and data analysis [KHP ⁺ 11].	26
3.11	Sales data visualized using a 2D scatterplot view on the top and a parallel coordinates view on the bottom.	31
3.12	An example visualization scenario using VISPLORE. The user has selected some data points in a 2D scatterplot and a bar chart visualizes the average profit per product category for the selected (red bars) and not-selected (black bars) data points.	32
3.13	Showing the interaction between user, event handling thread and visualization thread. The visualization thread is told to stop after each user interaction and intermediate results are displayed. After updating the parameters, the visualization thread is restarted and re-uses as many layers as possible. L1-L4 represent different layers [PTMB09].	32
3.14	This picture shows the integration of R into VISPLORE. (a) describes the workflow concept, (b) shows VISPLORE and the R parts, which consist of an object browser and an R console [KBFP12]	33
4.1	A data check C takes input values, has an internal blackbox evaluator function and outputs a plausibility vector. Additional context information is not shown here.	37
4.2	An imputation strategy I takes the original values and the plausibility evaluation as input, has an internal blackbox imputation function and outputs the imputed values. Additional context information is not shown here.	38
5.1	A mockup of the data-based overview in an early development stage. For each data row, mapped on the X axis, the rates of rejecting plausibility rules is mapped on the Y axis. Note that a rule works on a data column, while this overview works on data rows.	42
5.2	The data-based overview, showing implausible values detected by all rules and by a selected subset. For clarification, the implausible values are highlighted by orange rectangles in this screenshot. The data rows are sorted and grouped by the values of a data column which represents the timestamp. It can be clearly seen that most of the not plausible values happened in 2003.	44
5.3	Illustration of the visualization using partitioners. In this example based on table 5.1, the columns A and C are used as partitioners. The partitioners divide the data rows into groups which are separated visually. As defined by the user, the rows are ordered by the column Time within each group. $v(r)$ returns the visualization color for the data row r	45
5.4	A screenshot of the Rule Manager View, including the group-based overview. The different properties and groups can be seen in the upper part of the view. On the upper half of each button, a gray bar represents the percentage of rules which belong to this group (e.g., all rules belong to the group "CustomTag1", while only a few rules belong to the group "Pattern 17"). On the lower half of each button, the percentage of implausible data rows, as evaluated by the corresponding rules, is shown (e.g., nearly all data rows are rated as implausible by at least one of the rules which use the data check "CheckForMissing").	50
5.5	A screenshot of the Rule Manager View, including the group-based overview. The group ">0%" has been selected and thus, groups with a lower priority are filtered. The shown plausibility rules in the rule-based overview, which is located below the group-based overview, are also filtered accordingly.	51
5.6	A screenshot of the first prototype of the rule-based overview, showing three different plausibility rules. The first prototype used a tabular layout and placed the visual representations of the plausibility evaluations in a column, and additional rule details (tested data column, number of implausible values) in further columns.	52

5.7	The second prototype of the rule-based overview, together with the data-based overview on top. The visual representation of the plausibility evaluation is similar and the user can compare the location of implausible values. Additional rule details (used data check, tested data column, number of implausible entries) are now displayed for each rule above the visual representation, using vertical instead of horizontal space.	53
7.1	The configuration of data checks, imputation and suggestion strategies. a) lists the current configured data checks together with their names, types, parameters and usage counts. New data checks can be added or copied and unused data checks can be deleted. In area b) , the parameters of a data check can be changed by the user.	63
7.2	This screenshot shows the rule manager view with many plausibility rules loaded. (a) The group-based overview allows the user to select subsets of rules. (b) The data-based overview displays all not plausible entries with respect to all rules and all currently selected rules or all rules within a group, if a group is currently hovered. (c) The rule-based overview shows implausible entries separately for each rule.	68
7.3	This image shows a rule in the rule-based overview and the detailed visualization in a scatterplot.	69
7.4	Drilling down on partitioner groups. (a) All groups are shown. (b) The user right-clicks on the year 2007 and all other years are hidden. (c) The user right-clicks on the month 11 and all other months are hidden.	70
7.5	When data is edited by applying a plausibility rule, an additional widget which can be seen on the right side, provides fast interactive information about the ratio of missing, imputed and valid values.	70
7.6	In this view parameterization, original values can be compared with modified values. The modified values are drawn in blue while the original values are shown in orange.	71
7.7	The detailed modification history can be inspected using a tabular view as shown above.	71
8.1	Creating new rules for the selected data columns. Note that the shown rules are automatically suggested by VISPLORE.	82
8.2	(a) Many not plausible entries appear until 04-2007. (b) The data check "CheckForMissing" is responsible for many values evaluated to not plausible.	83
8.3	We have now selected the Group "CheckForMissing" and see the corresponding rules. Hovering the rules shows us detail information and we recognize that the column "Fehler_PRO_Verbrauch_inkl_WCF absolut" has missing values for 2003.	84
8.4	Using the rule-based overview, we identify additional implausible values around 10-2004 (highlighted by orange boxes).	85
8.5	Using the interaction methods, we drill down on the time axis and identify that two rules rate values as not plausible at 2004-10-29 03:00. Using the tooltip, which is not shown here, we can identify the affected data columns as Fehler_Pro_Verbrauch_inkl_WCP_relative and Fehler_Pro_Verbrauch_excl_WCP_absolut.	86
8.6	We select the group "Pattern 0" which represents a cluster of rules with a similar structure of not plausible values, as it then can be seen in the rule-based overview, which lists nine rules. We can also see, that seven of the nine rules also rate values at 11-2007 as implausible.	87
8.7	Using the group-based overview, we recognize that four data checks are used by rules which rate more than 0 but less than 1% entries as implausible.	88
8.8	IST_Niederschlag contains outliers. They can be clearly seen in this visualization and are also detected by the rule (red highlighted).	89
8.9	Applying the rule imputes the outliers using the set imputation strategy. In this case, the points are replaced by interpolated values. Furthermore, the outliers affected the scaling of the Y axis, which made it difficult to distinguish other values.	90

8.10	The original values (orange) can be easily compared with the modified values (blue).	91
8.11	The widget provides a fast overview of the ratio between original entries (green), modified entries (dark yellow) and missing entries (red). Hovering the areas shows detailed information (tooltip) and highlights the corresponding entries. In this screenshot, the dark yellow area is hovered by the user.	91
8.12	Instant imputation of implausible values. (a) shows the data including the outliers. They affect the scaling of the Y axis, so that other values are difficult to distinguish. The user selects them by brushing a rectangle and calls "Impute selected values" from the main menu. (b) shows the data after the instant imputation, which set those values to MISSING.	92

List of Tables

1.1	An example data set showing the structure of the data as it is used in this thesis.	6
5.1	Example data set to illustrate the use of partitioners	45
8.1	The suggested set of plausibility rules.	78
8.2	The final set of plausibility rules.	79

Bibliography

- [ATF10] Andreas Alfons, Matthias Templ, and Peter Filzmoser. An object-oriented framework for statistical simulation: The R package simframe. *Journal of Statistical Software*, 37(3):1–36, 2010.
- [BHL97] Michael Bankier, Anne-Marie Houle, and Manchie Luc. 1996 canadian census demographic variables imputation. In *Proceedings of the Survey Research Methods Section, American Statistical Association*, pages 389–394, 1997.
- [BHW02] Auguste C Boissonnade, Lawrence J Heitkemper, and David Whitehead. Weather data: cleaning and enhancement. *Climate Risk and the Weather Market*, pages 73–98, 2002.
- [BMMS91] Andreas Buja, John Alan McDonald, John Michalak, and Werner Stuetzle. Interactive data visualization using focusing and linking. In *Visualization, 1991. Visualization'91, Proceedings., IEEE Conference on*, pages 156–163. IEEE, 1991.
- [BPF11] Wolfgang Berger, Harald Piringer, Peter Filzmoser, and Eduard Gröller. Uncertainty-aware exploration of continuous parameter spaces using multivariate prediction. In *Computer Graphics Forum*, volume 30, pages 911–920. Wiley Online Library, 2011.
- [BT05] Adrian Baddeley and Rolf Turner. Spatstat: an R package for analyzing spatial point patterns. *Journal of statistical software*, 12(6):1–42, 2005.
- [BvLH⁺11] Sebastian Bremm, Tatiana von Landesberger, Martin Heß, Tobias Schreck, Philipp Weil, and Kay Hamacher. Interactive visual comparison of multiple trees. In *Visual Analytics Science and Technology (VAST), 2011 IEEE Conference on*, pages 31–40. IEEE, 2011.
- [CBK09] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Computing Surveys (CSUR)*, 41(3):15, 2009.
- [CC07] Christopher Collins and Sheelagh Carpendale. Vislink: Revealing relationships amongst visualizations. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1192–1199, 2007.
- [CD97] Surajit Chaudhuri and Umeshwar Dayal. An overview of data warehousing and olap technology. *ACM Sigmod record*, 26(1):65–74, 1997.
- [CKB08] Andy Cockburn, Amy Karlson, and Benjamin B Bederson. A review of overview+ detail, zooming, and focus+ context interfaces. *ACM Computing Surveys (CSUR)*, 41(1):2, 2008.
- [CM84] William S Cleveland and Robert McGill. The many faces of a scatterplot. *Journal of the American Statistical Association*, 79(388):807–822, 1984.

- [DGH03] Helmut Doleisch, Martin Gasser, and Helwig Hauser. Interactive feature specification for focus+ context visualization of complex simulation data. In *Proceedings of the symposium on Data visualisation 2003*, pages 239–248. Eurographics Association, 2003.
- [DJ03] Tamraparni Dasu and Theodore Johnson. *Exploratory data mining and data cleaning*, volume 479. John Wiley & Sons, 2003.
- [dWPS11] Ton de Waal, Jeroen Pannekoek, and Sander Scholtus. *Handbook of statistical data editing and imputation*, volume 563. Wiley.com, 2011.
- [Fay99] Robert E Fay. Theory and application of nearest neighbor imputation in census 2000. In *Proceedings of the Survey Research Methods Section*, volume 17, pages 112–121, 1999.
- [Fri91] Jerome H Friedman. Multivariate adaptive regression splines. *The annals of statistics*, pages 1–67, 1991.
- [GGAM12] Theresia Gschwandtner, Johannes Gärtner, Wolfgang Aigner, and Silvia Miksch. A taxonomy of dirty time-oriented data. In *Multidisciplinary Research and Practice for Information Systems*, pages 58–72. Springer, 2012.
- [Gil58] Edmund W Gilbert. Pioneer maps of health and disease in England. *The Geographical Journal*, 124(2):172–183, 1958.
- [GK97] Leopold Granquist and J Kovar. Editing of survey data: how much is enough. *Survey measurement and process quality*, pages 415–435, 1997.
- [goo] Google support. <https://support.google.com/earth/answer/148069?hl=en>. Accessed: 2014-02-11.
- [GPT06] Pedro Galeano, Daniel Peña, and Ruey S Tsay. Outlier detection in multivariate time series by projection pursuit. *Journal of the American Statistical Association*, 101(474):654–669, 2006.
- [GTC01] Georges Grinstein, Marjan Trutschl, and Urska Cvek. High-dimensional visualizations. In *Proceedings of Workshop on Visual Data Mining, ACM Conference on Knowledge Discovery and Data Mining*, pages 1–14, 2001.
- [gtk] the gtk+ project. <http://www.gtk.org/>. Accessed: 2013-10-13.
- [HA04] Victoria J Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.
- [HHSW09] Alexander Hubmann-Haidvogel, Arno Scharl, and Albert Weichselbraun. Multiple coordinated views for searching and navigating web content repositories. *Information Sciences*, 179(12):1813–1821, 2009.
- [HJ05] Charles D Hansen and Christopher R Johnson. *The visualization handbook*. Access Online via Elsevier, 2005.
- [HN06] Jeffrey Hoogland and Statistics Netherlands. Selective editing using plausibility indicators and slice. *Statistical Data Editing, Volume*, (3):106–130, 2006.
- [ID91] Alfred Inselberg and Bernard Dimsdale. Parallel coordinates. In *Human-Machine Interactive Systems*, pages 199–233. Springer, 1991.

- [JL97] Bruno Jobard and Wilfrid Lefer. Creating evenly-spaced streamlines of arbitrary density. *Visualization in scientific computing*, 97:43–55, 1997.
- [Jol02] Ian T Jolliffe. Graphical representation of data using principal components. *Principal Component Analysis*, pages 78–110, 2002.
- [JSMR09] Xin Jin, Soumik Sarkar, Kushal Mukherjee, and Asok Ray. Suboptimal partitioning of time-series data for anomaly detection. In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 1020–1025. IEEE, 2009.
- [KAF⁺08] Daniel Keim, Gennady Andrienko, Jean-Daniel Fekete, Carsten Görg, Jörn Kohlhammer, and Guy Melançon. *Visual analytics: Definition, process, and challenges*. Springer, 2008.
- [KBFP12] Johannes Kehrler, Roland N Boubela, Peter Filzmoser, and Harald Piringer. A generic model for the integration of interactive visualization and statistical computing using R. In *Visual Analytics Science and Technology (VAST), 2012 IEEE Conference on*, pages 233–234. IEEE, 2012.
- [KCH⁺03] Won Kim, Byoung-Ju Choi, Eui-Kyeong Hong, Soo-Kyung Kim, and Doheon Lee. A taxonomy of dirty data. *Data Mining and Knowledge Discovery*, 7(1):81–99, 2003.
- [Kei02] Daniel A Keim. Information visualization and visual data mining. *Visualization and Computer Graphics, IEEE Transactions on*, 8(1):1–8, 2002.
- [KHG03] Robert Kosara, Helwig Hauser, and Donna L Gresh. An interaction view on information visualization. *State-of-the-Art Report. Proceedings of EUROGRAPHICS*, 2003.
- [KHP⁺11] Sean Kandel, Jeffrey Heer, Catherine Plaisant, Jessie Kennedy, Frank van Ham, Nathalie Henry Riche, Chris Weaver, Bongshin Lee, Dominique Brodbeck, and Paolo Buono. Research directions in data wrangling: Visualizations and transformations for usable and credible data. *Information Visualization*, 10(4):271–288, 2011.
- [KKEM10] Daniel A Keim, Jörn Kohlhammer, Geoffrey Ellis, and Florian Mansmann. *Mastering The Information Age-Solving Problems with Visual Analytics*. Florian Mansmann, 2010.
- [KMH01] Robert Kosara, Silvia Miksch, and Helwig Hauser. Semantic depth of field. In *Information Visualization, IEEE Symposium on*, pages 97–97. IEEE Computer Society, 2001.
- [KMS⁺08] Daniel A Keim, Florian Mansmann, Jörn Schneidewind, Jim Thomas, and Hartmut Ziegler. *Visual analytics: Scope and challenges*. Springer, 2008.
- [KPHH11] Sean Kandel, Andreas Paepcke, Joseph Hellerstein, and Jeffrey Heer. Wrangler: Interactive visual specification of data transformation scripts. In *PART 5—Proceedings of the 2011 annual conference on Human factors in computing systems*, pages 3363–3372. ACM, 2011.
- [KPP⁺12] Sean Kandel, Ravi Parikh, Andreas Paepcke, Joseph M Hellerstein, and Jeffrey Heer. Profiler: Integrated statistical analysis and visualization for data quality assessment. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pages 547–554. ACM, 2012.
- [LSKS10] Alexander Lex, Marc Streit, Ernst Kruijff, and Dieter Schmalstieg. Caleydo: Design and evaluation of a visual analysis framework for gene expression data in its biological context. In *Pacific Visualization Symposium (PacificVis), 2010 IEEE*, pages 57–64. IEEE, 2010.

- [LVVJ03] Maurizio Lenzerini, Yannis Vassiliou, Panos Vassiliadis, and Matthias Jarke. *Fundamentals of data warehouses*. Springer, 2003.
- [MF03] Heiko Müller and Johann-Christoph Freytag. Problems, methods, and challenges in comprehensive data cleansing. Technical Report HUB-IB-164, Humboldt University, Berlin, 2003.
- [Moo98] Gordon E Moore. Cramming more components onto integrated circuits. In *Proceedings of the IEEE*, volume 86(1), pages 82–85, 1998.
- [MP13] Thomas Mühlbacher and Harald Piringer. A partition-based framework for building and validating regression models. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):1962–1971, 2013.
- [MS03] Wolfgang Müller and Heidrun Schumann. Visualization methods for time-dependent data – an overview. In *Simulation Conference, 2003. Proceedings of the 2003 Winter*, volume 1, pages 737–745. IEEE, 2003.
- [ope] Open graphics library. <http://www.opengl.org/>. Accessed: 2013-10-13.
- [Pir05] Harald Piringer. Design guidelines and concepts of the infovis library. (available via request at hp@vrvis.at), 2005.
- [PKH04] Harald Piringer, Robert Kosara, and Helwig Hauser. Interactive focus+ context visualization with linked 2d/3d scatterplots. In *Coordinated and Multiple Views in Exploratory Visualization, 2004. Proceedings. Second International Conference on*, pages 49–60. IEEE, 2004.
- [PTMB09] Harald Piringer, Christian Tominski, Philipp Muigg, and Wolfgang Berger. A multi-threading architecture to support interactive visual exploration. *Visualization and Computer Graphics, IEEE Transactions on*, 15(6):1113–1120, 2009.
- [RC94] Ramana Rao and Stuart K Card. The table lens: merging graphical and symbolic representations in an interactive focus+context visualization for tabular information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 318–322. ACM, 1994.
- [RD00] Erhard Rahm and Hong Hai Do. Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.*, 23(4):3–13, 2000.
- [RH01] Vijayshankar Raman and Joseph M Hellerstein. Potter’s wheel: An interactive data cleaning system. In *VLDB*, volume 1, pages 381–390, 2001.
- [Rob07] Jonathan C Roberts. State of the art: Coordinated & multiple views in exploratory visualization. In *Coordinated and Multiple Views in Exploratory Visualization, 2007. CMV’07. Fifth International Conference on*, pages 61–71. IEEE, 2007.
- [RTM⁺03] Theresa-Marie Rhyne, Melanie Tory, Tamara Munzner, Matthew O Ward, Chris Johnson, and David H Laidlaw. Information and scientific visualization: Separate but equal or happy together at last. In *IEEE Visualization*, volume 3, pages 611–614, 2003.
- [Sha05] Yakov Shafranovich. Common format and mime type for comma-separated values (CSV) files. *IETF RFC 4180*, 2005.

- [SS04] Jinwook Seo and Ben Shneiderman. A rank-by-feature framework for unsupervised multidimensional data exploration using low dimensional projections. In *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, pages 65–72. IEEE, 2004.
- [SS06] Ryota Suzuki and Hidetoshi Shimodaira. Pvclust: an R package for assessing the uncertainty in hierarchical clustering. *Bioinformatics*, 22(12):1540–1542, 2006.
- [Str94] Bjarne Stroustrup. *C++ Programming Language, 3/e*. Pearson Education India, 1994.
- [TC05] James J Thomas and Kristin A Cook. *Illuminating the path: The research and development agenda for visual analytics*. IEEE Computer Society Press, 2005.
- [Tea] R Development Core Team. R: a language and environment for statistical computing. <http://www.R-project.org>. R Foundation for Statistical Computing, Vienna. Accessed: 2014-08-16.
- [Tem08] Matthias Templ. Statistical disclosure control for microdata using the r-package sdcmicro. *Transactions on Data Privacy*, 1(2):67–85, 2008.
- [THF11] Matthias Templ, Karel Hron, and Peter Filzmoser. robCompositions: an R-package for robust statistical analysis of compositional data. *Compositional Data Analysis: Theory and Applications*, 2011.
- [TM04] Melanie Tory and Torsten Möller. Rethinking visualization: A high-level taxonomy. In *Information Visualization, 2004. INFOVIS 2004. IEEE Symposium on*, pages 151–158. IEEE, 2004.
- [Tuk77] John W Tukey. Exploratory data analysis. *Reading, Ma*, 231, 1977.
- [VdBCEH05] Jan Van den Broeck, Solveig Argeseanu Cunningham, Roger Eeckels, and Kobus Herbst. Data cleaning: detecting, diagnosing, and editing data abnormalities. *PLoS medicine*, 2(10):e267, 2005.
- [VHJG95] John Vlissides, R Helm, R Johnson, and E Gamma. Design patterns: Elements of reusable object-oriented software. *Reading: Addison-Wesley*, 49:120, 1995.
- [vis] Visual analysis (company). <http://www.visualanalysis.org/>. Accessed: 2013-10-22.
- [War00] Colin Ware. *Information visualization*, volume 2. Morgan Kaufmann San Francisco, 2000.
- [WBWK00] Michelle Q Wang Baldonado, Allison Woodruff, and Allan Kuchinsky. Guidelines for using multiple views in information visualization. In *Proceedings of the working conference on Advanced visual interfaces*, pages 110–119. ACM, 2000.
- [Wil08] Graham Wills. Linked data views. In *Handbook of Data Visualization*, pages 217–241. Springer, 2008.
- [Win99] William E. Winkler. State of statistical data editing and current research problems. In *UN/ECE Work Session on Statistical Data Editing, Working Paper n.29*, pages 2–4, 1999.
- [XGH06] Ling Xiao, John Gerth, and Pat Hanrahan. Enhancing visual analysis of network traffic using a knowledge representation. In *Visual Analytics Science And Technology, 2006 IEEE Symposium On*, pages 107–114. IEEE, 2006.