FAKULTÄT FÜR !NFORMATIK
Faculty of Informatics

Visual Analytics for Time Series Analysis

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Medizinische Informatik

eingereicht von

Markus Bögl
Matrikelnummer 0625252

an der
Fakultät für Informatik der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr. Peter Filzmoser
Ao.Univ.Prof. Mag. Dr. Silvia Miksch
Mitwirkung: Dipl.-Inf. Dr. Tim Lammarsch

Wien, 30.01.2013

(Unterschrift Verfasser) (Unterschrift Betreuung)

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Visual Analytics for Time Series Analysis

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Medical Informatics

by

Markus Bögl
Registration Number 0625252

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr. Peter Filzmoser
Ao.Univ.Prof. Mag. Dr. Silvia Miksch
Assistance: Dipl.-Inf. Dr. Tim Lammarsch

Vienna, 30.01.2013

(Signature of Author) (Signature of Advisor)

Erklärung zur Verfassung der Arbeit

Markus Bögl

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)

Acknowledgements

Primarily, I want to thank my advisor Peter Filzmoser and my co-advisor Silvia Miksch for their continued support, during the time I worked on this thesis. Despite their tight schedules, they were always there to give much valued advice and feedback. I would also like to express my gratitude to Tim Lammarsch, who helped me stay on track with his constructive criticism for the prototype and the thesis. I would like to thank all members of the HypoVis project team for their reviews and suggestions. I especially want to thank Alexander Rind for reading the thesis and giving valuable feedback and Wolfgang Aigner for his comments on my poster design.

Furthermore, I would like to thank Sandra Ghorbani who invested a lot of time and effort into proof-reading this document. I appreciate that a lot.

Finally, I dedicate this thesis to my family and all my friends. Without them I would not be where I am now.

Abstract

Time series analysis is a challenging task performed by domain experts in various fields, such as epidemiology, medical signal processing, neurophysiology, environmental sciences, and some research fields where biological data is analyzed. Time series for these fields are often unequally spaced and occasionally include missing values, while standard methods for time series analysis often require equally spaced time series without missing values. To enable the application of the standard methods for model selection in the time domain, such as the Box-Jenkins methodology, these time series therefore have to be transformed. However, the statistical software tools that implement the methods and models for time series analysis lack the adequate intuitive and interactive visual interfaces to support the user with the transformation, imputation, the seamless integration of this time series modifications into the workflow of model selection, and the workflow itself. The goal of this thesis is to overcome these problems by investigating and identifying appropriate Visual Analytics methods for the problem domain, use the findings to design a Visual Analytics process and implement this process in a prototype. The evaluation of the results is done by applying the prototype to an example time series following defined use case scenarios. The evaluation shows that Visual Analytics is a way to overcome the problems mentioned above and to support the user with interactive visualizations and short feedback cycles in the process of time series transformation and model selection.

Kurzfassung

Zeitreihenanalyse wird von Experten vieler Forschungsdisziplinen durchgeführt. Sie kommt unter anderem in der Epidemiologie, der medizinischen Signalverarbeitung, der Neurophysiologie, den Umweltwissenschaften und bei Anwendungen mit biologischen Daten zum Einsatz. Zeitreihen sind in diesen Bereichen häufig nicht äquidistant oder enthalten fehlende Werte, wobei Standardmethoden meist äquidistante Zeitreihen ohne fehlende Werte voraussetzen. Um hier dennoch eine Anwendung der Methoden, wie etwa der Box-Jenkins Methode, zu ermöglichen, müssen die Zeitreihen transformiert werden. Gängige Softwaretools für Zeitreihenanalyse, die solche Methoden und Modelle implementieren, verfügen oft nur über eine mangelhafte Unterstützung durch intuitive und interaktive visuelle Darstellungen für die Aufgaben der Transformation, der Imputation, der nahtlosen Integration im Prozess der Modellselektion sowie der Box-Jenkins Methode. Ziel der vorliegenden Arbeit ist es, diese Schwachpunkte auszugleichen, indem Visual Analytics Methoden im Problemfeld untersucht und identifiziert werden, anhand der gewonnenen Erkenntnisse ein Visual Analytics Prozess formuliert und als Prototyp implementiert wird. Zur anschließenden Evaluierung wird der Prototyp für definierte Anwendungsfälle mit Hilfe von Beispieldaten getestet. Die Szenarien zeigen, dass Visual Analytics Methoden zur Lösung der beschriebenen Probleme geeignet sind und den Benutzer/die Benutzerin zusätzlich mit interaktiven Visualisierungen und kurzen Feedbackzyklen bei der Problemstellung von Zeitreihen-Transformation und Modellselektion unterstützen können.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

Statistical time series analysis is a challenging task used by many experts in different domains. The dataset used for the analysis is obtained from observations collected over time, optimally at periodic and equally spaced intervals and ideally without missing values. This dataset is called a time series. A range of methods, algorithms, and models to analyze these time series exist in the literature. Moreover they are implemented in most common software tools for statistical computing. In this thesis we focus on one popular methodology for time series analysis known as Box-Jenkins methodology [Box and Jenkins, 1970]. We present and discuss this methodology and the theoretical underpinnings in Chapter 2. It is still challenging for domain experts to work with the software tools for time series analysis, because these do not provide an appropriate support for the workflow of the Box-Jenkins methodology.

A potential way to overcome these shortcomings, is the new research field of Visual Analytics that aims to "combine automated analysis techniques with interactive visualizations" [Keim et al., 2010]. In Chapter 3 we identify an existing general Visual Analytics process and a more specific Visual Analytics process for time-oriented data. As an interesting property of time-oriented data, we describe the concept of time granularities. From a comprehensive survey of visualization techniques for time-oriented data, we distill the relevant techniques for the problem domain of time series analysis. We discuss relevant related tools to outline the difference to our solution and identify in particular one recent contribution to the domain of Visual Analytics for time series preprocessing by Bernard et al. [2012].

These findings raise the question of how to use Visual Analytics methods to support the process of model building for time series analysis. A further question is how it can help to determine the best transformation for unequally spaced time series and missing values to allow the application of the Box-Jenkins methodology. We formulate and discuss the problem statement, the research question, and the motivation in more detail in Chapter 4.

The main objective of this thesis is to define a Visual Analytics process to tackle the problems stated briefly above. Based on this process we implement a prototype that supports the user in transforming time series, to cope with missing values, and supports the process of model building with Visual Analytics methods. We present the scientific approach to achieve this in Chapter 5.

The result is the definition of a Visual Analytics process that describes how the human reasoning and automated methods are combined to overcome the stated problems. Another result is the implementation of this process as a prototype, named VisuTimAlytics. For the implementation we use the insights from studying the Visual Analytics methods and the problem domain. We provide and discuss these results in Chapter 6. To evaluate the usability of the prototype and therefore the practicability of the Visual Analytics process, we define use case scenarios for the application of the prototype using an example time series. For this time series we choose a dataset from the domain of environmental epidemiology. We present this evaluation in Chapter 7. Throughout the thesis we use the example dataset that we introduce in Section 7.1 to generate most of the example plots and screenshots of the software tools and the prototype. If not otherwise stated we refer to Section 7.1 for the details on this dataset. In Chapter 8 we discuss the conclusions and sum up the thesis. With the overall conclusion, that Visual Analytics is useful to support the process of model building for time series analysis and helps to determine the best transformation for unequally spaced time series and missing values, we argue that the research hypotheses are valid, and we provide the answer to the main research question stated before. We end the thesis with an outlook on future work, that could further enhance the process and the prototype.

The thesis was done as part of the HypoVis project[1] and was supported by the Austrian Science Fund (FWF) project number: P22883.

[1] `http://www.ifs.tuwien.ac.at/~lammarsch/HypoVis` (07.01.2013)

CHAPTER 2

Time Series Analysis

In this chapter we discuss one popular state of the art method for time series analysis and the class of models used in this method. As stated briefly in Chapter 1 and in more detail in Chapter 4 and 5, we need to have a good understanding of the target problem to define and implement a Visual Analytics process and to define the design requirements for the prototype. Chapter 2 provides this understanding.

After explaining some basic characteristics and definitions in time series analysis in Section 2.1, we discuss the popular Box-Jenkins methodology for time series analysis in Section 2.2, which was introduced by Box and Jenkins [1970] and is used in most of the current (2012) textbooks on time series analysis [Box et al., 2008; Hamilton, 1994; Cryer and Chan, 2008; Bisgaard and Kulahci, 2011; Shumway and Stoffer, 2011]. An important consideration in time series analysis is how to handle unequally spaced time series and missing values in time series, which is discussed in Section 2.3. To apply the Box-Jenkins methodology, we introduce the models we are likely to use in our Visual Analytics process in Section 2.4, namely *autoregressive moving average* (ARMA) models, *autoregressive integrated moving average* (ARIMA) models, and *seasonal autoregressive integrated moving average* (SARIMA) models. Thereafter we discuss the theoretical background of the particular steps in the model selection process. The first of these steps is the model specification or model selection in Section 2.5. Next is the model fitting or parameter estimation in Section 2.6. Then there is the model diagnostics in Section 2.7. Finally we present important tools for time series analysis in Section 2.8 and discuss the support of time series analysis and the model building process in these tools.

The characteristics, definitions, descriptions, and models in time series and time series analysis we present in the next sections are based on the work of Box et al. [2008], Brockwell and Davis [1991], Cryer and Chan [2008], Shumway and Stoffer [2011], Hamilton [1994], Bisgaard and Kulahci [2011], Brockwell and Davis [2002], Cowpertwait and Metcalfe [2009], Pfaff [2008], and Brockwell [2011], where most of them are based on the landmark work of Box and Jenkins [1970]. There are some different notations used in the various textbooks about time series analysis. We mainly use the same notation as in Shumway and Stoffer [2011].

2.1 Characteristics and Definitions

A *time series* is a set of sequentially measured observations over time. Depending on whether the set is continuous or discrete, the time series is said to be *continuous* or *discrete*. The methods and models we present here, are based on discrete time series with equidistant time intervals h. The resulting set of times can be determined through the formula $T = t_0+h, t_0+2h, ..., t_0+Nh$, where N is the number of observations and h the *sampling interval*.

A discrete time series is a sequence of random variables x_t observed at times t. Such a collection of random variables is generally referred to as a *stochastic process*, which is defined as "a statistical phenomenon that evolves in time according to probabilistic laws" [Box et al., 2008]. The time series is said to be a *realization* of the process. The term *time series* is used to refer to the process or to the realization of the process and usually it is clear from the context of the discussion which is meant.

Assuming a discrete time series is convenient in regard to analysis techniques using computers, observed time series are usually discrete because of the method of collection by sampling a continuous time series or by accumulating a variable over a period of time [Shumway and Stoffer, 2011].

Basic Definitions

There are two essential operators that we need to explain the models. These are the *backshift operator* and the *differences operator*.

Definition 2.1. *The **backshift operator**, B, is defined as*

$$B^k x_t = x_{t-k} \tag{2.1}$$

where k is the time shift or lag.

Another important operator is the *backward difference* operator, defined as

$$\nabla x_t = x_t - x_{t-1} = (1 - B)x_t \,.$$

Definition 2.2. *The **differences operator, ∇, of order d**, is defined as*

$$\nabla^d = (1 - B)^d \,. \tag{2.2}$$

Stationarity

The goal of time series analysis is to describe the structure of the stochastic process with a mathematical model. If the underlying physics of the process are fully known, the model could describe the process and correctly predict future observations. These models are called *deterministic* models. Usually time series are based on time-dependent phenomena with many unknown factors. It is not possible to find a deterministic model fully describing them, but it is possible to derive a model that is able to describe and predict the process with a certain probability and confidence interval, which is called a *probability* or *stochastic* model.

To derive a stochastic model, there has to be some sort of regularity in the behavior of a time series. This characteristic of a time series is called *stationarity*, which is a basic requirement for any time series analysis.

Definition 2.3. *A time series is **strictly stationary**, if the joint distribution function of a set of k observations*

$$\{x_{t_1}, x_{t_2}, \ldots, x_{t_k}\}$$

remains the same for another set of k observations shifted by h time units

$$\{x_{t_1+h}, x_{t_2+h}, \ldots, x_{t_k+h}\}$$

for all $k = 1, 2, \ldots$, all time points $t_1, t_2, \ldots, t_k$ and all time shifts $h = 0, \pm 1, \pm 2, \ldots$, and can therefore be described as

$$P\{x_{t_1} \leq c_1, \ldots, x_{t_k} \leq c_k\} = P\{x_{t_1+h} \leq c_1, \ldots, x_{t_k+h} \leq c_k\}.$$

Because the definition of strict stationarity is too strong for most time series, the term *stationary time series* usually means the weaker definition. To explicitly refer to the strict stationary definition, the term *strict stationary time series* is used.

Definition 2.4. *A **weak stationary** time series is a finite variance process where*

(i) the mean value function, μ_t, is constant and independent of time t, and

(ii) the autocovariance function, $\gamma(s,t)$, (see below) depends only on s and t based on the difference $|s-t|$.

For stationary time series the mean function is defined as

$$\mu_x = E(x_t) = \int_{-\infty}^{\infty} x f(x)\, dx\,, \tag{2.3}$$

because by definition a stationary process implies the same probability distribution $f_t(x)$ for all times t, written as $f(x)$. Therefore $\mu_t = \mu$ is constant and can be estimated by the sample mean

$$\overline{x} = \frac{1}{n}\sum_{t=1}^{n} x_t\,. \tag{2.4}$$

By definition the variance is constant too,

$$\sigma_x^2 = E\left[(x_t - \mu)^2\right] = \int_{-\infty}^{\infty} (x-\mu)^2 f(x)\, dx \tag{2.5}$$

and is estimated by the sample variance of the time series

$$\hat{\sigma}_x^2 = \frac{1}{n}\sum_{t=1}^{n} (x_t - \overline{x})^2\,. \tag{2.6}$$

The definition of a stationary process implies that the joint probability distribution $f(x_{t_1}, x_{t_2})$ for all times t_1, t_2, with constant interval is the same. The covariance of values x_{t_1} and x_{t_2} with k intervals of time in between, named *lag k*, is called *autocovariance* at lag k.

Definition 2.5. *The **autocovariance function** is defined as*

$$\gamma(s,t) = cov(x_s, x_t) = E[(x_s - \mu_s)(x_t - \mu_t)] \tag{2.7}$$

for all s and t.

Definition 2.6. *The **autocorrelation function (ACF)** is defined as*

$$\rho(s,t) = \frac{\gamma(s,t)}{\sqrt{\gamma(s,s)\gamma(t,t)}}\,. \tag{2.8}$$

Because of Definition 2.4 the autocovariance function depends on s and t only based on their difference $|s-t|$. We denote this difference as k, which is called time shift or lag. For that reason, the notation can be simplified to

$$\gamma(k) = cov(x_{t+k}, x_t) = cov(x_k, x_0) = \gamma(k,0)$$

where zero is dropped, so that we have $\gamma(k)$, also written as γ_k.

Definition 2.7. *The **autocovariance function of a stationary time series** is defined as*

$$\gamma(k) = cov(x_{t+k}, x_t) = E\left[(x_{t+k} - \mu)(x_t - \mu)\right]\,.$$

The *autocorrelation* at lag k is defined in the same way as for the autocovariance.

Definition 2.8. *The **autocorrelation function (ACF) of a stationary time series** is defined as*

$$\rho(k) = \frac{\gamma(t+k,t)}{\sqrt{\gamma(t+k,t+k)\gamma(t,t)}} = \frac{\gamma(k)}{\gamma(0)}\,.$$

where $\rho(k)$ is also written as ρ_k.

An additional way for describing the correlation between two points is the *partial autocorrelation function*, where the linear effects of the points in between are removed.

Definition 2.9. *The **partial autocorrelation function (PACF)** is defined as*

$$\phi_{11} = corr(x_{t+1}, x_t) = \rho(1)$$

and

$$\phi_{kk} = corr(x_t - \hat{x}_t, x_{t+k} - \hat{x}_{t+k}), \quad k \geq 2\,.$$

For the formal definition to remove these linear effects, $\hat{x}$ is defined for $k \geq 2$ as the regression on the elements in between. The coefficients β are by definition of stationarity the same in both cases,

$$\hat{x}_t = \beta_1 x_{t+1} + \beta_2 x_{t+2} + \cdots + \beta_{k-1} x_{t+1}$$

and

$$\hat{x}_{t+k} = \beta_1 x_{t+k-1} + \beta_2 x_{t+k-2} + \cdots + \beta_{k-1} x_{t+1}\,.$$

Because $\gamma_k = \rho_k \sigma_x^2$, a normal stationary process is completely characterized by its mean μ and its autocovariance function γ_k, or by its mean μ, variance σ_x^2, and autocorrelation function ρ_k.

Definition 2.10. *A **linear process** is a linear combination of white noise variates* w_t

$$x_t = \mu + \sum_{j=-\infty}^{\infty} \psi_j w_{t-j}, \qquad \sum_{j=-\infty}^{\infty} |\psi_j| < \infty\,. \tag{2.9}$$

White noise is perhaps the most fundamental example of a stationary process. It is a collection of uncorrelated random variables, w_t, with mean 0 and finite variance σ_w^2. *White independent noise (iid)* are independent and identically distributed random variables and *Gaussian white noise* are independent normal random variables, both with mean 0 and variance σ_w^2.

Forecasting

According to Box et al. [2008], forecasting is one important area of application for a time series model. Bisgaard and Kulahci [2011] stated that it is one of the main goals in time series modeling to make forecasts about the future. If an adequate model is found based on the time series from current and past values it is possible to use this model to predict future values [Box et al., 2008; Bisgaard and Kulahci, 2011]. A methodology to find an adequate model of the classes in Section 2.4 is described in the following section.

2.2 Box-Jenkins Methodology

The Box-Jenkins methodology, also called Box-Jenkins process or Box-Jenkins approach, is an iterative model building process to select an adequate model for a given time series. This methodology was first introduced in the landmark work of Box and Jenkins [1970]. Since then it has been widely used in time series analysis and is the method for model selection used in current (2012) textbooks about time series analysis (see for example, [Box et al., 2008; Bisgaard and Kulahci, 2011; Cryer and Chan, 2008; Shumway and Stoffer, 2011]). The basics of the model building process for time series analysis we present in this section are based on the work of Box et al. [2008]. The problem in finding or selecting a model that describes a given time series, is due to the underlying physical mechanisms of a phenomenon. If these mechanisms would be fully known and understood, we could give an exact mathematical expression, a theoretical model, to describe that phenomenon. Because that is not possible for time series, it is necessary to

(1) use the incomplete theoretical knowledge about that mechanisms and the experience from theory and practice to consider a useful general class of models. Fitting these models directly to data, would be too extensive.

(2) Therefore we apply methods to select an appropriate parsimonious subclass of models. These methods additionally give some rough estimates of the parameters in the model.

(3) In the next stage, the model is fitted to the data and its parameters are estimated.

(4) Finally the model is checked with diagnostics, to uncover possible lack of fit and find the cause.

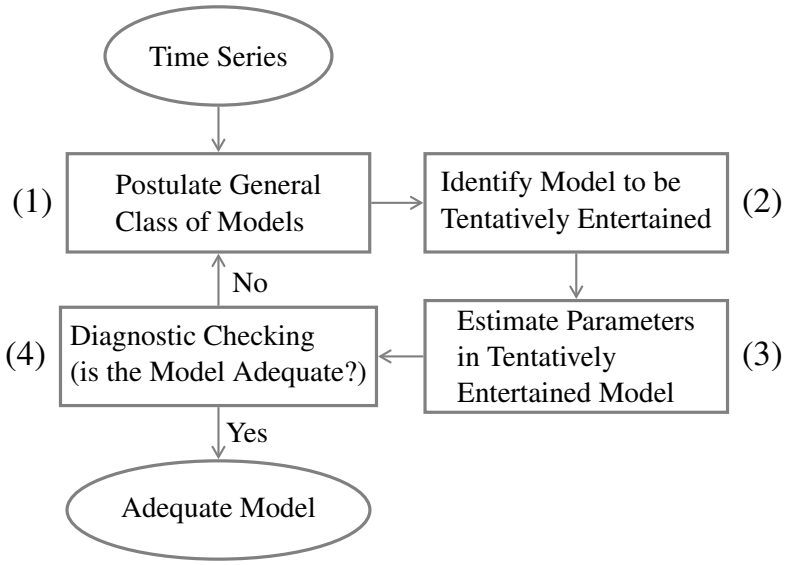

Figure 2.1: Original Box-Jenkins Methodology. An iterative process for model building of time series adapted from Box et al. [2008]. Based on the time series we decide on a general class of models (1). For the model classes we refer to Section 2.4. To identify a model that can be tentatively entertained (2), the order of the model and the level of difference are selected. Based on the time series the model parameters are estimated (3). The question whether the model is adequate is answered by diagnostic checking (4) of the residuals, which are the remaining parts that are not explained by the model. If the model is not adequate, the whole process is repeated with a different model configuration. Otherwise, if the model is adequate, it can be used for forecasting.

If an adequate model is found, the model is ready to use for forecasting, otherwise the iterative process of identification, estimation, and diagnostic checking is repeated until an adequate model is found. This method for model selection is a process with iterative stages, as shown in Figure 2.1.

In the textbooks about time series analysis this method is modified in different ways. In one version the steps are named with more detail to match the task of model selection for ARIMA models, which are presented in Section 2.4. The modified version is shown in Figure 2.2. It is also popular to reduce the original process and introduce a simplified version. This simplification is shown in Figure 2.3 along with the original process to make a direct comparison possible.

The crux of model selection is summarized in the famous quote of George Box that "Essentially, all models are wrong, but some are useful". When introducing the Box-Jenkins methodology, Box and Jenkins [1970] use a language that indicates that there is a "useful" and "adequate" model for a time series, but we can not assume that it is a "true" or "correct" model. That there is nothing definite about such a model is particularly obvious when using the term "tentatively entertaining a model" [Bisgaard and Kulahci, 2011].

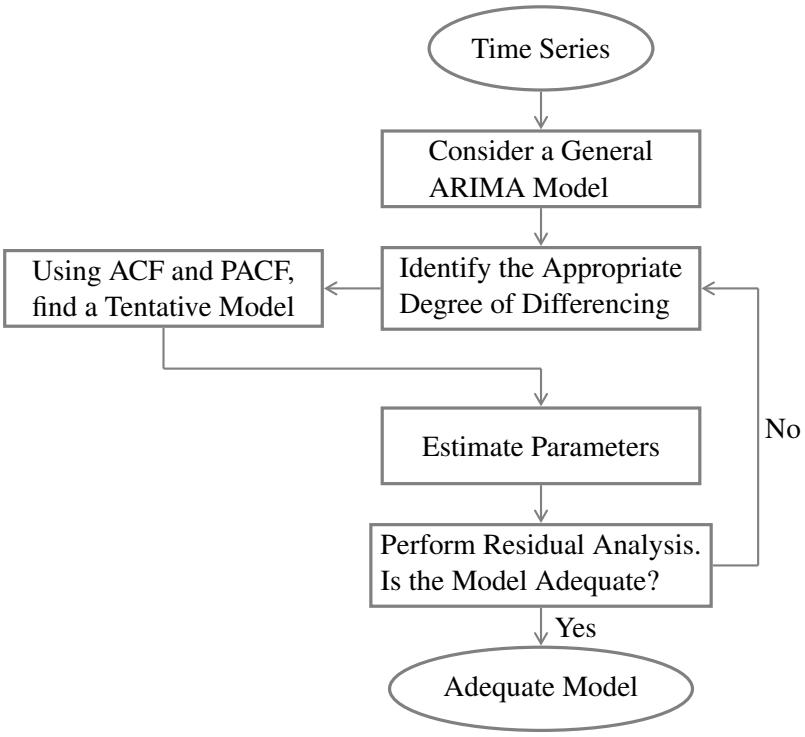

Figure 2.2: Modified Box-Jenkins Methodology for ARIMA models. Using the original process in Figure 2.1 as a basis, this process has been adapted by Bisgaard and Kulahci [2011] for ARIMA models. This figure displays an adapted version of their process. The acronyms ACF and PACF are the autocorrelation function and the partial autocorrelation function respectively. The definition and details about the ACF/PACF plot is presented in Section 2.5.

Parsimony

An important principle in the model selection process proposed by Box et al. [2008] is the principle of *parsimony*. The principle of parsimony is described best by quoting Albert Einstein "It can scarcely be denied that the supreme goal of all theory is to make the irreducible basic elements as simple and as few as possible without having to surrender the adequate representation of a single datum of experience." [Einstein, 1934], which is often paraphrased as "everything should be made as simple as possible but not simpler" [Bisgaard and Kulahci, 2011]. In the process of model selection this means that if there are different candidate models to adequately represent the time series, we prefer the model with the least parameters [Cryer and Chan, 2008].

For the model selection process "our objective, must be to obtain adequate but parsimonious models" [Box et al., 2008]. For an illustration to legitimate the principle of parsimony we refer to Box et al. [2008, p.16].

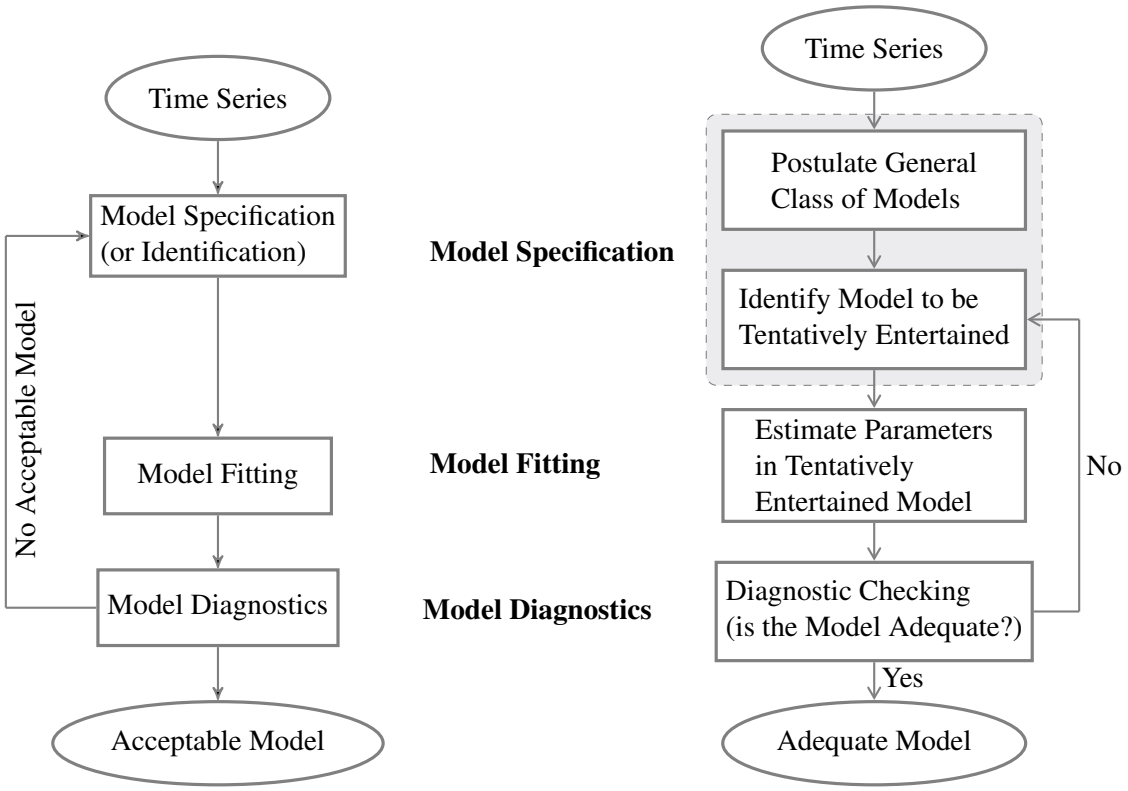

Figure 2.3: Simplified Box-Jenkins methodology along with the original method. The simplified version is proposed by Cryer and Chan [2008].

2.3 Unequally Spaced Time Series and Missing Values

According to Jones [1985], unequally spaced time series are either caused by missing observations, or the observations have no underlying sampling interval and are truly unequally spaced. For unequally spaced time series it is possible to use state space representations of the process for data analysis [Jones, 1985]. These advanced methods are very complicated and complex, and in most cases, especially if there are only a few missing values at random positions, it is preferable to simply replace the missing values and proceed as usual. Another reason to estimate the missing values and proceed as usual is that for these time series well established methods and models can be applied. One way is to estimate the value at the affected position is by means of an adequately calculated average based on the observations before and after the affected position. Another way to estimate the missing values is to use interpolation techniques, such as spline regression [Bisgaard and Kulahci, 2011].

Visualization

Templ et al. [2012] introduced a collection of visualizations to explore incomplete data with missing values. The visualization techniques are implemented in the R package `VIM`[1]. The `VIM` package provides functionality to visualize, impute and analyze missing values within R. The visualization techniques in this package are a good starting point to get an idea on how to highlight missing values.

2.4 ARIMA and Seasonal ARIMA Models

With classical regression it is often not possible to explain a time series sufficiently. The regression model is limited because the dependent variable is influenced only by the current independent variables, and not by past independent variables or its own past values, which would be desirable in time series [Shumway and Stoffer, 2011]. Therefore alternative models exist. One class of models is the *autoregressive moving average* (ARMA) model class. This class of models include the subclasses for *autoregressive* (AR) models, *moving average* (MA) models, and the combination of AR and MA models. It is possible to apply this class of models if the time series is stationary, which means that there is no seasonal effect or trend. An extension of this class of models is the *autoregressive integrated moving average* (ARIMA) model class. Models from this class can deal with time series containing trends. *Seasonal autoregressive integrated moving average* (SARIMA) models are another extension and are able to cope with seasonal time series and trends. The details about these models are introduced in the following.

Autoregressive (AR) Models

Autoregressive models are models, where the current value x_t is explained as a function of a number p of past values $x_{t-1}, x_{t-2}, \ldots, x_{t-p}$.

Definition 2.11. *An **autoregressive model** of order p, **AR(p)**, is defined as*

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + w_t \tag{2.10}$$

where x_t is stationary, $\phi_1, \phi_2, \ldots, \phi_p$ are constants, with $\phi_p \neq 0$, w_t is assumed to be Gaussian white noise with a mean value of zero and variance σ_w^2. The mean value of $x_t = 0$ or otherwise x_t is replaced with $x_t - \mu$. This allows to rewrite the definition of the autoregressive model as

$$x_t - \mu = \phi_1(x_{t-1} - \mu) + \phi_2(x_{t-2} - \mu) + \cdots + \phi_p(x_{t-p} - \mu) + w_t$$

or

$$x_t = \alpha + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + w_t \tag{2.11}$$

where $\alpha = \mu(1 - \phi_1 - \cdots - \phi_p)$.

Definition 2.12. *The **autoregressive operator**, ϕ, is defined as*

$$\phi(B) = 1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p \,. \tag{2.12}$$

[1] `http://cran.r-project.org/web/packages/VIM` (09.01.2013)

Using the backshift and the autoregressive operator it is possible to write the AR(p) model in a more compact form

$$(1 - \phi_1 B - \phi_2 B^2 - \cdots - \phi_p B^p) x_t = w_t \tag{2.13}$$

or

$$\phi(B) x_t = w_t \,. \tag{2.14}$$

Moving Average (MA) Models

Moving average models are models where the current value x_t is explained as a linear combination of the current white noise term and the q past white noise terms.

Definition 2.13. *A **moving average model** of order q, **MA(q)**, is defined as*

$$x_t = w_t + \theta_1 w_{t-1} + \theta_2 w_{t-2} + \cdots + \theta_q w_{t-q} \tag{2.15}$$

where w_t is Gaussian white noise with mean value zero and variance σ_w^2, there are q lags in the moving average and $\theta_1, \theta_2, \ldots, \theta_q (\theta_q \neq 0)$ are parameters.

Definition 2.14. *The **moving average operator** is defined as*

$$\theta(B) = 1 + \theta_1 B + \theta_2 B^2 + \cdots + \theta_q B^q. \tag{2.16}$$

Using the backshift and the moving average operator, the MA(q) model is written as

$$x_t = \theta(B) w_t, \tag{2.17}$$

The moving average process is stationary because it is a finite sum of stationary white noise terms.

Autoregressive Moving Average (ARMA) Models

In some cases it is not possible to model a time series with only AR or MA models, because it would demand a high-order model with many parameters. This is in conflict with the principle of parsimony as introduced in Section 2.2. For these cases, Box and Jenkins [1970] presented *autoregressive moving average* (ARMA) models. To achieve parsimony, ARMA models combine the ideas of AR and MA models.

Definition 2.15. *A general **autoregressive moving average model** of order p and q, **ARMA(p, q)**, is defined as*

$$x_t = \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p} + w_t + \theta_1 w_{t-1} + \cdots + \theta_q w_{t-q} \tag{2.18}$$

where $\phi_p \neq 0$, $\theta_q \neq 0$, $\sigma_w^2 > 0$, and w_t is a Gaussian white noise with mean value zero and variance σ_w^2, unless otherwise stated. The parameter p is the autoregressive order and q the moving average order. If the mean value μ of x_t is nonzero, we add $\alpha = \mu(1 - \phi_1 - \cdots - \phi_p)$ and write the model as

$$x_t = \alpha + \phi_1 x_{t-1} + \cdots + \phi_p x_{t-p} + w_t + \theta_1 w_{t-1} + \cdots + \theta_q w_{t-q} \,. \tag{2.19}$$

For a general ARMA(p, q) model with $q = 0$, the model is called an autoregressive AR(p) model of order p. With the parameter $p = 0$, the model is called a moving average MA(q) model of order q. Using the AR operator and the MA operator, the Formula (2.18) can be written as

$$\phi(B)x_t = \theta(B)w_t \tag{2.20}$$

With this general definition of ARMA(p, q) models, there are a number of problems regarding (1) parameter redundant models, (2) stationary AR models that depend on the future, and (3) MA models that are not unique, as pointed out by Shumway and Stoffer [2011]. To solve these problems they suggest some additional restrictions on the parameters, see Shumway and Stoffer [2011, p. 94–96].

Autoregressive Integrated Moving Average (ARIMA) Models

In many practical cases, a time series is non-stationary due to seasonal effects or trends. It is possible to transform this time series to stationary time series by applying by differencing, sometimes called detrending. To recover the original time series, the differenced time series need to be aggregated, or also called integrated. These models are called *autoregressive integrated moving average* (ARIMA) models.

Definition 2.16. *A process x_t is said to be **ARIMA(p, d, q)** if the dth difference of x_t*

$$\nabla^d x_t = (1 - B)^d x_t$$

is ARMA(p, q). In general the model is written as

$$\phi(B)(1 - B)^d x_t = \theta(B)w_t. \tag{2.21}$$

If $E(\nabla^d)x_t = \mu$, the model is written as

$$\phi(B)(1 - B)^d x_t = \delta + \theta(B)w_t.$$

where $\delta = \mu(1 - \phi_1 - \cdots - \phi_p)$.

Seasonal Autoregressive Integrated Moving Average (SARIMA) Models

To include seasonal terms in a model it is necessary to multiplicative combine an ordinary non-seasonal ARIMA model with an ARIMA model that is extended to the seasonal period s. Therefore the AR and the MA operator are extended to the lags of the seasonal period s.

Definition 2.17. *The **seasonal autoregressive operator** of order P, with seasonal period s, is defined as*

$$\Phi_P(B^s) = 1 - \Phi_1 B^s - \Phi_2 B^{2s} - \cdots - \Phi_P B^{Ps} \tag{2.22}$$

*and the **seasonal moving average operator** of order Q, with seasonal period s, is defined as*

$$\Theta_Q(B^s) = 1 + \Theta_1 B^s + \Theta_2 B^{2s} + \cdots + \Theta_Q B^{Qs} \tag{2.23}$$

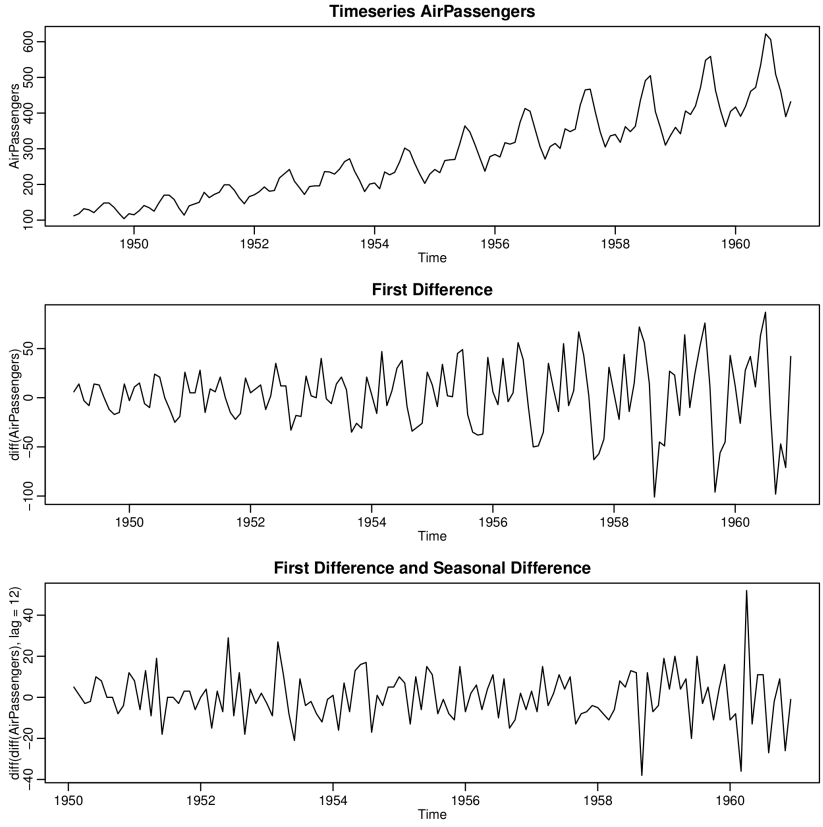

Figure 2.4: Air Passengers Time Series Plots. The time series is plotted as a function of values over time. The first chart shows the plot of the raw data, the second shows the first difference on lag 1, and the third shows the first difference on lag 1 and the first seasonal difference, meaning the difference on the lag of the seasonal length which is in this case 12. This figure displays the air passengers dataset by [Box et al., 1976].

The seasonal difference is also applied like the non-seasonal difference, but to the lags equal to the seasonal period s. This removes the additive seasonal effects. The resulting models are called *seasonal autoregressive integrated moving average* (SARIMA) models.

Definition 2.18. *A multiplicative* ***seasonal autoregressive integrated moving average*** *model, denoted as* $\boldsymbol{ARIMA(p,d,q) \times (P,D,Q)_s}$*, is defined as*

$$\Phi_P(B^s)\phi(B)\nabla_s^D\nabla^d x_t = \delta + \Theta_Q(B^s)\theta(B)w_t \tag{2.24}$$

where w_t is the Gaussian white noise. The ordinary difference component represents $\nabla^d = (1-B)^d$ and the seasonal difference $\nabla_s^D = (1-B^s)^D$.

2.5 Model Specification

In Section 2.2 we introduced the Box-Jenkins methodology. As stated there, it is possible to simplify the process to the separate steps *Model Specification*, *Model Fitting*, and *Model Diagnostics*. In Section 2.5, 2.6, and 2.7 we present a more detailed description of these separate steps in the model building process.

For the task of model specification, the goal is to decide on a class of models that could be appropriate for the given time series, select the level of differencing and determine the order of the model. The order of the model specifies the number of parameters used in the model. The first step to achieve this goal is to take a look at the given time series. Usually this is done by viewing the time series plot, as shown in Figure 2.4. After applying all required transformations, such as a difference operation or log transformation, the ACF/PACF plot is checked to support the decision of the model order.

The example plots provided in the following sections are based on the air passengers dataset by Box et al. [1976]. This dataset is more representative for explaining the behavior of the ACF and PACF plots than the example data introduced in Section 7.1.

Autocorrelation Function (ACF) and Partial Autocorrelation Function (PACF)

The *autocorrelation function (ACF) plot* is a spike graph, which is a special sort of bar chart, of the ACF γ_k, see Definition 2.8, as a function over lag k. The *partial autocorrelation function (PACF) plot* is likewise the PACF ϕ_{kk}, see Definition 2.9, as a function over lag k. As mentioned in the definition, the PACF is the correlation between two points where the linear effects of the points in between is removed. This PACF plot combined with the ACF plot, called ACF/PACF plot, where this linear dependence is included, enables us to choose the number of parameters for the model. The ACF/PACF plot in addition to the time series plot, also provide a first idea for the level of difference and seasonal difference. In Figure 2.5 we show the ACF/PACF plot. The ACF and PACF, as defined in Definitions 2.8 and 2.9, are plotted on the y-axes, and the lags on the x-axes. In this case the labels are seasonal lags, which means that one lag is one seasonal cycle. The inner seasonal lags are fractions of one, depending on the seasonal length. For example, in a dataset with 12 months in one year and seasonal length of 12, the seasonal lags are $1, 2, \ldots$ and the inner seasonal lags are $\frac{1}{12}, \frac{2}{12}, \ldots$.

To make it more clear, how the level of difference and the order of the model is chosen, we use the air passengers dataset by Box et al. [1976] to give a short example. The first plot in Figure 2.4 shows the raw time series without transformations or differences, and Figure 2.5 shows the ACF/PACF plot. The decision that we need the first difference of the time series is based on (1) the time series plot, because the time series is apparently not stationary, as well as on (2) the ACF/PACF plot, because the high values of the ACF and the slow decrease in addition to the cut in the PACF after the first lag are based on the linear dependence between the ACF lags. By using the first difference, this dependency is removed and the time series plot is more likely stationary, which is visible in the second plot of Figure 2.4. In Figure 2.6 the ACF/PACF plot of the first difference of the time series is shown. If we take a close look at the seasonal lags, which are the bars on the x-axes in positions $1, 2, \ldots$, we can observe the same behavior as before, only this time on the seasonal lags. This indicates that we have to take the first seasonal difference

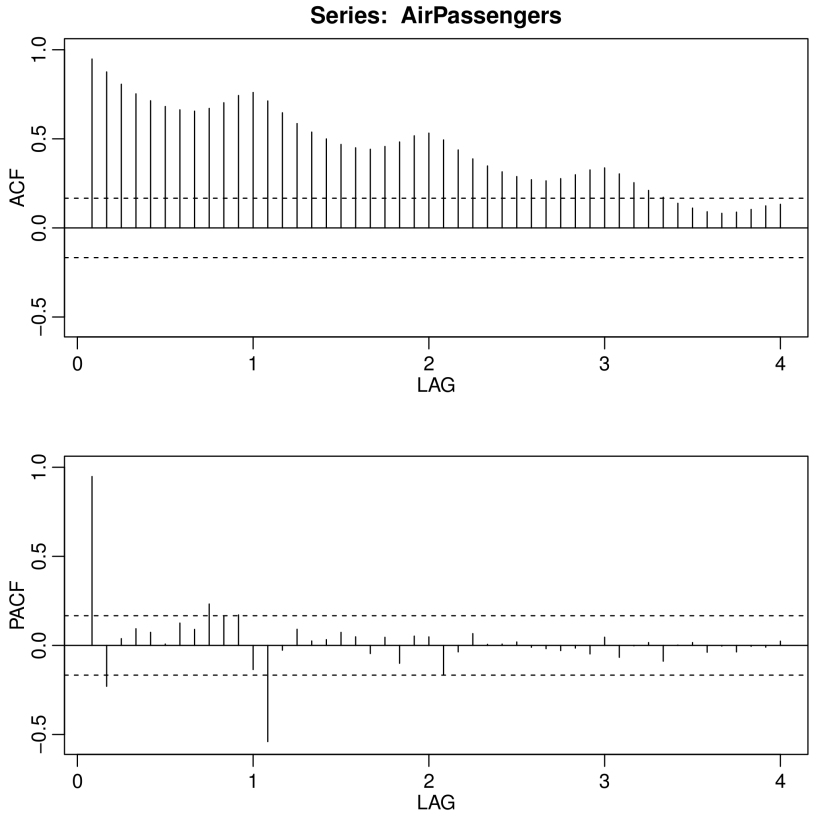

Figure 2.5: Autocorrelation and Partial Autocorrelation Function over lags. The behavior of the lags enables us to decide on the order of the model according to Table 2.1. This plot displays the air passengers dataset by [Box et al., 1976].

of the time series, as defined in Definition 2.18. The result of calculating the first difference and the first seasonal difference of the time series is shown in the last plot in Figure 2.4. According to Bisgaard and Kulahci [2011] this time series shows an increasing amplitude over time. We notice this also in the two plots of the difference and seasonal difference in Figure 2.4. They suggest to apply the natural logarithm before the difference operations to overcome that problem.

Using the definitions of the autoregressive models, moving average models, ACF, and PACF, Shumway and Stoffer [2011] show and prove the basic behavior of the ACF and the PACF for AR, MA, and ARMA models. The behavior is shown in Table 2.1. Likewise it is possible to describe the behavior for the seasonal component of the model in a similar way, which is shown in Table 2.2.

Information Criteria

Another way to decide if a model is worth to be considered, is to examine information criteria. To calculate these criteria, it is necessary to have the maximum likelihood estimation L_k for the

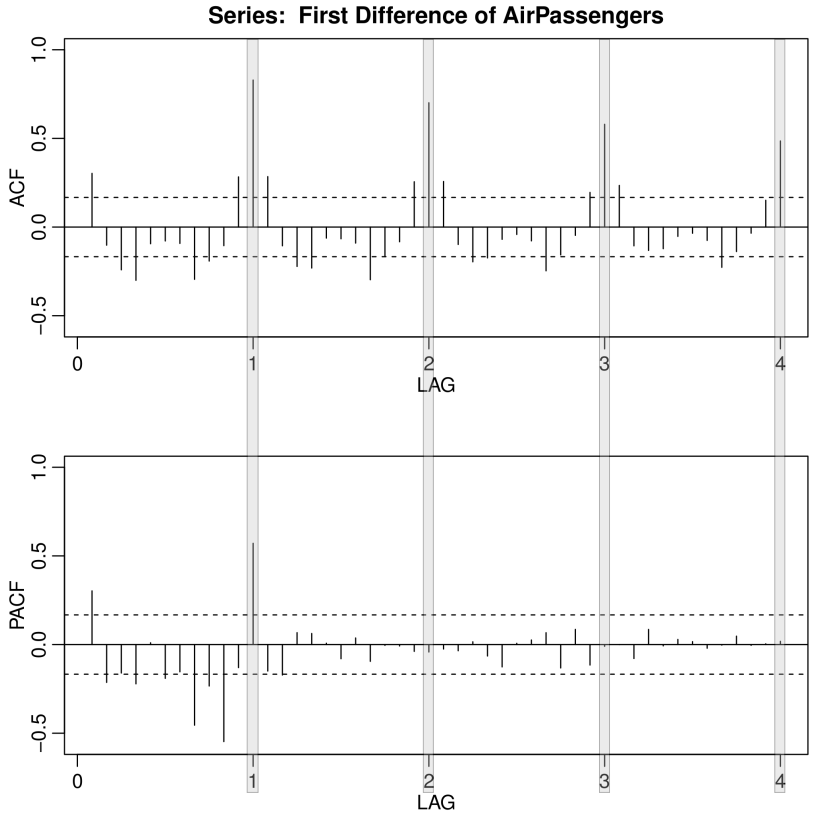

Figure 2.6: Seasonal lags in Autocorrelation and Partial Autocorrelation Function over lags. The seasonal lags are highlighted by the gray shade. Based on the behavior of these lags, the order of the seasonal component in the model is selected according to Table 2.2. This figure displays the air passengers dataset by [Box et al., 1976].

model, where k is the number of parameters. L_k can be determined by fitting the model to the time series and can therefore be seen as a diagnostic step or as a model selection step. More often than not it is categorized as a model diagnostic step, because the model parameters are estimated based on the time series. However we also take the information criteria into account to decide which model to choose. For this reason we present it here in the model selection section.

The first of the criteria that is often used is Akaike's information criterion (AIC).

Definition 2.19. ***Akaike's Information Criterion (AIC)***

$$AIC = -2 \log L_k + 2k \tag{2.25}$$

where L_k is the maximum likelihood estimation and k is the number of parameters in the model.

The model with the smallest AIC is considered to be the "best" model. Another criterion based on the AIC is the bias corrected form of the AIC.

	AR(p)	**MA(q)**	**ARMA(p, q)**
ACF	Tails off	Cuts off after lag q	Tails off
PACF	Cuts off after lag p	Tails off	Tails off

Table 2.1: ACF and PACF behavior for ARMA models [Shumway and Stoffer, 2011]. The behavior of the ACF and the PACF indicate which class of model and what number of parameters could be adequate.

	AR(P)$_s$	**MA(Q)$_s$**	**ARMA(P, Q)$_s$**
ACF*	Tails off at lags ks	Cuts off after lag Q_s	Tails off at lags ks
PACF*	Cuts off after lag P_s	Tails off at lags ks	Tails off at lags ks

*for nonseasonal lags $h \neq ks$, for $k = 1, 2, \ldots$, is zero.

Table 2.2: ACF and PACF behavior for SARMA models [Shumway and Stoffer, 2011]. The behavior of the ACF and the PACF indicate which class of model and what number of parameters could be adequate for the seasonal part of the model.

Definition 2.20. ***Bias Corrected Akaike's Information Criterion (AICc)***

$$AICc = AIC + \frac{2(k+1)(k+2)}{n-k-2} \tag{2.26}$$

where k is again the number of parameters in the model and n is the sample size.

In contrast to the AICc, which does behave very well for smaller samples, the next criterion is well suited for larger samples.

Definition 2.21. ***Bayesian Information Criterion (BIC)***

$$BIC = -2\log L_k + k\log(n) \tag{2.27}$$

where k is the number of parameters in the model and n is the sample size.

In order to get the "best" model, the goal is to determine k by selecting a number of parameters for the model, thus minimizing the criteria. Based on these values for different model configurations, it is possible to decide on one of the model configurations.

2.6 Model Fitting

In the previous section, we discussed how to decide on a class of models that we presented in Section 2.4. When deciding on a class of models, we also have to select the order of the model, which is also presented in Section 2.5. The result is a model, for example a seasonal ARIMA model as in Definition 2.18,

$$\Phi_P(B^s)\phi(B)\nabla_s^D\nabla^d x_t = \delta + \Theta_Q(B^s)\theta(B)w_t,$$

where the level of difference d, the level of the seasonal difference D, the seasonal length s, the number of parameters p and q, as well as the number of seasonal parameters P and Q are set according to the steps presented in the previous Section 2.5. Note that the parameters p and q determine the order of the model, which is the number of parameters in $\phi(B)$ and $\theta(B)$ as shown in Definition 2.15. Box et al. [2008] use the term *tentatively entertained model* for such a model. Once the model is identified, it is fitted to the time series data to estimate the unknown parameters of the model. There are several methods to estimate the parameters. The most important is the *maximum likelihood-estimation*. Other methods are *method of moments*, *least squares estimation*, and *unconditional least squares*. For details and theoretical discussion we refer to the textbook by Shumway and Stoffer [2011, p. 121–140]

2.7 Model Diagnostics

To evaluate how well the model represents the underlying time series, model diagnostic methods are applied. The most common method is to analyze the residuals, which means the remaining part that is not explained by the model. The exploratory analysis of the residuals is done by plots as shown in Figure 2.7. If the model is well fitted to the time series, the remaining part is expected to behave like white noise. This evaluation is done by plotting the residuals or the standardized residuals to check if the resulting plot looks like white noise, which means that it is a random process. This time series plot of the standardized residuals should also unveil any remaining underlying processes. The ACF of the residuals is calculated and plotted over the lags to check that there is no remaining structure in the residuals. White noise is standard normally distributed. If the model is well fitted, the standardized residuals are expected to be standard normally distributed too. This can be checked using the normal quantile-quantile plot [Cleveland, 1993], where the quantiles of the standardized residuals are plotted over the theoretical quantiles of the standard normally distribution. If the standardized residuals are approximately standard normal distributed, the points lie on a line $x = y$. The last graph commonly used is the plot of the Ljung-Box statistic [Box and Pierce, 1970; Ljung and Box, 1978]. This is a test that helps to check if the residuals for each lag are independent. In the plot of the Ljung-Box statistic we can confirm that no lag is significant within the boundary, and that it can therefore be assumed that there is no remaining autocorrelation within the residuals. If all this is fulfilled, the model is well specified, otherwise the model needs to be readjusted.

Another way to diagnose the model is the application of the information criteria presented in Section 2.5. In addition to the diagnostic plots discussed before, they provide a good basis to decide on the fitness of the model.

2.8 Software Tools for Time Series Analysis

In all major mathematical and statistical software tools the state of the art methods and models for time series analysis we described in previous sections are implemented. The most important

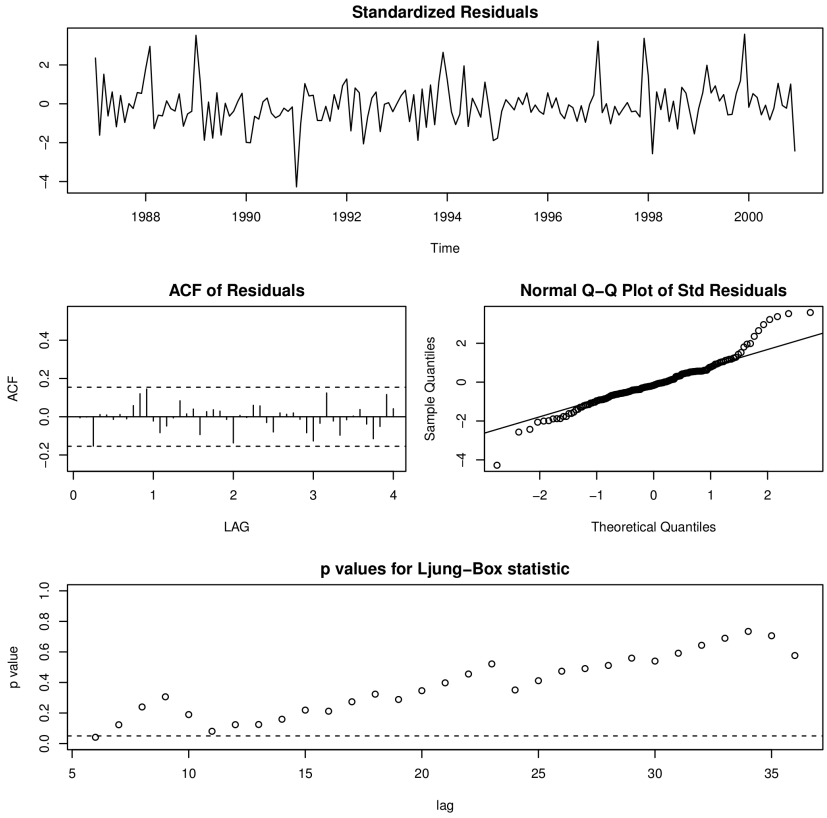

Figure 2.7: Diagnostic Plots, Residual Analysis. The remaining part that is not fitted by the model is checked to determine whether it is a random process. This is done by plotting the standardized residuals over time, the ACF of the residuals over the lags, the normal quantile-quantile plot of the standardized residuals, and the p values for the Ljung-Box statistic over lags. This is basically an explorative assessment for the properties of a random process.

tools are for example EVIEWS[2], Mathematica[3], MATLAB[4], and SAS[5]. With these tools a user is usually able to plot time series, browse the plot (zoom, select), plot ACF and PACF, calculate AIC, AICc, and BIC, fit a model to the time series, use an automatically selected model to fit to a time series mostly based on the AIC, AICc, BIC, and plot diagnostics, and carry out forecasting. These software features map to the separate steps of the Box-Jenkins methodology for model selection presented in Section 2.2. The separate steps are carried out by executing command line commands, where the parameters of the methods and models are inserted by the user, or if automatic model selection is supported, by the calculated criterion. The tools additionally have a graphical user interface, where user can execute commands using buttons and menu items, and

[2] `http://www.eviews.com` (09.01.2013)
[3] `http://www.wolfram.com/mathematica` (09.01.2013)
[4] `http://www.mathworks.com` (09.01.2013)
[5] `http://www.sas.com` (09.01.2013)

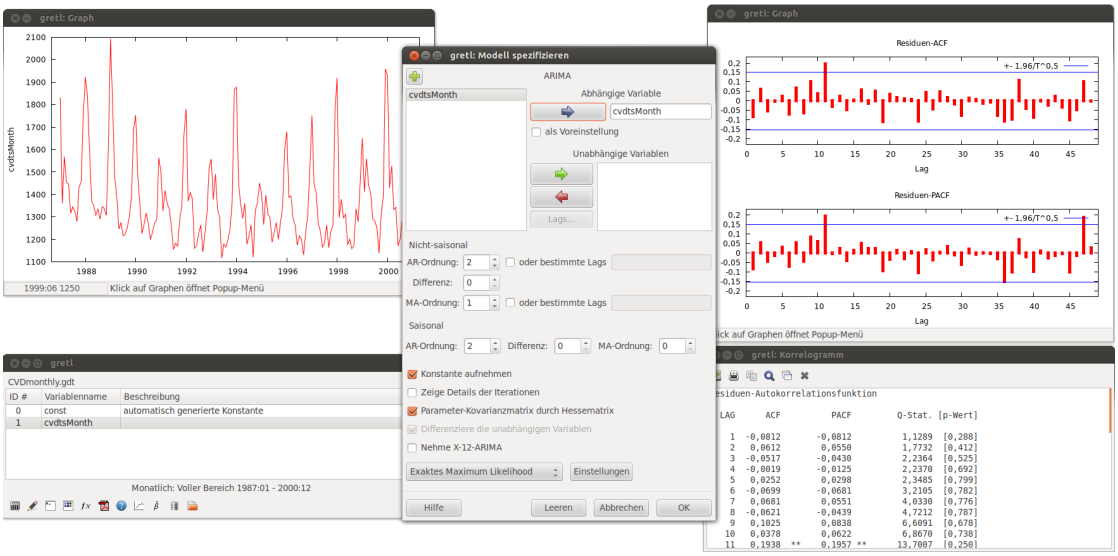

Figure 2.8: Statistical Software Tool Gretl – Model Specification Plots. The dataset for this plots is the example time series used for the evaluation in Chapter 7. Details about the data can be found in Section 7.1. The upper left window shows the time series line plot of the time series data. The upper right window shows the plotted ACF and PACF. The values of the single lags are displayed in the window on the lower right side. In the main window, which is in the center, the model is specified by selecting the parameter estimation algorithm and the order parameter of the model.

set parameters using input forms.

Furthermore, there are software packages available that implement specialized methods and models, for example, TRAMO/SEATS[6] and X-12-ARIMA[7]. These software packages usually do not have a graphical user interface and are used as command line tools only. However, most of the mathematical and statistical software tools mentioned above support the usage of these specialized implementations through their own graphical user interface. In this case it is possible to either use the methods and models implemented in the software tool or the ones implemented in one of the external packages.

Another tool is Gretl[8], which is a graphical user interface for the methods, models and implementations of external software packages. Gretl itself does not implement any methods for time series analysis, but can be connected to either R, TRAMO/SEATS or X-12-ARIMA. In Figure 2.8 Gretl is shown. The functionality is accessed by using the main menu and context menu. It is possible to import data from various sources using the menu. The loaded datasets are listed in the main window. After selecting a dataset, it is possible to apply different operations by using the main menu or the context menu. These operations include the transformation and calculation of time series, model fitting and displaying diverse plots. In Figure 2.8 some of the

[6] `http://www.bde.es/servicio/software/econom.htm` (09.01.2013)
[7] `http://www.census.gov/srd/www/x12a` (09.01.2013)
[8] `http://gretl.sourceforge.net/` (09.01.2013)

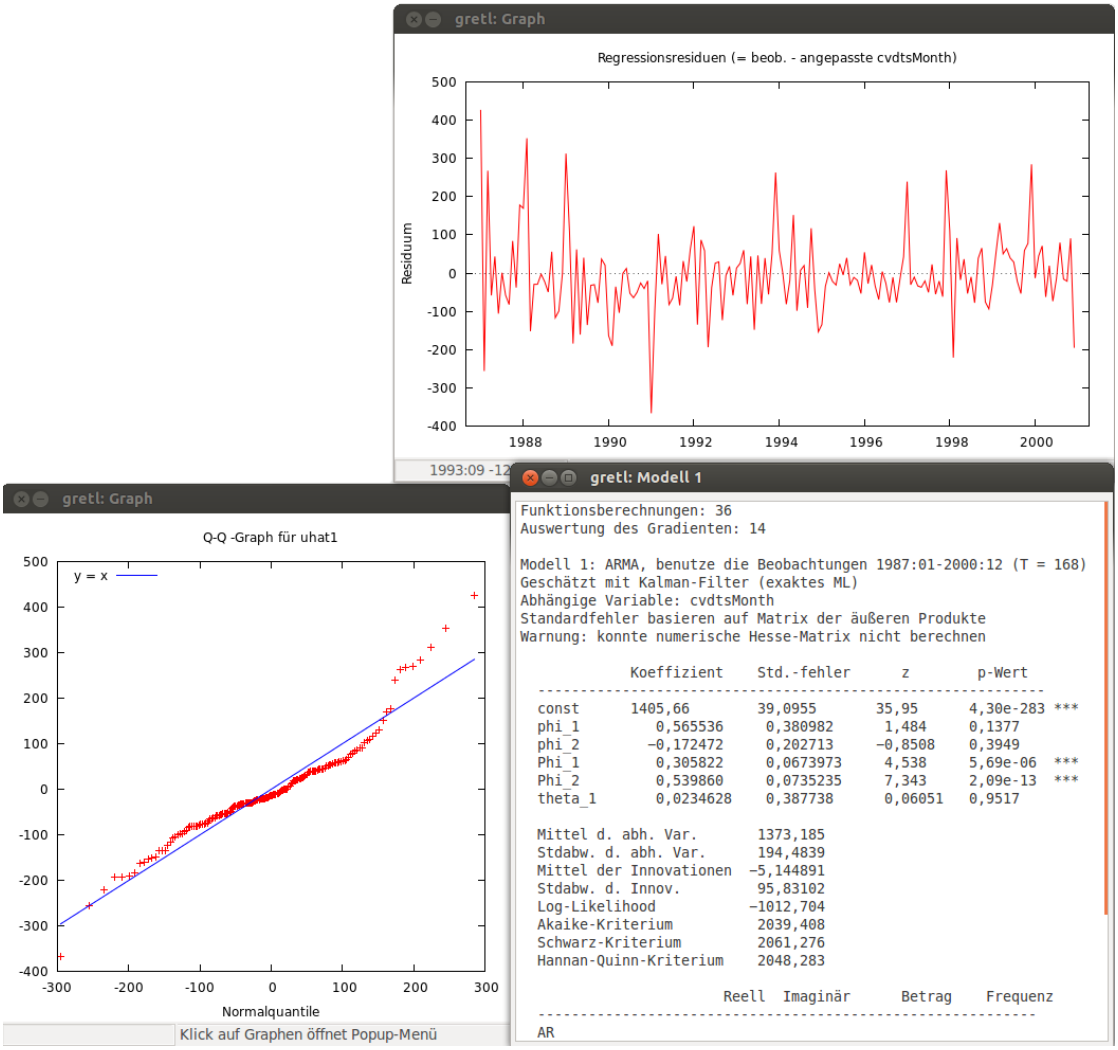

Figure 2.9: Statistical Software Tool Gretl – Results of Parameter Estimation. These are the results of the parameter estimation for a specific model. The top window shows the residual time series plot, the left window the normal quantile-quantile plot, and the bottom right window the parameter estimation and other results of the parameter estimation.

time series windows are shown. In Figure 2.9 the results of applying the parameter estimation of a specified model is shown. These diagnostic plots enable us to analyze the model candidate. In Figure 2.9 the residual time plot, the normal quantile-quantile plot and the estimated parameters are displayed.

The R project for statistical computing [R Development Core Team, 2012], plays an important role as a statistical software tool. R is heavily used at universities and in scientific research. It is an open source tool that is available for free and therefore very popular. Time series analysis is supported in R by different packages. A good overview is the task view for time series analy-

sis.[9] There are different algorithms implemented in different packages. Some textbooks for time series analysis provide modified functions or plots, such as the `TSA`[10] package by Cryer and Chan [2008] or the implementations by Shumway and Stoffer [2011] in the `astsa`[11] package. Because Gretl uses mainly R for the computation and plotting, it is possible in R to generate the same plots and outputs as shown before. The only difference is that the plots and outputs are generated by executing commands in the R command line or by executing R scripts.

Another important R package is the `x12GUI`[12] package. It provides an interactive graphical user interface for the `x12`[13] package, which provides a wrapper function to the X-12-ARIMA software. The user interface supports the user in selecting a time series and adjusting the parameters for the X-12-ARIMA calls. It also provides a history for parameter configurations, so that it is possible to load previous settings. The focus of the tool is to explore the time series and the results of the seasonal time series adjustment and to enable the user in interactive manual editing of outliers [Kowarik et al., 2012].

2.9 Summary

In this chapter we introduced to the field of statistical time series analysis in the time domain using ARIMA and seasonal ARIMA models. After outlining some basic characteristics and definitions, we have identified and presented the Box-Jenkins methodology, which is a well recognized and widely used method and an essential approach for using to solve the model selection problem in our Visual Analytics process and prototype implementation. We discovered the special challenges of unequally spaced time series and missing values. For the understanding of the Box-Jenkins methodology and the theoretical underpinnings, we presented the main models in the class of ARIMA and multiplicative seasonal ARIMA models. We discussed in more detail the separate steps of the Box-Jenkins methodology, namely the model specification, the model fitting, and the model diagnostics. To argue the contribution of our work and the difference to existing solutions, we showed related software tools for time series analysis and discussed their key features. We found that the related software tools for statistical computing lack in supporting the overall process and especially the workflow of this process in an intuitive and user friendly way. The investigation of the problem domain unveiled that it is necessary to have a good level of domain knowledge about statistical time series analysis to apply this process and understand and interpret the visual representations.

[9] `http://cran.r-project.org/web/views/TimeSeries.html` (09.01.2013)
[10] `http://cran.r-project.org/web/packages/TSA` (09.01.2013)
[11] `http://cran.r-project.org/web/packages/astsa` (09.01.2013)
[12] `http://cran.r-project.org/web/packages/x12GUI` (18.01.2013)
[13] `http://cran.r-project.org/web/packages/x12` (18.01.2013)

CHAPTER 3

Visual Analytics and Time-Oriented Data

In this chapter we first introduce the field of Visual Analytics followed by Visual Analytics processes in Section 3.1. For the visualization of time-oriented data it is necessary to consider their special properties and characteristics, which we present in Section 3.2. We discuss a systematic approach to visualization techniques in Section 3.3, present the selection of appropriate visualization techniques in Section 3.4 and related tools in Section 3.5.

One early definition of the new research field of Visual Analytics by Thomas and Cook [2005] is, that Visual Analytics is defined as "the science of analytical reasoning facilitated by interactive human-machine interfaces". This definition evolved to a more specific definition in another important book about Visual Analytics by Keim et al. [2010], updating and extending the fundamentals discussed by Thomas and Cook [2005]. Keim et al. [2010] propose the definition, that "Visual analytics combines automated analysis techniques with interactive visualizations for an effective understanding, reasoning, and decision making on the basis of very large and complex datasets". Although Visual Analytics is not easy to define, these two definitions are most commonly cited.

According to these definitions, the goals of Visual Analytics are stated by Keim et al. [2010] as the creation of tools and techniques to enable people to (see also [Thomas and Cook, 2005]) "synthesize information and derive insight from massive, dynamic, ambiguous, and often conflicting data", "detect the expected and discover the unexpected", "provide timely, defensible, and understandable assessments", and "communicate these assessment effectively for action". There are high-level Visual Analytics processes defined that try to achieve these goals, which we present in the next section.

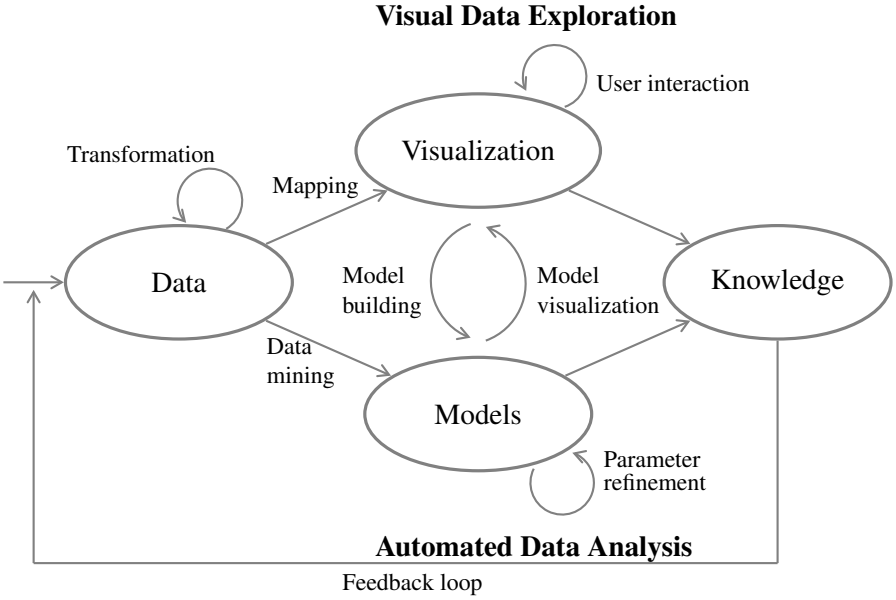

Figure 3.1: General Visual Analytics process, adapted from Keim et al. [2010]

3.1 Visual Analytics Process

In order to achieve the goals of Visual Analytics, a high-level Visual Analytics process must be defined. An abstract overview of such a process as introduced by Keim et al. [2008, 2010] is presented in Figure 3.1. For a more formal definition of this process we refer to Keim et al. [2008].

The first stage in the Visual Analytics process is the integration of usually heterogeneous data sources, which in most cases have to be pre-processed and transformed. After the data is prepared, the data is either mapped to a visual representation or automatic analysis methods are applied to generate models representing the original data. The key characteristic of Visual Analytics is the intertwinedness of human reasoning (through visualization), the automated methods (models), which are enabled in this process as visualized models and their visual evaluation and interactive modification. This interaction between models and visualizations strives for continuous refinement and verification of preliminary results. In this process we get insights through the visualizations and models, which generates knowledge [Keim et al., 2010].

The famous information seeking mantra "overview first, zoom/filter, details on demand" [Shneiderman, 1996] for visually exploring data is adapted and extended in the context of Visual Analytics to the Visual Analytics mantra by Keim et al. [2008]:

> "Analyze First - Show the Important - Zoom, Filter and Analyze Further - Details on Demand"

Lammarsch et al. [2011] propose to combine the human reasoning process and automated methods even further, by including specific characteristics of certain kinds of data. In their case

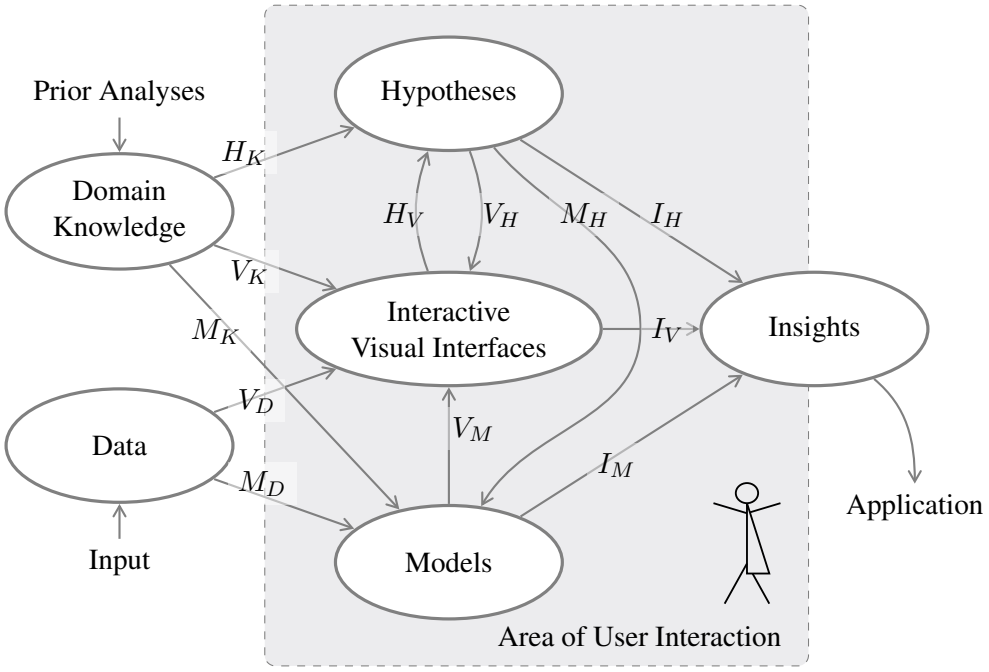

Figure 3.2: Visual Analytics process for time-oriented data, adapted from Lammarsch et al. [2011]

the structure of time in time-oriented data is integrated into their Visual Analytics process. The proposed Visual Analytics process is shown in Figure 3.2. The process description is based on the processes from Keim et al. [2008, 2010] and Bertini and Lalanne [2009]. The *Inputs* of the Visual Analytics process, are values as *Data* and *Hypotheses* or *Models* from *Prior Analyses* as *Domain Knowledge*. *Interactive visual interfaces* represent and transfer data to and from the user through visualization and interaction. In the Visual Analytics process by Lammarsch et al. [2011], the definition of *Hypotheses* is restricted to be a subclass of *Models*, so that only results validated on existing data are considered as *Models*. Hence not validated results are *Hypotheses*. Representations of a system of entities, phenomena, or processes are *Models*. A suggested explanation for an observable problem, or a reasoned proposal predicting a possible causal correlation among multiple phenomena are *Hypotheses*. Understandings gained by users are *Insights*. Users are guided to further actions or analysis of certain *Data*, *Hypotheses*, or *Models* by their *Insights*.

The main contribution of Lammarsch et al. [2011] is to handle time-oriented data according to the structure of time and to the Visual Analytics process model. The structure of time, as used in their process is presented in the next section.

3.2 Time-Oriented Data

According to Aigner et al. [2011], it is necessary for the visualization of time-oriented data to take its special properties and characteristics into account. In their work, Aigner et al. [2011] present and discuss in detail the definitions and characteristics of time and time-oriented data in relation to visualization. In this section we provide an overview of the definitions and characteristics we deemed important for the selection of the visualization methods in the following sections.

The definition mostly used for defining time-oriented data is the one by Müller and Schumann [2003]:

Definition 3.1. ***Time dependent data**, d, are data elements described as a function of time in the form*

$$d = f(t). \tag{3.1}$$

The relationship between data and discrete time stamps t_i is defined as

$$D = \{(t_1, d_1), (t_2, d_2), \cdots, (t_n, d_n)\}, \tag{3.2}$$

where

$$d_i = f(t_i). \tag{3.3}$$

Another definition cited as a similar definition by Aigner [2006] and Lammarsch [2010] is the one by Weber et al. [2001]:

Definition 3.2. ***Time series data**, D, are data elements described as a function of time in the form*

$$D = \{(t_1, y_1), (t_2, y_2), \cdots, (t_n, y_n)\}, \tag{3.4}$$

where

$$y_i = f(t_i). \tag{3.5}$$

As stated by Aigner [2006] and Lammarsch [2010], this is the same definition, if $y_i = d_i$. This declares the equivalence of the terms *time dependent data* and *time series data*, however Lammarsch [2010] states that the term time series data is misleading, because it implies nothing more than the consecutive order of the elements, and Aigner [2006] states the opinion that "time series is only a part of the broader field of time-oriented information". As our target problem stated briefly in Chapter 1 is in the domain of time series analysis as described in Chapter 2, that this definition is adequate for our case. For integrity we add the broader definition by Aigner [2006], that solves the problem that no concurrent data elements can exist:

Definition 3.3. ***Time-oriented information** is*

> *"Information, where changes over time or temporal aspects play a central role or are of interest."*

Design Aspects of Time

Before we visualize time-oriented data, the physical dimension of time has to be transferred to a model of time that reflects the phenomena of the real world and supports the considered analysis task in an information system. To transfer time into an adequate model, there are several design aspects to be considered. These characteristics of different types of time are categorized by Aigner et al. [2011] into the following aspects:

Scale: ordinal vs. discrete vs. continuous From a scale perspective, time could be in an ordinal time domain, where only the relative order is given, as before or after. In the discrete time domain the time is mapped to integers, by for example giving milliseconds since a specific point in time occurred. The continuous time domain is the mapping of time to real numbers, where for each two points in time, another point exists in between.

Scope: point-based vs. interval-based A data element is referring to a specific point in time, or to an interval. For example, the time value "01.03.2012" is referring to the time point "01.03.2012 00:00:00" or to the interval "[01.03.2012 00:00:00, 01.03.2012 23:59:59]".

Arrangement: linear vs. cyclic General perception dictates that time is a linear process starting in the past and running into the future. However, time can also be considered as reoccurring time values, and can therefore be seen as cyclic. An example for cyclically reoccurring time values, are the seasons of the year (spring – summer – fall – winter).

Viewpoint: ordered vs. branching vs. multiple perspectives The time is viewed as an ordered time domain, where one thing happens after another. Branching is a time domain, where different branches are possible alternative ways, but only one of them can actually occur. The multiple perspectives time domain extends the branching time domain in a way that allows multiple branches to occur at the same time.

According to Aigner et al. [2011], the hierarchical organization and definition of concrete time elements needed to relate data to time are:

Granularity and calendars: none vs. single vs. multiple Granularities are abstractions that support users to grasp the hierarchical structure of time, and make it easier to deal with time in every-day life. If only abstract ticks are modeled as values, there is no granularity used. Single granularities on the other hand are more concrete values measured in milliseconds or days. If multiple layers of granularities and whole calendar systems are used, multiple granularities are supported, such as weeks that consist of days.

Time primitives: instant vs. interval vs. span Time primitives are a basic set of elements, used to relate data to time. The primitives are categorized as anchored (absolute) primitives, called instant and interval, and unanchored (relative) primitives, called span.

Determinacy: determinate vs. indeterminate If any uncertainties are introduced into time-oriented data, the specification is indeterminate. Uncertainties are either incomplete or inexact information about time specifications, or errors introduced by converting between granularities. If a full knowledge about all temporal aspects is present, the specification is determinate.

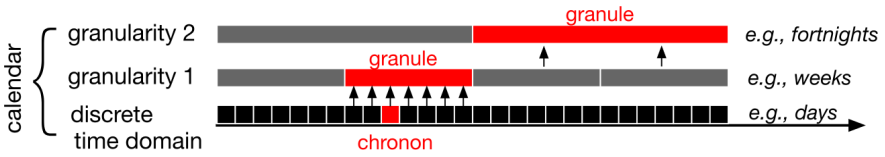

Figure 3.3: Example of granularities in a discrete time domain by Aigner et al. [2011]

Granularities

The formalizations and time granularity concepts are defined by Bettini et al. [2000]:

Definition 3.4. *A **granularity**, G, is a mapping G of the integers (the* index set*) to subsets of the time domain such that: (1) if $i < j$ and $G(i)$ and $G(j)$ are nonempty, then each element of $G(i)$ is less than all elements of $G(j)$, and (2) if $i < k < j$ and $G(i)$ and $G(j)$ are nonempty, then $G(k)$ is nonempty.*

Aigner et al. [2011] describe the formal definition of granularities as mappings of time values to larger or smaller conceptual units. An example of time granularities is shown in Figure 3.3. The elements in a granularity are called *granules*, except for the elements of the lowest granularities, which are called *chronons* for the smallest unit in the time domain. In Bettini et al. [2000] the relationships, conversions, and systems for granularities, as well as algebraic operations are defined. So far the main concept and basics about granularities are adequate for our purpose [Lammarsch, 2010].

Data Characterization

The design aspects of time discussed in the previous section are for modeling of the time domain. The next step according to Aigner et al. [2011] that needs to be characterized, is the time-oriented data. The data that is connected or associated to time primitives is characterized according to certain fundamental design alternatives. This characteristics of data components needs to be considered to design appropriate visual representations.

Scale: quantitative vs. qualitative Discrete or continuous ranges allow numeric comparisons and, are therefore quantitative variables. Qualitative variables are data values that are derived from nominal or ordinal data sets.

Frame of reference: abstract vs. spatial If the data includes a *where* aspect, meaning that the underlying data model describes for example a geographic position, the data is classified as spatial data. Other data is classified as abstract data.

Kind of data: events vs. states Event data are markers of a state change, whereas states are phases of continuity between events.

Number of variables: univariate vs. multivariate Univariate data sets have only one time-dependent variable, which means that for one time primitive only a single data value exists. Multivariate data sets have multiple time-dependent variables, which means that for one time primitive multiple data values exist.

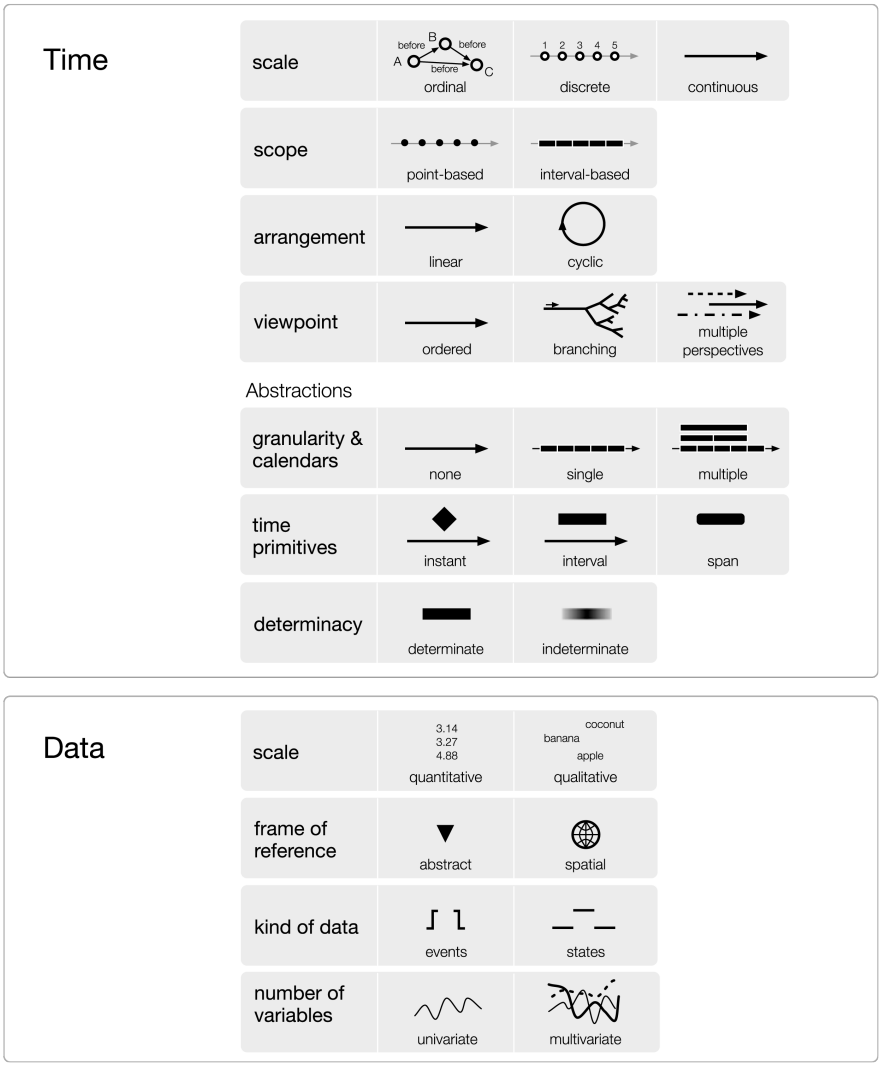

Figure 3.4: Design aspects of time-oriented data by Aigner et al. [2011]

Figure 3.4 shows a useful summary by Aigner et al. [2011] of the characteristics of modeling the time and time-oriented data.

3.3 Visualization of Time-Oriented Data

Because of the special characteristics of time and time-oriented data we discussed in the previous section, it is important to select appropriate visualization techniques. To allow the selection of visualization techniques suitable for our purpose in Section 3.4 we rely on the systematic view of visualizations for time-oriented data as described in the survey by Aigner et al. [2011]. Ac-

cording to Aigner et al. [2011], this systematic view is needed, because of the variety of generic and dedicated visualizations for time-oriented data and is structured by using three practical questions

(1) What is presented? - Time & data.

(2) Why is it presented? - To facilitate user tasks.

(3) How is it presented? - Through visual representation.

The first question deals with the details of the data under consideration. The answer to this question includes the characteristics of the data and the time domain, as presented in Section 3.2. The second question provides information on the problem domain, as well as the tasks and goals for the user. The answers to question (1) and (2) describe the reasons for the visualization. Question (3) is determined by the answers of question (1) and (2). This last question asks how we present the visualization in order to enable users of a specific domain to meet their tasks and goal based on the given time-oriented data [Aigner et al., 2011].

The aim of this section is to use this systematic view to understand the relevant visualization techniques for time-oriented data and distill the suitable techniques for the application and implementation of our prototype. We present our selection of suitable techniques in the following Section 3.4. To do so we first had to define the problem domain, the users, and the expected results of our thesis in regard to this systematic characterization. We also had to answer the above questions. We answered questions (1) and (2) in Chapter 2 where we investigated and specified the problem domain of time series analysis and partially answered question (3) where we explained how time series are plotted in this domain. The selection of the relevant visualization techniques based on the survey by Aigner et al. [2011] as described in Section 3.4 supports the answer to this question. For the definitive answer on how it is presented and how we enable the users to solve their tasks and meet their goals, we refer to the detailed discussion on the design of the prototype in Chapter 6 and the evaluation of the applicability of the prototype in Chapter 7.

3.4 Survey of Visualization Techniques

A comprehensive survey of existing visualization techniques dedicated to time and time-oriented data was done by Aigner et al. [2011]. Although the authors state that the survey is not exhaustive, they cover a wide spectrum of key techniques. In the thesis we focus on a subset of the techniques presented by Aigner et al. [2011] and refer to the original survey for more techniques. The basic visualization techniques relevant for visualizing time series data are presented in detail by Cleveland [1993]. Some basic visualizations from that publication are included in the survey, and most of the techniques investigated in the survey are extensions or advances of these basic visualizations.

The basic visualization techniques are needed when dealing with statistical time series analysis, because they are used in all of the mathematical and statistical software tools. Therefore we present these basic visualization techniques in this thesis, while we only present those advanced techniques gathered in the survey that could be useful for the design of the prototype. We

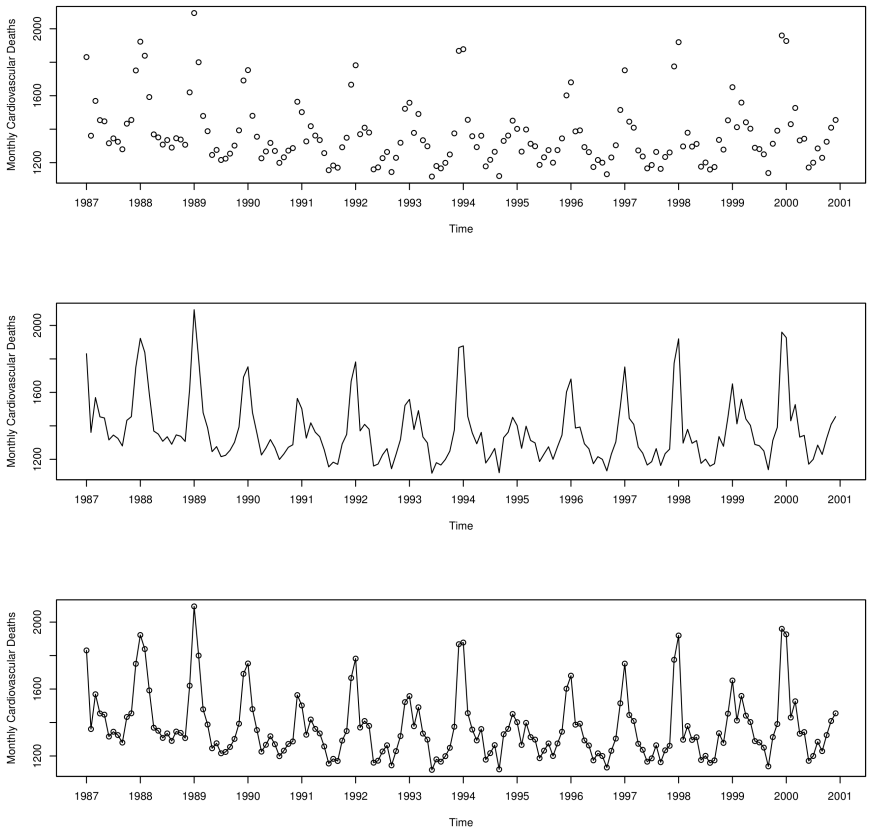

Figure 3.5: Point plot, line plot, and combination of both used in one plot, showing the example dataset from Section 7.1 generated with R.

are mainly interested in visualizations where the data has an abstract frame of reference and the variables are univariate. Visualizations designed for multivariate data are only considered if they could be reduced to visualize univariate data and are particularly useful. Visualizations suited for spatial data are not considered, because they are not relevant for our purpose of visualizing simple univariate time series. For the time arrangement visualizations with linear and cyclic arrangements are considered, while instant time primitives are more relevant than intervals.

Point Plot

The most basic way to display a time series is in a Cartesian coordinate system, using time as the horizontal axis and the data values as the vertical axis. This technique is called a point plot. An example for a point plot is shown in Figure 3.5 in the first graph. Other names for this technique are point graph, and scatter plot. Some of the more advanced techniques, are extensions of these basic visual representations.

Line Plot

An extension of the point plot is the line plot, where the points are located in the same way as in the point plot, but instead of displaying a point for each position, the positions are connected with a line. An example for the line plot is shown in Figure 3.5 in the second graph, a combination of a point and a line plot is shown in the third graph of this figure.

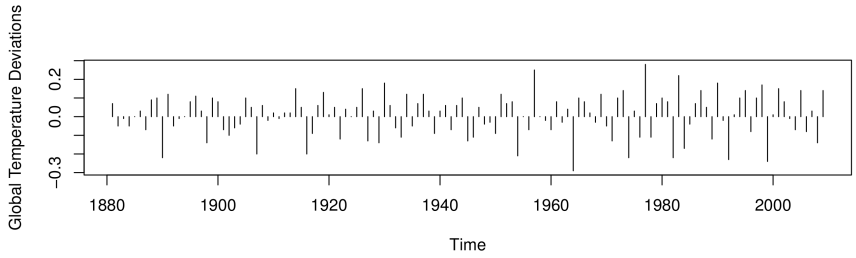

Figure 3.6: Bar graph of the global temperature deviations data from Shumway and Stoffer [2011] generated with R. In this figure the baseline is zero.

Bar Graph, Spike Graph

Bar graphs or spike graphs represent each value as the length of a bar or spike. They can only be applied if there is a natural baseline in the data. Spike graphs should be used very carefully, because peaks could appear to stand more out than troughs [Bisgaard and Kulahci, 2011]. Figure 3.6 is showing the bar graph of the global temperature deviations data.

Cycle Plot

The cycle plot, introduced by Cleveland [1993], is used to display the trend and the seasonal component of a time series. A season is subdivided at the data level, and for each cycle subseries the trend and the mean of the values are displayed. The subseries are positioned in the graph to show the seasonal cycle. A cycle plot is shown in Figure 3.7.

Tile Maps

Tile maps arrange data values based on temporal granularities in a matrix. The values are encoded by varying the shade of the tiles. Depending on the granularities, each cell (or tile) represents one granule, e.g., one cell a day, one column a week and one matrix a year. This representation can be interpreted very well, and it is possible to identify trends and weekly patterns. A tile map is shown in Figure 3.8.

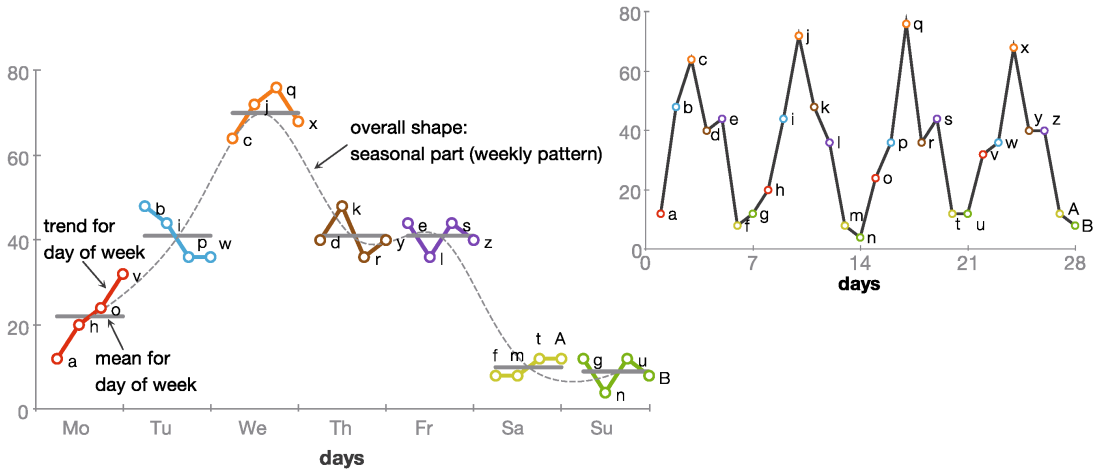

Figure 3.7: Cycle plot example adapted by Aigner et al. [2011] from Cleveland [1993].

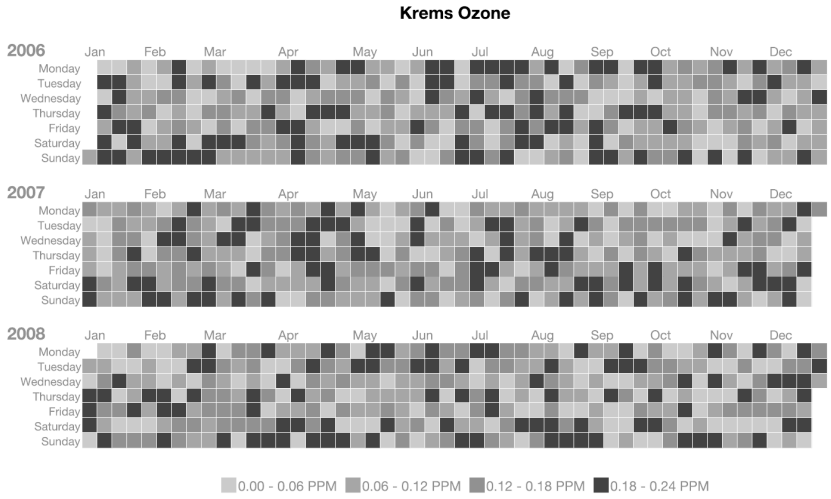

Figure 3.8: Tile map example adapted by Aigner et al. [2011] from Mintz et al. [1997].

Recursive Pattern

Recursive pattern visualization suggested by Keim et al. [1995] is a pixel based visualization that is particularly suited for large time-series because large amounts of data can be visualized using very little space. In this visualization technique pixels are arranged according to the inherent hierarchical structure of multiple granularities. This concept is shown in Figure 3.9 as an example from the GROOVE visualization.

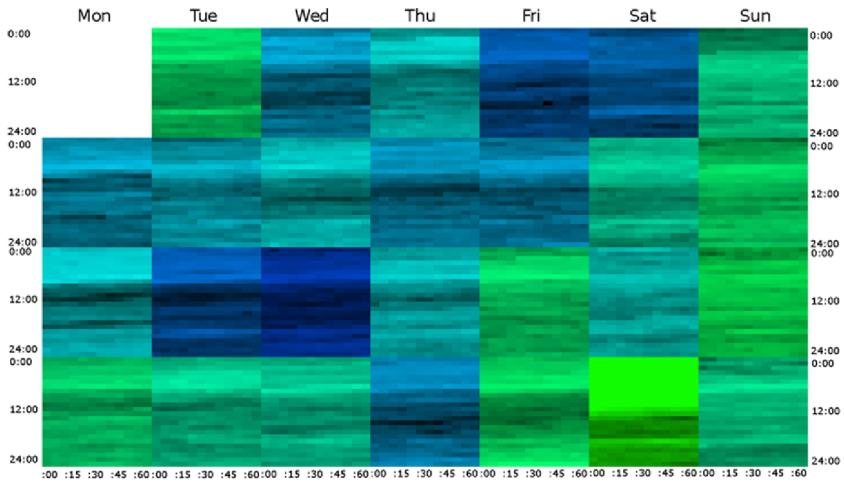

Figure 3.9: GROOVE example generated by Aigner et al. [2011] using GROOVE software from Lammarsch et al. [2009].

GROOVE

The granularity overview overlay visualization (GROOVE) introduced by Lammarsch et al. [2009], is based on the concept of recursive pattern mentioned before, and enables the user to configure a set of four time granularities to partition a dataset. The technique combines overview, by aggregated values, and details in one place using overlays. In Figure 3.9 an example of the GROOVE visualization is shown.

Enhanced Interactive Spiral

In the enhanced interactive spiral by Tominski and Schumann [2008], the time primitive is mapped to the spiral segments and the data values are displayed by using a two-tone coloring method. This method enables the overview and detail concept by design. In Figure 3.10 an enhanced interactive spiral visualization is shown. The following visualization, the spiral graph, is similar to this technique.

Spiral Graph

Spiral graphs focus on cyclic structures in data, such as seasonal trends. By enabling the user to interactively move the length of the spiral, it is possible to find the underlying seasonal trend in the data. This visualization is very similar to the one in Figure 3.10 showing the enhanced interactive spiral. The main difference is that the spiral graph is able to encode multivariate data.

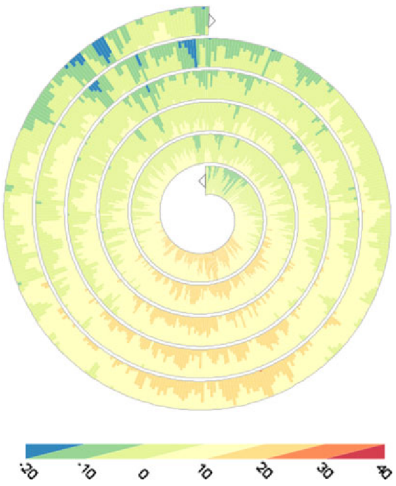

Figure 3.10: Enhanced Interactive Spiral example generated by Aigner et al. [2011] using the enhanced interactive spiral display tool by Tominski and Schumann [2008].

BinX

The BinX tool by Berry and Munzner [2004] enables the user to explore different aggregations of time series. The aggregations are applied on user selected bins, which are set using an interactive tool that offers users a quick way to try out differently sized bins. The aggregated information is visualized by basic line plots, box plots, and min-max bands or any combinations of them. An example is shown in Figure 3.11.

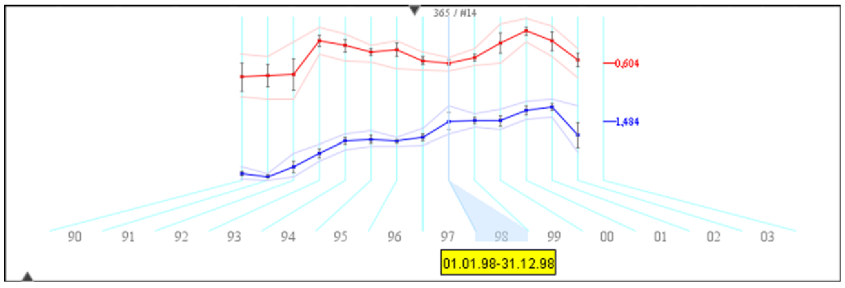

Figure 3.11: BinX example generated by Aigner et al. [2011] using the BinX tool by Berry and Munzner [2004].

Facet Zoom

Facet zoom by Dachselt et al. [2008] is a visual navigation aid to navigate hierarchically structured information spaces. For time-oriented data this tool is used to navigate the hierarchical structures of granularities. For navigation, the temporal granularities are displayed as a horizontal time axis with stacked bars. The facet zoom is shown in Figure 3.12 and was selected here because the concept of navigating is possibly very useful for our purpose.

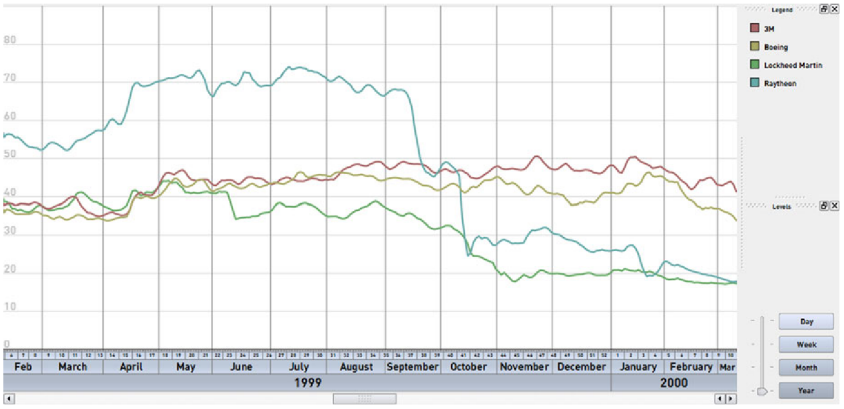

Figure 3.12: Facet Zoom example from Aigner et al. [2011] based on Dachselt et al. [2008].

Summary on Visualization Techniques

In the prototype we rely mostly on the basic techniques for visualization, as the representations for time series data are limited and dictated by the theory and visualizations used in time series analysis presented in Chapter 2. However, there are no such limitations for the realization of the interaction techniques. For the basic visualization of time series, and the time series representation of the residuals, we focus on point and line plots. The line in the plot is necessary to determine the progression of the time series and catch the seasonal patterns, which would be difficult with points alone as shown in Figure 3.5. Additionally a combination of a line and points allows a user to easily identify the separate data points. For that reason we use points and a line for the time series plot in the prototype.

The normal quantile-quantile plot is most informative if it is shown as point plot without a line between the points. A line is only used to show if the standardized residuals are normally distributed. For the ACF/PACF plot it is not recommended to use point or line plots, because with neither points nor lines is it easy to see the pattern in the lags. Because the ACF and PACF plots have a baseline at zero, the bar or spike graph is well suited for their visualization. The facet zoom is a very helpful tool for the interactive navigation of time series. We consider it as a way to enhance the time series plot in our prototype to navigate and explore through the time series and not lose the context of the time axis. The navigation of the hierarchical structure of granularities in Figure 3.12 provides the idea for the granularity slider in the prototype. The

aggregation of the granularities is inspired by the BinX tool and GROOVE. Although we do not consider the visualization of the bins itself and of GROOVE for our user interface at the moment, it is the underlying technique of aggregating the data by different sized bins and granularities that is interesting for our purpose.

The other visualization techniques (cycle plot, tile maps, recursive pattern, GROOVE, enhanced interactive spiral, and spiral graph) are very useful ways to display time series and enable us to recognize recursive or seasonal patterns. For the first version of the prototype we do not consider these visualizations for implementation. However they are strong techniques for larger datasets and we keep them in mind for further improvements of the prototype. The idea is to use these techniques as additional viewpoints on the time series to better recognize the seasonal behavior.

3.5 Related Tools for Time Series Analysis

Bernard et al. [2012] introduce a visual-interactive preprocessing system for time series data. It enables domain experts to interactively adjust the preprocessing pipeline, see visualizations of the intermediate steps and choose the right order and parametrization of these steps. Bernard et al. [2012] argue that in contrast to their solution, others are more of a 'black box' approach designed by computer scientists. In their case study they show the usefulness for domain experts. However, the authors concede that their system does not use a comprehensive toolkit of transformation methods known from literature.

TimeSearcher by Buono et al. [2005, 2007] is a visualization tool for time series data. The main objective is to search and explore large time series data. The more recent version 3 aims to provide forecasting through similarity-based forecasting. They use dynamic queries to specify constraints for the time series. The queries are applied to find patterns in historical time series and to display different possible forecasts. This queries specifying the pattern search can be adjusted by the user. They provide a user interface to display multiple forecasts.

3.6 Summary

In this chapter we introduced the research field of Visual Analytics and discussed the state of the art in that field. We presented a generic Visual Analytics process and a more specific process for time-oriented data. The definitions of time-oriented data and time-oriented information was presented and we answered the question how these aspects influence the design characteristics of time-oriented data. We discussed time granularities and their role for the structure of time. We explained how to visualize time-oriented data and provided a short survey of relevant visualization techniques. The study of these visualization techniques and related tools unveiled that Visual Analytics is a possible way to overcome the challenges we presented in Chapter 2 as our problem domain. Especially in the work of Bernard et al. [2012] we identified the fact that no comprehensive toolkit was used, as a motivation for our work. In contrast to their system, our solution is based on the R project for statistical computing. This enables us to use a broad range of different algorithms and methods implemented in R and the enormous amount of additional

packages. Furthermore the R project and the packages are considered free software, as they are published under the GNU general public license[1] and similar licenses.

[1] `http://www.r-project.org/Licenses` (08.01.2013)

CHAPTER 4

Problem Statement and Research Question

In Chapter 1 we briefly introduced the specific challenges of statistical time series analysis and in Chapter 2 we studied our target problem in more detail. In this chapter we provide a concise statement of the problems and derive the research question to solve them. We motivate the question and justify its importance by discussing and directly referencing to the previous chapters.

The first problem is that in many domains time series data naturally occurs unequally spaced and occasionally has missing values. Examples for this are, time series observations by a failing sensor, or measurements of irregular events [Box et al., 2008]. To analyze this sort of data, there are some specialized and advanced methods, e.g. by [Jones, 1985] and Box et al. [2008], dealing with missing values and unequally spaced time series [Bisgaard and Kulahci, 2011; Eckner, 2012]. The most common approach is to apply interpolation techniques to equalize the time series and to estimate the missing values and thereby enable the applicability of the various methods defined for equally spaced time series also for unequally spaced time series [Bisgaard and Kulahci, 2011]. We get a rich set of existing methods and implementations in statistical software tools to apply for time series analysis. However, these transformations are not flawless and it is important to carefully apply the techniques and the estimation of missing values, to avoid major drawbacks.

The second problem is to find the "best" model for a given time series. This model building process is known as Box-Jenkins methodology [Box and Jenkins, 1970], which we presented and discussed in Section 2.2. Although most of the methods in the single stages are implemented in statistical software tools, there is rarely interactive visual support for this process. The tools often support only the single stages of the process, but not the workflow of the whole process. We discussed the lack of that support in the software tools in more detail in Section 2.8.

We introduced to the new research field Visual Analytics in Chapter 3, that aims to "combine automated analysis techniques with interactive visualizations" [Keim et al., 2010]. Visual Analytics methods are considered to enable the user to effectively understand large and complex

datasets and to support reasoning and decision making on the basis of these datasets. The Visual Analytics process presented in Section 3.1 is a good starting point to formulate a Visual Analytics process for the target problem of time series analysis discussed before. The granularities discussed in Section 3.2 are a great chance to use the structure of time for aggregating time series with missing values and to use the granularities as lattice with different intervals to equalize time series that are not equally spaced. The discussion about the visualization of time-oriented data in Section 3.3 and the survey of the visualization techniques in Section 3.4 support the evidence that Visual Analytics can overcome the problems and challenges stated before. The discussed problems in the problem domain of time series analysis and the provided techniques of Visual Analytics as possible solutions, lead to the main research question:

- **How can Visual Analytics support the process of model building for time series analysis and help to choose the best transformation for unequally spaced time series and missing values?**

In order to answer this question and to tackle the problems, the goal is to design a Visual Analytics process and implement a prototype that supports the user

- in transforming unequally spaced time series to equally spaced time series and handling missing values, where the drawbacks of ordinary interpolation are negligible,
- and in the whole process of model building with interactive visualization methods.

To achieve the goals and to support the main research question, we formulate the hypotheses for this thesis, which are:

- **Visual Analytics enables the user to choose the best transformation for unequally spaced time series and missing values.**
- **Visual Analytics supports the model building process for time series analysis.**

Although Visual Analytics methods have not been applied to the specific target problem of time series analysis so far, there are solutions that are related in some way to ours. We discussed the related tools and related work in Section 2.8 and 3.5. We already presented the definitions of a high level Visual Analytics process and a more detailed process for time-oriented data in Section 3.1, but these processes do not answer the question of the target problem. Equally, our description of the target problem and the Box-Jenkins methodology in Section 2.2 guide to ask the question, but do not answer it. The methodology and the tools presented and discussed in Section 2.8 provide ideas on how visualizations are supporting some of the single steps and how they are partially implemented and used in different tools. These ideas again motivate to ask the question and confirm that it is worthwhile to answering it.

Another motivation to answer the question is that time series analysis is used in a broad range of different fields by domain experts with different skill levels. We refer to important example applications mentioned in textbooks and research in the corresponding fields of scientific journals. The *Journal of Time Series Analysis* [1], a leading journal in its field, refers to fields of

[1] `http://onlinelibrary.wiley.com/doi/10.1111/jtsa.2012.33.issue-5/issuetoc` (08.01.2013)

application in neurophysiology, studies of biological data and signal processing, which itself is applied for medical applications. A recent (September 2012) special issue of the journal is about Time Series in the Biological Sciences, which shows the importance of time series analysis in this field.

In textbooks about time series analysis, for example by Shumway and Stoffer [2011], the authors list examples for the impact of time series analysis on scientific applications. In epidemiology it is applied to research and predicts the number of influenza cases. In medicine it helps evaluate drugs used for treating hypertension by analyzing blood pressure measurements taken over a period of time. Furthermore, brain-wave time series patterns from functional magnetic resonance can be used to study how the brain reacts to certain stimuli under various experimental conditions.

It is beneficial to think about Visual Analytics methods as a tool to support these users in their tasks. It is evident that human perception and cognition could be advantageous in some parts of the model selection process. Although the calculations of the models are better performed by a computer, it is difficult for a computer to evaluate which of a set of "good" models is the "best". By providing visualizations of these models, it is easy for humans to evaluate them. By supporting this process and assisting in handling missing values and unequally spaced time series, a domain expert, like a biologist, chemist, or epidemiologist, is able to find the "best" models for the time series he or she is working on.

CHAPTER 5

Scientific Approach

We follow roughly the characteristics of design studies described by Munzner [2008]. Hence we need a comprehensive understanding of the problem domain and the target problem to achieve the goals and verify the hypotheses defined in Chapter 4. In our case the domain is time series analysis and the target problem is discussed in Chapter 4. The first step in the previous chapter was to describe the problem precisely and formulate the main research question and the hypotheses for the thesis.

To enable the reader to judge our solution, we explained the background information about time series analysis in Chapter 2. To justify the application of a Visual Analytics process to the target problem, and argue the design choices, we presented the basics of Visual Analytics and time-oriented data in Chapter 3. These two chapters provide the results of studying the state of the art in each of the two areas of research. This was the second step after specifying the problem statement and stating the research question.

We combined these findings to define a Visual Analytics process as described in the following Chapter 6. After defining the Visual Analytics process we formulated the requirements for the prototype implementation. Before we started to implement the prototype, we decided on the technologies that we then used for the implementation. The design of the process and the prototype, as well as the implementation of the prototype were done iteratively using agile methods.

This thesis was carried out as part of the HypoVis[1] project. The intermediate steps of the state of the art research, the design choices and the implementation of the Visual Analytics process and the prototype were shown in the project meetings of this project. In these regular meetings we presented the findings and solutions, which were reviewed and discussed with the project team members. The resulting suggestions to improve the solutions were considered in the next iterations of the design and implementation process. Besides the continuous evaluation of the process definition and prototype implementation in the project team, we tested and evaluated the prototype using an example dataset and tailored use case scenarios. The critique,

[1] `http://www.ifs.tuwien.ac.at/~lammarsch/HypoVis` (08.12.2013)

suggestions and findings unveiled by this evaluation and testing are considered in the discussion and conclusion of this work. Any open points are considered for future work.

We stated that we partially applied agile methods and processes for the design and the implementation of the Visual Analytics process and the prototype. We followed the practices and agile philosophy of Shore and Warden [2008]. We created an initial Visual Analytics process definition and an initial prototype based on basic requirements that we formulated as epics and user stories. Based on the initial definition and implementation, we gradually evolved the Visual Analytics process and the prototype. We extended the process definition and added more features to the prototype. This way the design of the Visual Analytics process and the prototype iteratively grew to the final definition, design and implementation.

User Stories and Epics

User stories are a way to describe the requirements for a software product, which in our case is the Visual Analytics process and the prototype. User stories evolved from the XP (extreme programming) software development methodology introduced by Beck [2000] and have an important role in other lean and agile software development methodologies. One popular methodology is Scrum [Cohn, 2010], where user stories are used to build the product backlog and for sprint planning. The application of user stories in Scrum is discussed in detail by Cohn [2004, 2010].

User stories help to formulate the requirements in a way that is easy to use in discussions within the development team, with customers or other stakeholders. It is possible to use any template formulation that is suitable for a specific project. We used the simple template proposed by Cohn [2010], which is

As a `<type of user>`, I want `<some goal>` so that `<some reason>`.

User stories are a simple tool that make communication easier. They are not tied to a specific medium and in most cases simple index cards are used to write them down.

Epics are a form of user stories. Usually in the beginning only high level goals and requirements are known. These goals and requirements formulated as user stories are called epics. Through the process of refinement the epics are broken down to smaller epics and further on until they are very specific user stories. For these user stories it is then possible to estimate the amount of work and implement the user story. It is usually not possible to draw an exact line between epics and user stories. It depends on the project and the context.

CHAPTER 6

Design of the Visual Analytics Process

In Chapter 2 we discovered the details of the problem domain. We identified Visual Analytics methods in Chapter 3 as a basis to define a Visual Analytics process to overcome the stated problems and answer the research question summarized in Chapter 4. In this chapter we rely on our findings to present the main contributions of our work and the results of this thesis. To do so, we provide the definition of a tailored Visual Analytics process in Section 6.1 that is used for the implementation of the prototype. In Section 6.2 we formulate the requirements for the prototype as epics and user stories. We introduce the technologies we used for the implementation of the prototype in Section 6.3. In Section 6.4 we provide the final design and the description of the prototype, and in Section 6.5 we discuss the packages and methods of the R project that we used for computations regarding the prototype.

6.1 Visual Analytics Process Definition

In Section 3.1 we introduced two state of the art Visual Analytics processes. We presented the general Visual Analytics process (K) by Keim et al. [2010] in Figure 3.1 and the more specific process definition using the structure of time (L) by Lammarsch et al. [2011] in Figure 3.2. We considered these two processes as the basis for the definition of our tailored process (C) that we implemented in the prototype. To derive our tailored process (C), we adjusted processes (K) and (L) to fit the specific domain problem of time series analysis. This resulted in two new process definitions, one for the process of time series modification (M) in regard to missing values and unequally spaced time series and the other for the problem of model selection (S). We then combined processes (M) and (S) to a high level iterative process (C), that enabled us to use the results of (S) as input for (M), which creates a feedback loop to adjust the time series to the new insights of (S).

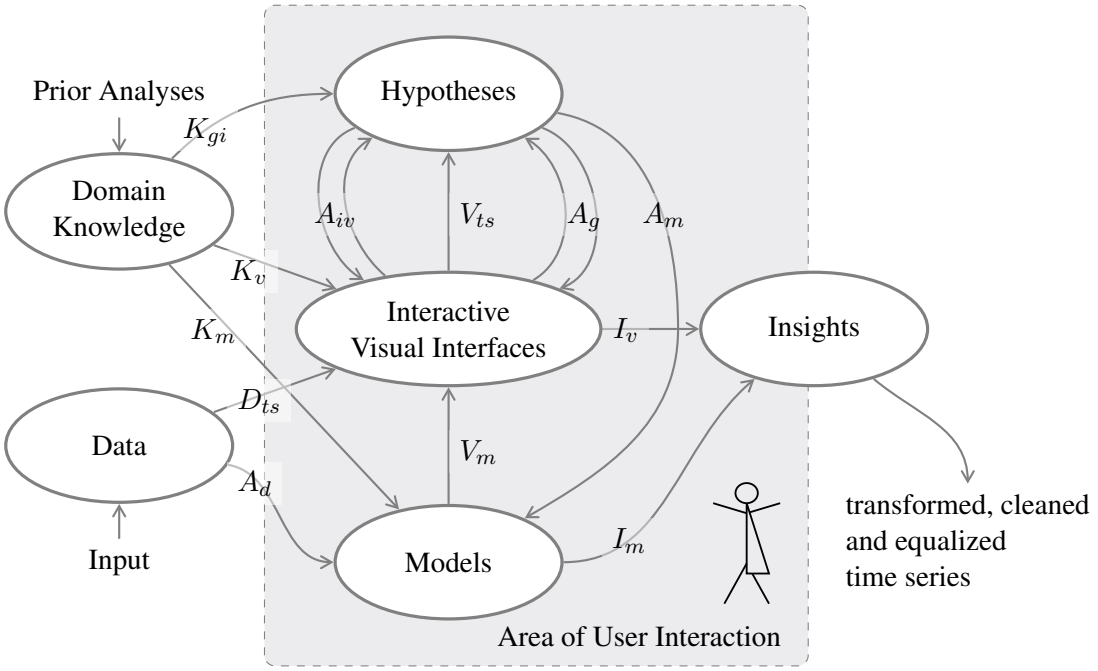

Figure 6.1: Visual Analytics Process for Time Series Manipulation (M). The figure displays the process of data manipulation and data transformation to granularity levels, to handle missing values.

Visual Analytics Process for Time Series Manipulation

In Figure 6.1 we show the design of the process for modifying the time series. The goal of this process is to aggregate the time series to a level of granularity and/or to impute missing values and/or transform unequally spaced time series to equally spaced time series. The *Data*, which is the time series to analyze, is provided as an *Input*. *Domain Knowledge* is based on experience and *Prior Analyses*. The *Data* is visualized by an *Interactive Visual Interface* (D_{ts}). The *Domain Knowledge* about granularities, the structure of time, and missing values is used (K_v) to view visualizations and build *Hypotheses* (V_{ts}) by interpreting the visualizations. The *Domain Knowledge* about granularities and imputation (K_{gi}) is used to refine the *Hypotheses* and the *Interactive Visual Interface* by adjusting the imputed values (A_{iv}) and the level of granularity (A_g). The granularity mapping and missing value transformations are applied (A_m) to build *Models* from *Hypotheses* using the given time series *Data* (A_d). The transformed and/or granularity mappings of the time series are then visualized again in the *Interactive Visual Interfaces* (V_m). *Insights* are gained from the *Interactive Visual Interfaces* (I_v) and/or from the *Models* (I_m). The result is a time series that is equally spaced and does not contain any missing values (*transformed, cleaned and equalized time series*). The *Area of User Interaction* is highlighted in gray and indicates the process steps, where the user is part of the process through user interaction.

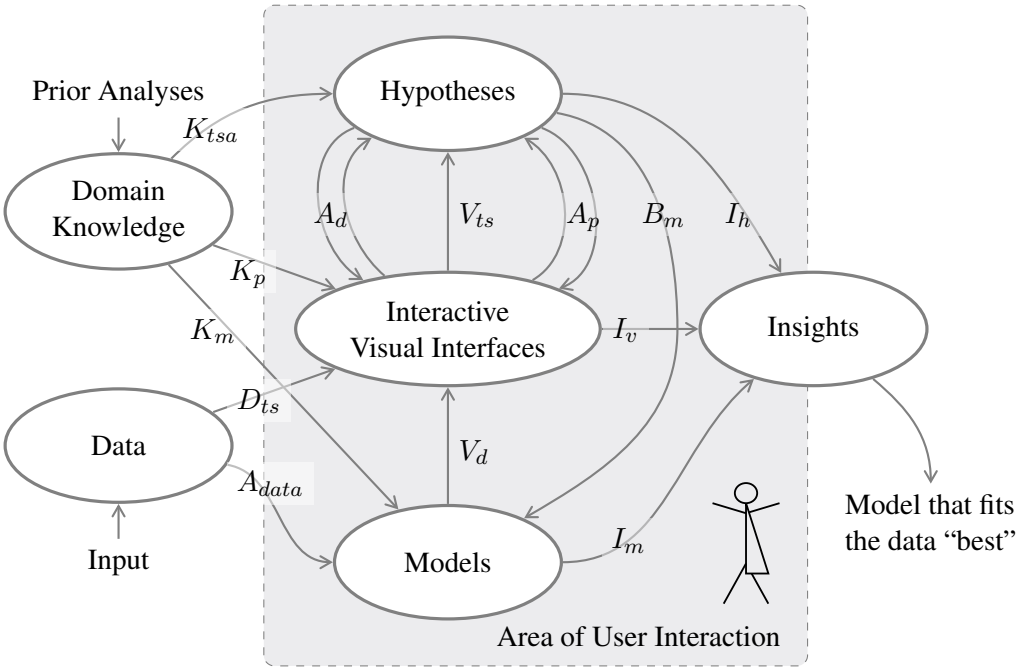

Figure 6.2: Visual Analytics Process for Model Selection (S). The figure shows the Visual Analytics process for selecting and adjusting the model iteratively, to find the "best" fitted model.

Visual Analytics Process for Model Selection

The basis of this model selection process is the Box-Jenkins methodology presented and discussed in Section 2.2 of Chapter 2. The methodology is shown in Figures 2.1, 2.2, and 2.3 of the section. We show the Visual Analytics process definition in Figure 6.2. The goal of the process is to find a model and adjust the order of this model and the level of differencing to estimate a model that is the "best" fit for the given time series. The details of the theoretical underpinnings of this process in statistical time series analysis are discussed in Chapter 2. Like in Figure 6.1, the time series in Figure 6.2 is *Data* provided as *Input*. The *Domain Knowledge* is based on experience and *Prior Analyses*. The *Interactive Visual Interface* is used to visualize the *Data* (D_{ts}) to decide on the class of models and adjust the number of parameters as well as the level of differencing. To interpret the *Interactive Visual Interfaces*, the *Domain Knowledge* about time series analysis (K_{tsa}) and about visualizations of time series and time series models (K_p) is used. Based on this knowledge and the visual representations of the time series and time series model, the *Hypotheses* are formed (V_{ts}). By adjusting the level of differencing (A_d) and the order of the model (A_p), the *Hypotheses* are refined. Based on the *Hypotheses* the model is estimated with the given parameters (B_m) to build a model based on the *Data* (A_{data}). The resulting model is analyzed using the *Domain Knowledge* about time series models and model diagnostics (K_m) and the visualizations of the residuals and model parameters (V_d). In this iterative refinement of the process, *Insights* are gained by (1) interpreting the *Interactive Visual*

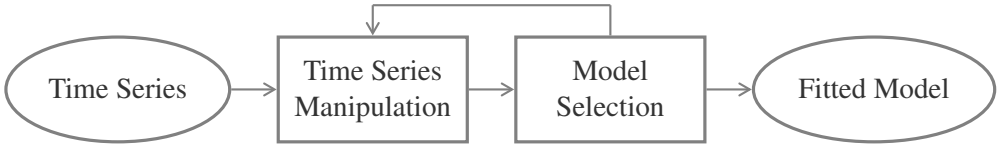

Figure 6.3: Combined Visual Analytics Process (C). This high level process shows how the previously presented processes are combined and how they interact.

Interfaces (I_v) deciding the fitness of the underlying model that is visualized, (2) the parameter estimations which lead to the model configuration (I_m), and (3) the refinement process of the *Hypotheses* building (I_h). The result is a model configuration with estimated parameters, that is the "best" fit for the given time series, and can be used for forecasting. The *Area of User Interaction* is again highlighted in gray.

Combined Visual Analytics Process

The Visual Analytics processes for time series manipulation (M) and model selection (S) are combined to a higher level Visual Analytics process, where the result of the time series manipulation, the adjusted and imputed time series, is used as the input data for the Visual Analytics process for model selection (M). The insights and models derived from the Visual Analytics process for model selection (S) then provide new knowledge for the time series manipulation. The final model configuration of process (S) could be used to get a better estimation of the imputed missing values in process (M). This iterative combination of the two Visual Analytics processes enables us to refine the model selection further and results in a better model for the given time series. In Figure 6.3 the iterative combination of the previously presented Visual Analytics processes is shown.

6.2 Prototype Requirements

The requirements are driven by the main research question and the goal of the thesis we stated in Chapter 4. The goal we defined to overcome the discussed problems, is to design a Visual Analytics process and implement a prototype that supports the user

- in transforming unequally spaced to equally spaced time series and to handle missing values, where the drawbacks of ordinary interpolation are negligible, and
- in the whole process of model building with interactive visualization methods.

As we already stated in Chapter 5 about the scientific approach, we applied some basic techniques from agile software development methodologies to implement the prototype. The research question and the goal of the Visual Analytics process and the prototype, indicate how we derived the following final epics and user stories in an iterative process. We then used these epics and user stories to formulate the Visual Analytics process and implement the prototype.

Epics

As mentioned in Chapter 5 epics are large user stories. They do have the same structure, but tell a story about the intended use case from a higher perspective [Cohn, 2010]. We split the epics into smaller user stories to refine the requirements. The idea in this paragraph is to provide the most important requirements for the Visual Analytics process and the prototype without going into the full detail of the low level user stories.

Epics that formulate the high level goals:

- As a domain expert (user), I want to handle missing values in a way so that drawbacks are minimized that we sometimes have for ordinary interpolation.
- ..., I want to transform unequally spaced to equally spaced time series so that I can apply less complex time series analysis methods.
- ..., I want to build a statistical time series model so that I can use that model for different purposes, e.g. forecasting.

User Stories

User stories are defined to get a more detailed understanding of the requirements [Cohn, 2010].

- As a domain expert (user), I want to immediately notice gaps and missing values in the time series so that I can decide on how to handle them.
- ..., I want to impute missing values with a method of my choice so that I have no missing values any more.
- ..., I want to adjust the values for the missing values that were imputed by the method of my choice so that I can fit the values to any pattern that is perceived from the visual representation of the time series.
- ..., I want to adjust the level of granularity so that I can decide on the level of detail in the time series.
- ..., I want to adjust the level of granularity so that I can overcome missing values by aggregating to higher levels.
- ..., I want to adjust the level of granularity so that I can use this granularity as a lattice to transform unequally to equally spaced time series.
- ..., I want to select a certain region in the time series so that I can use any subregion of the time series for the model selection step.
- ..., I want to see all important visualizations of the time series and the model so that I can decide on the model and assess how well the model fits the time series with one glimpse.
- ..., I want to adjust the model parameter at the place where the visualization provides the information about this model parameter so that I can intuitively find the “best” model.
- ..., I want to include and exclude the seasonal components of the model and the seasonal parameter inputs so that I can compare the seasonal influence and if no seasonal components are needed, they do not distract me.
- ..., I want to see how a new model configuration compares to the previous model configuration so that I can decide if one model configuration is better than the other.

6.3 Implementation Technologies

In Section 6.1 we introduced the Visual Analytics process for time series analysis and derived the requirements for the prototype in Section 6.2. Before we provide the details about the prototype, we present the technologies used in the implementation.

Java

The prototype is implemented in the object oriented high-level programming language Java[1]. Java is widely used, especially at universities, because it is platform independent and available as free and open source software. It is also well documented and there is a wide range of libraries with special functionality already implemented. The main reasons to implement the prototype in Java are the available libraries to use R in Java programs and the *prefuse* framework that supports us in creating interactive visualizations. Furthermore there is the TimeBench API, which provides an implementation for time-oriented data.

Prefuse

Heer et al. [2005] introduced the software framework prefuse for Java that supports programmers in creating dynamic visualizations. Prefuse consists of components to create and customize interactive visualizations. The visualizations and interaction techniques are based on the findings in the Information Visualization community. It is very easy for programmers to customize and extend the existing components and string together the visualizations, as well as modify and extend them.

JRI - Java/R Interface

We used the R project for statistical computing [R Development Core Team, 2012] in the prototype. The details about the R project, specific procedures and packages used in the prototype are discussed in Section 6.5. Using Java/R Interface (JRI) enables us to use R in combination with Java. It is possible to run R commands and output the results from inside a Java application. JRI was first developed as a standalone package,[2] but was later included in the rJava package [Urbanek, 2011].

TimeBench

TimeBench is a Java API for time-oriented data, which is developed as part of the HypoVis project[3] and CVAST[4]. It provides basic objects and elements as well as calendar operations for time-oriented data. The most important data structure we used in our prototype is the temporal dataset class. The temporal dataset was used to store the time series and provide it to the visualization methods. The calendar operations support the granularity mappings of temporal datasets.

[1] `http://java.sun.com` (09.01.2013)
[2] `http://www.rforge.net/JRI` (13.01.2013)
[3] `http://www.ifs.tuwien.ac.at/~lammarsch/HypoVis` (07.01.2013)
[4] `http://www.cvast.tuwien.ac.at/cvast` (16.01.2013)

Figure 6.4: VisuTimAlytics Overview. The figure is showing the complete user interface, where (1) is the time series display, (2) the model configuration area, (3) the ACF/PACF plot as well as further model configurations, and (4) the residual analysis plots.

In our prototype we used the granularity actions to build an aggregation tree, which contains the different granularity hierarchy levels for the time series.

6.4 Prototype Design

Based on the process definition in Section 6.1 and the epics and user stories in Section 6.2, we designed the prototype using the technologies discussed in Section 6.3. As it was a progress of refinement along to the formulation of the user stories, we present the final version of the design.

Before we discuss the design choices and implementation ideas in detail, we first highlight the most important design decisions for the prototype. One important decision was to use the R project for statistical computing [R Development Core Team, 2012] as a comprehensive toolkit for time series analysis and other calculation tasks. R provides a broad variety of methods known from literature and our prototype is designed in a way that allows us to choose any suitable implemented method from any package existing in R for any step in the process. The current implementation only uses one specific set of methods, but the user is free extend the interface to use any other method.

Because the calculations in time series analysis can be very time-consuming, especially with large datasets, it is important that the user interface is still responsive to user input, while

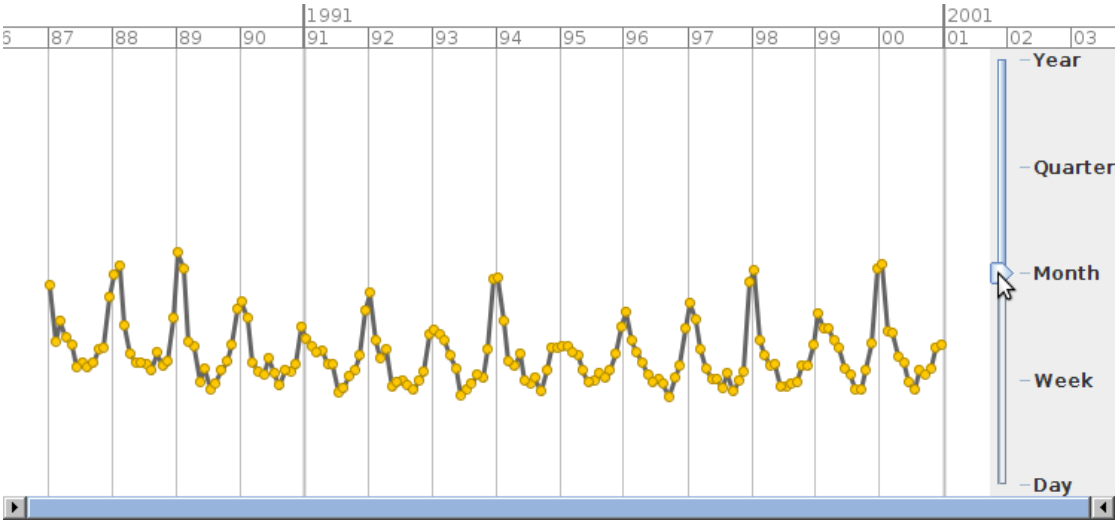

Figure 6.5: VisuTimAlytics Time Series Plot. The figure shows the detailed view of the time series plotted over time. The time axis is shown on the top and is adjusted when using the range slider on the bottom. On the right side there is the slider for adjusting the granularity level.

calculations are carried out. This is achieved by using Java threads, which allow the computer to pre-compute model configurations and provide them upon request. As a result, the user interface shows good reaction times for user input, even if the calculations are running in the background.

We named our prototype *VisuTimAlytics*, which is an abbreviation of Visual Analytics and time series analysis.

User Interface

The user interface of the prototype is based on the workflow of the Visual Analytics process we defined in Section 6.1 and the findings of the state of the art research in Chapters 2 and 3. The visualizations are inspired by the plots that are used when applying the Box-Jenkins methodology in R and by the findings when studying the related work and the survey of visualization techniques. We extended these visualizations so that the user is able to interactively adjust and configure the dataset and model configurations. The result is an early prototype that implements the Visual Analytics process for time series analysis.

The graphical user interface of the VisuTimAlytics prototype consists of a menu bar and four areas. An overview of the graphical user interface is shown in Figure 6.4. Using the menu bar the user can adjust some data related behavior. The main area of the application (1) displays the time series plot, where the granularity level can be adjusted and missing values are handled. Area (2) is the toolbox that can be used for configuring the model selection. The ACF/PACF plot is shown in area (3), where the number of model parameter can be adjusted directly within the plot. The plots in area (4) show the results of the parameter estimation as the plots for the residual analysis.

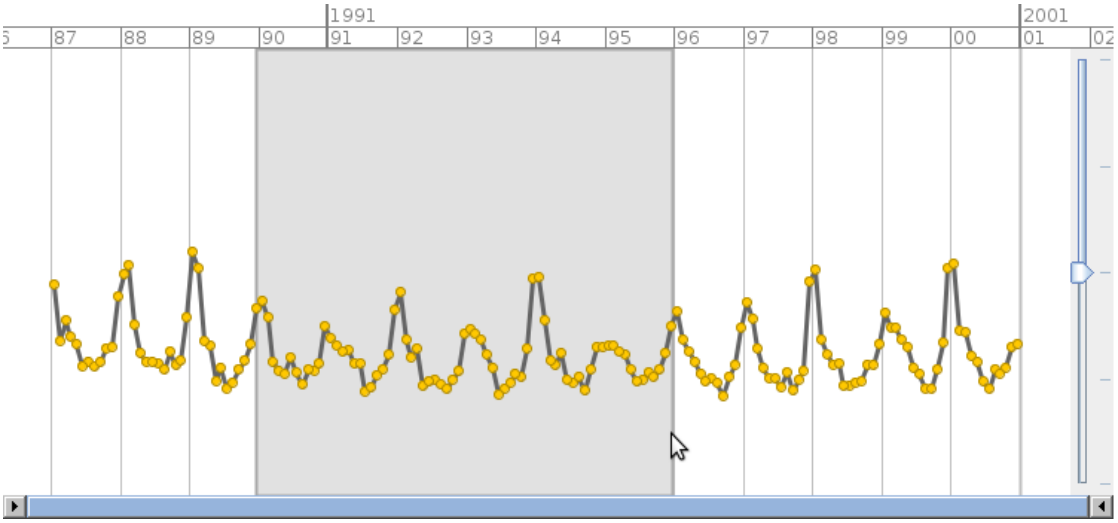

Figure 6.6: VisuTimAlytics Time Series Plot with Selected Region. The figure shows the time series plotted over time, where the user has selected a specific region that is used for the next steps in the time series analysis.

The time series plot (1) is shown in more detail in Figure 6.5. In this area of the graphical user interface it is possible to explore the time series. The horizontal range slider on the bottom allows the user to zoom in and navigate through the time series. When changing the zoom level on the range slider, the time axis is adjusted to show a suitable resolution of time. Details about the time points are provided on demand when moving the mouse cursor over a certain time point. The essential part in this figure is the granularity slider on the right side, which is discussed in more detail below. Figure 6.6 shows again the time series plot, this time with a specifically selected time interval that we will use for the next steps in the time series analysis. To create a region selection, the user has to press the control key and then click and drag the left mouse button. When the mouse button is released, the region is selected. By pressing the control key and clicking inside the region, it is possible to adjust the selection to the left and right. By pressing the control key and clicking at either side of the region it is possible to resize the region. To remove the region selection, the user has to double click the right mouse button on the region.

In Figure 6.7 the model selection toolbox (2) and the ACF/PACF plot (3) are shown in more detail. These are the areas for the configuration of the model. In the toolbox the *max lag* input changes the number of lags in the ACF/PACF plot below and in the ACF plot of the residuals in area (4). The *Include Seasonal Parameters* check box enables or disables the configuration of the seasonal component in the model, which also enables or disables the input for the *Seasonal Span*, as well as the *Seasonal Difference* slider. With the *Difference* slider and the *Seasonal Difference* slider the numbers for the parameter d and seasonal parameter D are selected. The continuous vertical lines in Figure 6.7 can be adjusted along the x-axis to select the order of the model, which is synonymous with the number of parameters. There is one vertical line for p, which is

Figure 6.7: VisuTimAlytics Model Configuration Toolbox and ACF/PACF Plot. The toolbox at the top is used to configure the maximum number of lags to display, to include the seasonal components in the model, adjust the seasonal length (span), and change the difference and seasonal difference of the time series. The four continuous vertical lines next to the cursor are for the configuration of the time series model. In this figure they are set to $p = 2$, $q = 1$, $P = 2$, and $Q = 0$, which is the final model configuration for this time series. The plot is the ACF and PACF over the lags and is interpreted according to the Tables 2.1 and 2.2 presented in Chapter 2.

the order and therefore the number of parameters for the autoregressive part of the model AR(p). There is another vertical line for q, which is the order and therefore the number of parameters of the moving average part of the model MA(q). If the seasonal components are enabled by the check box, two additional continuous vertical lines appear, one for P, which is the order and therefore the number of parameters of the autoregressive part of the seasonal component of the model AR$(P)_s$, and another one for Q, which is the order and therefore the number of parameters of the moving average part of the seasonal component of the model MA$(Q)_s$. The seasonal span s can be adjusted using the *Seasonal Span* spin box in the toolbox.

The residual analysis in the diagnostic plots (4) are shown in detail in Figure 6.12. In this

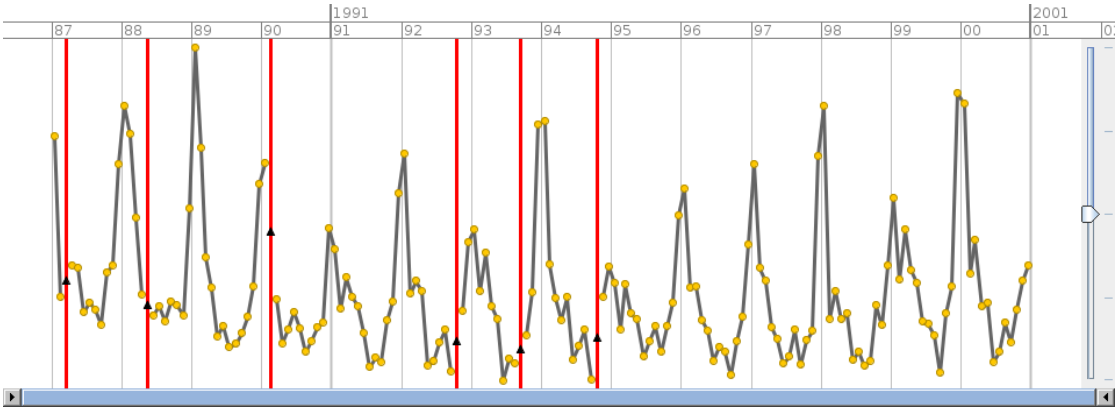

Figure 6.8: VisuTimAlytics Missing Values Highlighted. The horizontal red lines indicates the position of the missing values. The triangles are at the positions of the estimated values. It is possible to drag the triangles and adjust the estimated values to any patterns.

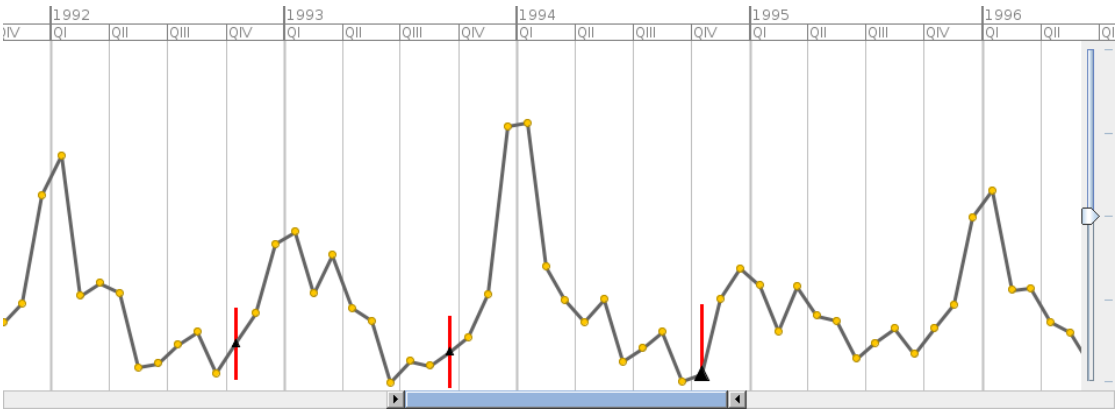

Figure 6.9: VisuTimAlytics Missing Values with Confidence Interval. The red lines highlighting the missing values are used to mark the upper and lower boundaries of confidence intervals, provided by the estimation of the missing values.

area the result of the model estimation are shown. The top plot shows the standardized residuals over time. In the middle there is the ACF of the residuals over the lags and the normal quantile-quantile plot of the standardized residuals. On the bottom plot the p-values of the Ljung-Box statistic over lags are shown. More details on the diagnostic plots in the prototype are provided below in another section.

After introducing the graphical user interface, we discuss the specific functions of the prototype in the following sections.

Missing Values

By default missing values are highlighted by a vertical line in the time series plot as shown in Figure 6.8. In the visualization representation of the time series it is possible to either leave a gap where the missing value occurs, or to connect the point before and after the gap, which simply ignores the missing values. It is also possible to calculate an estimated value for the missing value. The prototype implements the basic methods for missing value estimation used in R. These R methods are implemented using an interface, which enables us to extend the prototype and choose any missing values estimation method from R for application. After applying the method, it is possible for the user to adjust the estimated value by dragging the data point along the vertical axis, see Figure 6.9. If the estimation method provides any confidence interval, the adjustment of the data point is limited to within these borders. In this case the vertical line highlighting the missing values is reduced to the length of the interval and marks the lower and upper boundaries of the confidence interval. This is shown in Figure 6.9.

Granularity Levels

In the detailed view of the time series plot, shown in Figure 6.5, the granularity levels are displayed on the right side of the time series plot. By moving the slider vertically, the user can adjust the corresponding granularity level. The calculation of the new values for a higher or lower granularity level is implemented by providing an aggregation function interface for each granularity level. The default aggregation functions *sum* and *mean* are implemented in the current version of the prototype. Both aggregation functions are implemented in a Java method directly and by using the R interface to use these methods implemented in R. It is possible to implement any aggregation method in Java and use any aggregation method that is implemented in R. How to handle missing values in the aggregation is defined in the function definition and should be considered when implementing a custom aggregation function. It is possible to use different aggregation methods and handle missing values differently for each granularity level.

Time Series Manipulation

Below we explain how the Visual Analytics process for time series manipulation is implemented in the VisuTimAlytics prototype. For each transition in the process we provide the corresponding labels from Figure 6.1 in parentheses. Where the transition affects the user interface, we refer additionally to the corresponding Figures 6.5, 6.8, and/or 6.9.

Viewing the data in the time series plot of the user interface (Figures 6.8, 6.9; D_{ts}, V_{ts}) unveils the missing values. One possibility to deal with the missing values is to aggregate the time series to a higher granularity level (Figure 6.5; A_g). Adjusting the granularity level triggers the granularity mapping of the time series (A_m, A_d) and results in a new time series that is provided to the user interface (Figure 6.5; V_m), which is again viewed in the time series plot (V_{ts}). Another way is to impute the missing values and display the resulting time series (Figures 6.8, 6.9; A_m, A_d, V_m, and then V_{ts}). It is possible to manually adjust these imputed values (Figure 6.9; A_{iv}). The insights (I_v, I_m) and the domain knowledge (K_{gi}, K_v, K_m) are part of the user interaction, but not part of the user interface.

Figure 6.10: VisuTimAlytics Progress of Difference Slider. The user interface shows the change to the visualizations when adjusting the level of difference. This enables the user to evaluate if a higher level of difference is better than a lower one. This is not the case in this figure, because the time series plot and the ACF/PACF plot indicate a stationary behavior for the trend.

Model Selection

In this paragraph, we describe how the model selection process defined in Section 6.1 is implemented in the VisuTimAlytics prototype. We explain in detail how the user interface facilitates this process and creates short feedback cycles for the task of model selection. For each transition in the process that we describe, we provide the corresponding labels from Figure 6.2, the number of origin from the original Box-Jenkins methodology in Figure 2.1, and the number of the affected area in the user interface in Figure 6.4. By viewing the plots in the user interface, we decide on a general class of models (Figure 2.1: (1); D_{ts}, V_{ts}). By adjusting the level of difference and the number of model parameters (Figure 6.4: 2, 3; A_d, A_p, V_{ts}), we identify a so called tentatively entertained model (Figure 2.1: (2); B_m). The adjustment of the relevant faders triggers the system to estimate the parameters of the model (Figure 2.1: (3); A_{data}) and show the resulting diagnostics immediately in the user interface (Figure 6.4: (4); Figure 2.1: 4; V_d, D_{ts}, V_{ts}). The insights (I_h, I_v, I_m) and the domain knowledge (K_{tsa}, K_p, K_m) are again part of the user interaction, but not part of the user interface.

Figure 6.11: VisuTimAlytics Progress of Parameter Order. The user interface shows the change to the visualizations when adjusting the model configuration. This enables the user to evaluate if the next model configuration is better than the previous model configuration. In this figure the continuous vertical line of the order p is moving from $p = 0$ to $p = 1$, which shows a slight improvement in the residuals.

Parameter Adjustment

Another important design requirement was to visualize the change of the plots when adjusting the model parameters. This was achieved by using sliders for the level of difference, continuous vertical lines for the model parameters, and fading the resulting plots with different colors. This process is shown in Figures 6.10, 6.11 and in the next chapter in Figure 7.4, 7.5, and 7.6. Once the slider or a vertical line is dragged from one value in the direction of the next value, the new model configuration is calculated and the plots are seamlessly faded from one display to another by using translucent fading of the bars, points, and lines. The colors for the plots are selected by using the online tool ColorBrewer2[5], which is originally a tool for coloring maps. We used this tool to get a qualitative color scheme, which is easy to distinguish on a screen. This set of colors is used as an endless cyclic sequence for the coloration of the plots. This ensures that each parameter combination is a different color, and the fading process uses always two separate colors.

The toolbox to adjust the parameters and the continuous vertical lines in the ACF/PACF plot

[5] `http://colorbrewer2.org` (18.01.2013)

are shown in Figure 6.7. By default the check box to include the seasonal parameters is disabled. Therefore the seasonal span and the seasonal difference input are disabled and the vertical lines for the order of the seasonal autoregressive and the moving average component of the model are not visible. This ensures that the users do not accidentally fit a seasonal model, if they need a non-seasonal model. By ticking the check box the inputs are enabled and the vertical lines for the seasonal order appear.

In Figure 6.10, we show the fade process when adjusting the level of differencing. This also impacts the ACF/PACF plot, therefore fading it too. In Figure 6.11 one of the continuous vertical lines for the order of the model has been dragged to the right and therefore only the residuals are fading. When sliding the vertical lines for the seasonal order P and Q in the ACF/PACF plot shown in Figure 7.6, the seasonal lags in the ACF/PACF plot are highlighted and any other lags are shown translucent. This supports the user to more easily decide on the seasonal order of the model.

Residual Analysis

To evaluate how well a model configuration is fitted to the time series, the prototype is designed to show visual representations for the model diagnostics. We discussed the model diagnostics in Section 2.7. The goal is to visually explore the remaining part of the time series that is not described by the model, and check if it is likely to be white noise. The visual representation of the residuals in the user interface of the prototype is inspired by the representation used in R. In Figure 6.12 the area displaying the plots for the analysis of the residuals is focused on. There are four different plots. The first plot (1) is the time series plot of the standardized residuals. The second plot (2) is the ACF over the lags of the residuals. The third plot (3) is the normal quantile-quantile plot of the quantile of the standardized residuals against the quantile of the standard normal distribution. The fourth plot (4) is the probability (p values) of the Ljung-Box statistic for each lag. All four of these plots are also included in the interactive fading process presented before. When the user modifies the model configuration, the residual plots are fading from one to the other configuration continuously. This enables the user to see the change of the model configuration and evaluate if the model fitness improves or worsens. This progress is shown in Figure 6.10, 6.11, 7.4, 7.5, and 7.6.

Residual analysis and tests for white noise, which are essentially tests for the randomness of a dataset, are manifold in statistics and there are many implementations of these methods in R. In our implementation of the prototype we focused on the standard tests and visualizations from the standard packages used in the textbooks for time series analysis. Of course it is desirable to enable the user to adjust and customize which tests and visualizations he or she wants to use in the process. This is a possible feature for how to extend the prototype in future work.

6.5 R for Statistical Computing

We already mentioned that the VisuTimAlytics prototype is using the R project for statistical computing [R Development Core Team, 2012]. R is used for the calculations and the time series model estimation. The results are displayed in the visualizations of the prototype using Java

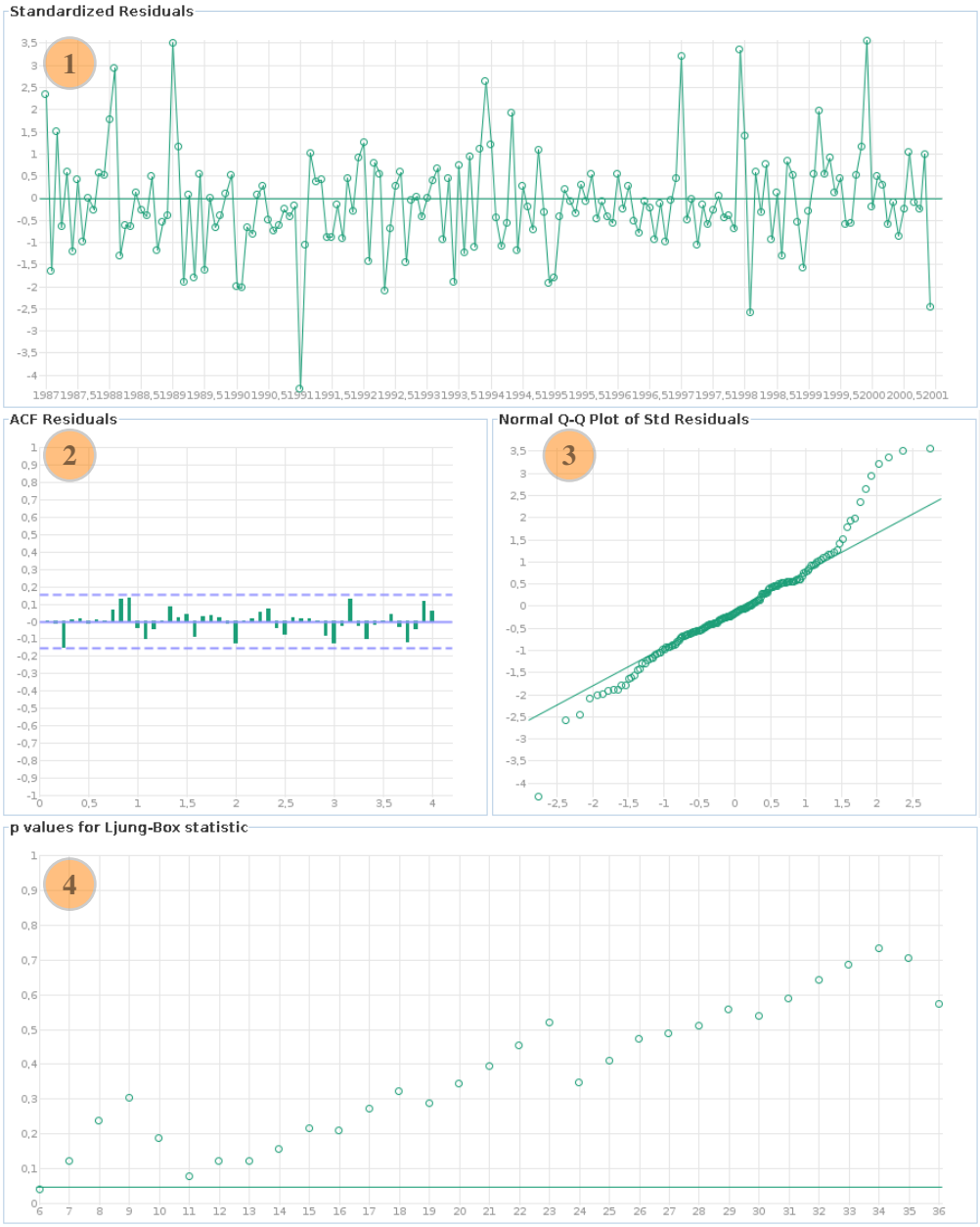

Figure 6.12: VisuTimAlytics Residual Analysis. The figure displays the area for the residual analysis. The plots are (1) the standardized residuals over time, (2) the ACF of the residuals over the lags, (3) the quantile of the standardized residuals against the quantile of the standard normal distribution, and (4) the probability of the Ljung-Box statistics over lags.

and prefuse. R is a widely used system for statistical computing and graphics. It is used by the scientific community in statistics and other research fields that apply statistical methods. One reason why it is so widely used is that it is distributed as free software under a GNU-style copyleft license[6]. Another reason is the large number of statistical procedures that are already included in the basic packages and that ease with which additional packages can be included. There is a very active community that implements new state of the art statistical methods and algorithms in R on a regular basis. In the R Journal[7] this progress is traceable.

Besides the methods in the standard packages, we used more specific methods for time series analysis from additional packages. A good starting point and overview of the packages for time series analysis is the task view[8] for time series analysis. The class `ts` is the default class used to store and calculate time series in R. It only supports numeric equally spaced time series. The index is an ordered numeric number without gaps. Besides the `ts` class we also use the extended class `zoo` based on the `ts` class, which can cope with timestamps as indexes that do not necessarily need to be equally spaced. Another package we heavily used, is the `astsa` package. It was published along with the textbook about time series analysis by Shumway and Stoffer [2011]. Besides directly using the functions from the `astsa` package, we modified some of these functions so that they do not start a graphical window session to show the plots.

6.6 Summary

The objective of this chapter was to provide the results of this thesis, which are the definition of the Visual Analytics process in the time series analysis domain and the design and implementation of this process as a prototype.

We provided the definition of two processes and the consolidation of these processes to one high level process to overcome the stated problems and answer the research question. First we introduced the process for data manipulation and data transformation. This process describes how to adjust the granularity levels and apply the imputation to handle missing values and unequally spaced time series. The second process we presented was the process for progressively selecting and adjusting the model iteratively to get the "best" fitted model. The combined process shows how they are connected iteratively.

We derived the requirements of the prototype using the main research question, the problem statement, and the process definition. We then formulated the requirements as epics and user stories. We selected Java as the programming language to implement the prototype and to use the R project for the statistical computing of the time series and the time series models. For the graphical visualization of the data in Java we decided to use the prefuse toolkit.

We designed the prototype in a way that the workflow of the process definition is mapped to the user interface. The VisuTimAlytics prototype supports the workflow used in time series analysis and guides the user with interactive visualizations and immediate feedback through the process. To compute the models and time series and provide the data for the visualization, we use classes and methods from the standard and additional packages in R. The most important

[6] `http://www.r-project.org/Licenses` (13.01.2013)

[7] `http://journal.r-project.org` (13.01.2013)

[8] `http://cran.r-project.org/web/views/TimeSeries.html` (13.01.2013)

classes for the time series are the `ts` and `zoo` class. Besides the `zoo` package, we used mainly the `astsa` package and some modified functions from this package.

To confirm that the process definition and prototype implementation meet their goals, we evaluate both of them in the following chapter.

CHAPTER 7

Evaluation of the Prototype

In the previous chapter we introduced the Visual Analytics process for time series analysis and the implementation of this process in the VisuTimAlytics prototype. We showed the design of the prototype and how the user can interact. In this chapter, we discuss the evaluation of the process and its implementation, the VisuTimAlytics prototype. In Section 7.1 we provide the details about the dataset we used for the evaluation. We present use case scenarios and describe how the prototype is applied using the dataset in Section 7.2. The presented scenarios then act as a basis for our discussions in Chapter 8 about the benefits of the process and the prototype as well as the limitations and what should be considered for future work. As already mentioned previous chapters, e.g. Chapter 5, this thesis was written as part of the HypoVis project. The project team consists of experts in Information Visualization, Visual Analytics, Human Computer Interaction, User Interface Design, Computational Statistics, and Statistics. The intermediate results of this thesis were presented in the project meetings, where they were reviewed and discussed with the team members. The findings of the reviews are another source for the discussion in Chapter 8.

7.1 Example Dataset

In the introduction, we stated the importance of time series analysis amongst others in the domain of public health and epidemiology. Hence we have chosen a dataset from this domain. The dataset is the daily number of deaths from cardiovascular disease in people aged 75 and older in Los Angeles for the years 1987 to 2000 from the NMMAPS study by Samet et al. [2000]. More specifically, this is a dataset of the environmental epidemiology domain, because it does not only contain data about the number of deaths by cardiovascular diseases, but also the daily mean temperature and the levels of air pollution. The original dataset contains the data of different cities in the United States of America, but we focused on the number of cardiovascular disease deaths in Los Angeles only. However we consider the original dataset for future research, which we will also discuss in Section 8.3.

The relevant columns in the dataset are *date* and *cvd*, which is the daily number of deaths from cardiovascular disease. The date range of the dataset is from 1987-01-01 to 2000-12-31.

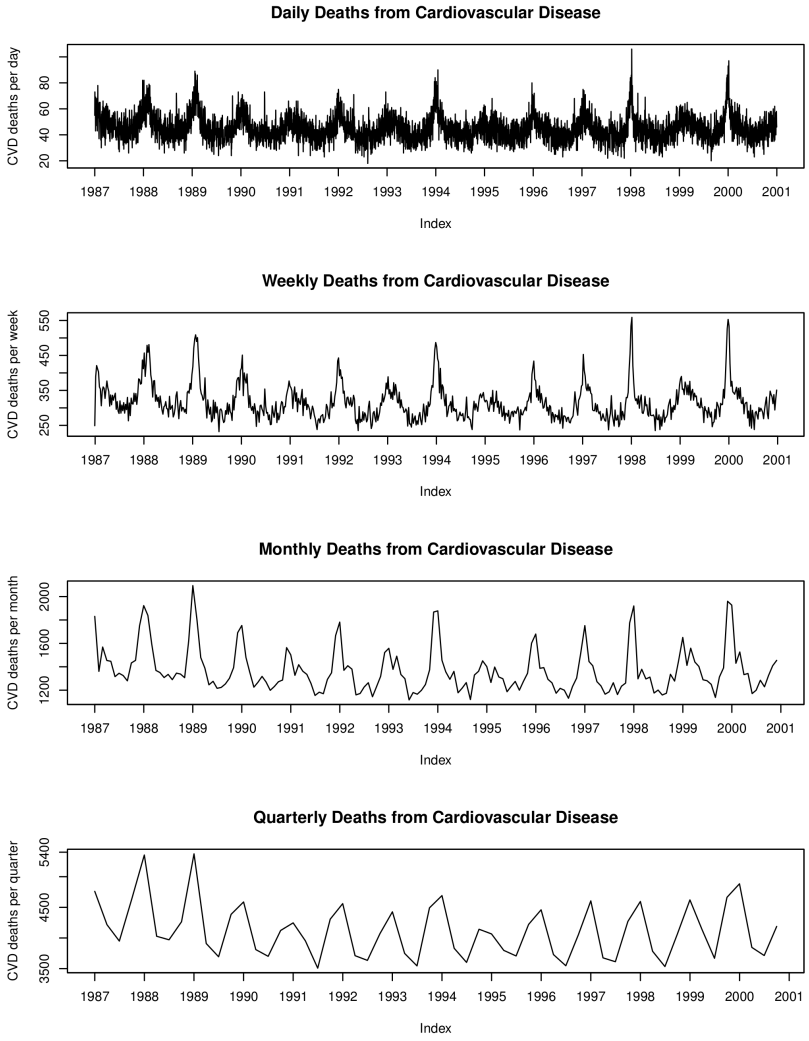

Figure 7.1: Time Series Plot of the Cardiovascular Diseases Deaths Dataset.

There are no missing values in the dataset. The finest granularity of the dataset is day. The number of deaths ranges from 18 to 106, with a mean of 45.11, median of 44, first quantile of 38, and third quantile of 50. Figure 7.1 shows the time series plot in different granularities. The first plot displays the number of deaths from cardiovascular disease per day, the second per week, the third per month, and the fourth per quarter.

To evaluate how the prototype deals with missing values, we use the original dataset, but introduce missing values to the dataset to show the functionality.

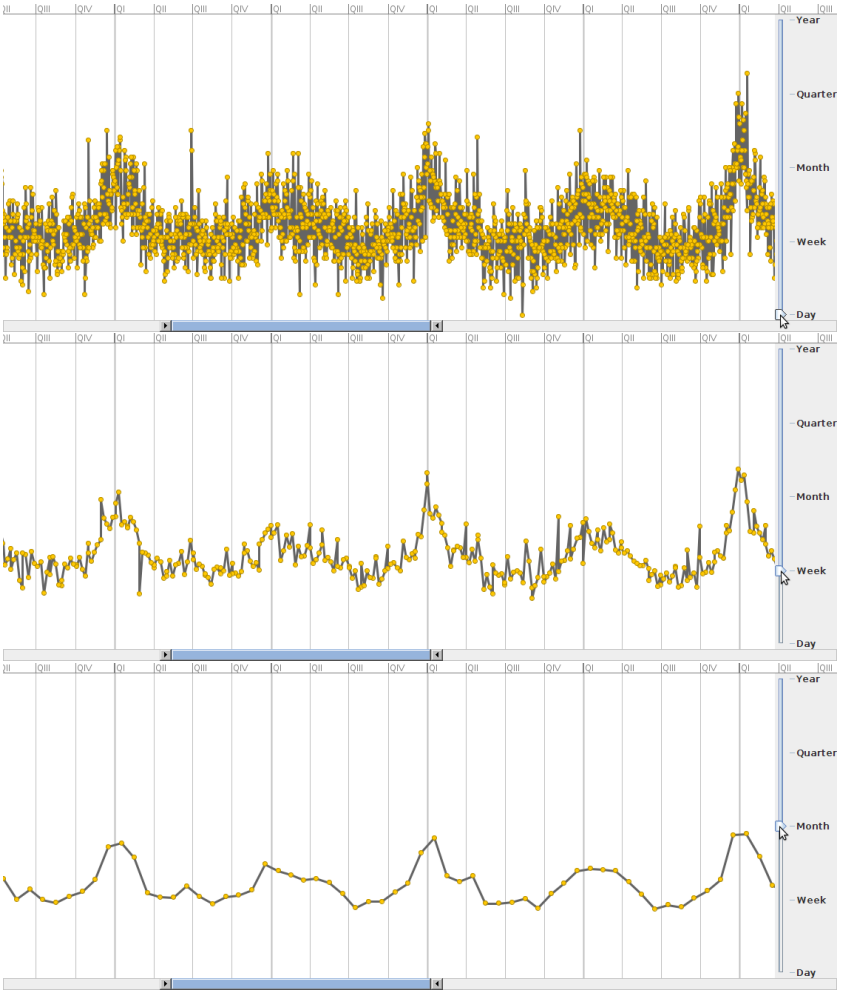

Figure 7.2: VisuTimAlytics Evaluation Granularity Level Day, Week and Month. Showing the process of granularity selection in a higher zoom level.

7.2 Use Case Scenarios

Select the Level of Granularity

The first use case scenario deals with the task to select an appropriate level of granularity for the time series. In Figure 7.1 we already showed the time series plot of the cardiovascular diseases deaths dataset in different granularity levels. The prototype provides an intuitive slider to change the granularity level. In the details about the implementation of the prototype, we stated that it is necessary to provide the information about the granularities along with the dataset. The specification defines what the base granularity of the dataset is, which granularity aggregation function must be used for the granularity steps, and up to which granularity level must be aggregated. In the case of our example dataset the base granularity is day. We use the mean function for each

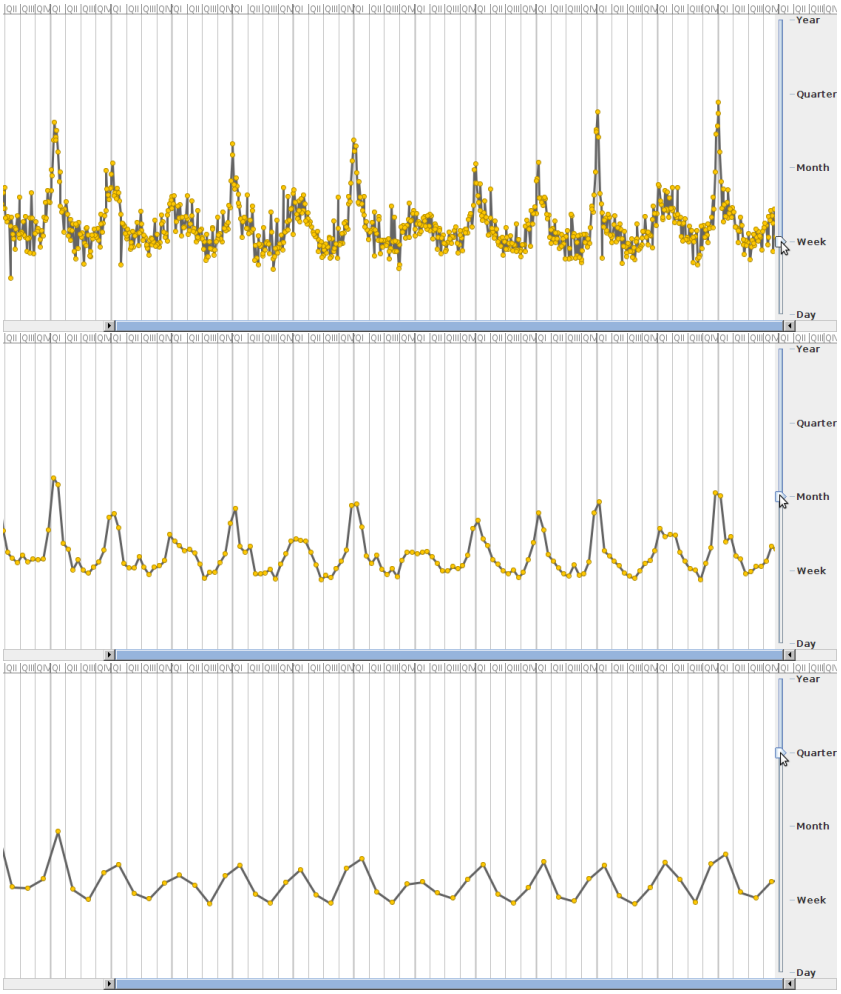

Figure 7.3: VisuTimAlytics Evaluation Granularity Level Week, Month and Quarter. Showing the process of granularity selection.

granularity level to aggregate until we reach the granularity level year. In Figures 7.2 and 7.3 we show this process once with a zoom level showing a range of approximately five years and the other with a longer time range. By operating the slider in the user interface, the user can directly compare the different levels. The daily data in Figure 7.2 is too noisy around the seasonal trend, which we can already identify. The weekly granularity level is much better, but still not perfect. In both, Figures 7.2 and 7.3 the monthly granularity level is better than the weekly granularity level. Figure 7.3 shows that an even higher granularity level, namely the quarterly granularity level, is too flat to show enough variation in the time series. We therefore decide to continue our work with the granularity level month.

Model Selection

In the previous step, we showed how the prototype is applied on the example dataset to select the level of granularity. In this use case scenario we show how to select a suitable model. Following the Box-Jenkins methodology presented in Chapter 2, we first consider the time series plot and the autocorrelation function (ACF) and partial autocorrelation function (PACF) plot. According to the selected granularity level month in the time series plot of the VisuTimAlytics prototype in the following Figure 7.4, we consider that no difference may be needed. Moving the difference slider shown in Figure 6.10 confirms that the change in the ACF/PACF plot, as well as the residual analysis plots is marginal and therefore supports this decision.

We evaluate the ACF/PACF plot according to the behavior of the non-seasonal order of the model in Table 2.1. For the non-seasonal component of the model we decide to have a mixed ARMA model. In Figure 7.4 we show how sliding the parameter p affects the diagnostic plots. The upper display of this figure shows that the adjustment of the non-seasonal AR model to order $p = 1$ results in a more random appearance of the residual time series plot, a more straight line behavior in the normal quantile-quantile plot, and lower lags in the inner-seasonal lags of the ACF plot. In addition to further improvement of the plots in the lower display of this figure, more p-values of the Ljung-Box statistics improve, if the order is changed to $p = 2$. The result of the model configuration $p = 2$, which is an AR(2) model, is shown in the upper display of Figure 7.5. The lower display shows the effect on the residuals by adding a MA component of order $q = 1$ to the model. The lower display shows the transition of this adjustment and creates the diagnostic plots for the assumed mixed ARMA model with $p = 2$ and $q = 1$. This configuration advances the model to show more randomness in the residuals and the diagnostic plots strengthen the assumption that the normalized residuals are standard normal distributed.

The seasonal behavior is not covered by the model yet. Therefore the next step is to adjust the seasonal parts of the model. We consider an autoregressive model, because sliding the parameter P in Figure 7.6 highlights and unveils the seasonal lags and the cut off on seasonal lag 2 in the PACF. This indicates, when consulting Table 2.2 for the behavior of the seasonal order of the model, that the seasonal component is likely to have order $P = 2$. The upper display in the figure shows the improvement of the model when moving from seasonal order $P = 0$ to $P = 1$. The lower display shows the improvement when moving from seasonal order $P = 1$ to $P = 2$. With this configuration, we get a seasonal model of the following form:

$$\text{ARIMA}(p = 2, d = 0, q = 1) \times (P = 2, D = 0, Q = 0)_{s=12}$$

If we estimate the model and include the estimated parameters, we get the following time series model:

$$(1 - 0.3068B^{12} - 0.5444B^{24})(1 + 0.3143B - 0.3112B^2)x_t = (1 - 0.9072B)w_t$$

To support the decision for this model configuration in addition to the residual plots in Figure 7.6, we evaluate the model criteria introduced in Section 2.5. The resulting criteria are shown in Table 7.1. According to this table, the model we selected with the VisuTimAlytics prototype is the one with the minimum values for the AIC and AICc. The BIC for the selected model is one of the smallest, but two other model configurations have a slightly smaller BIC. According

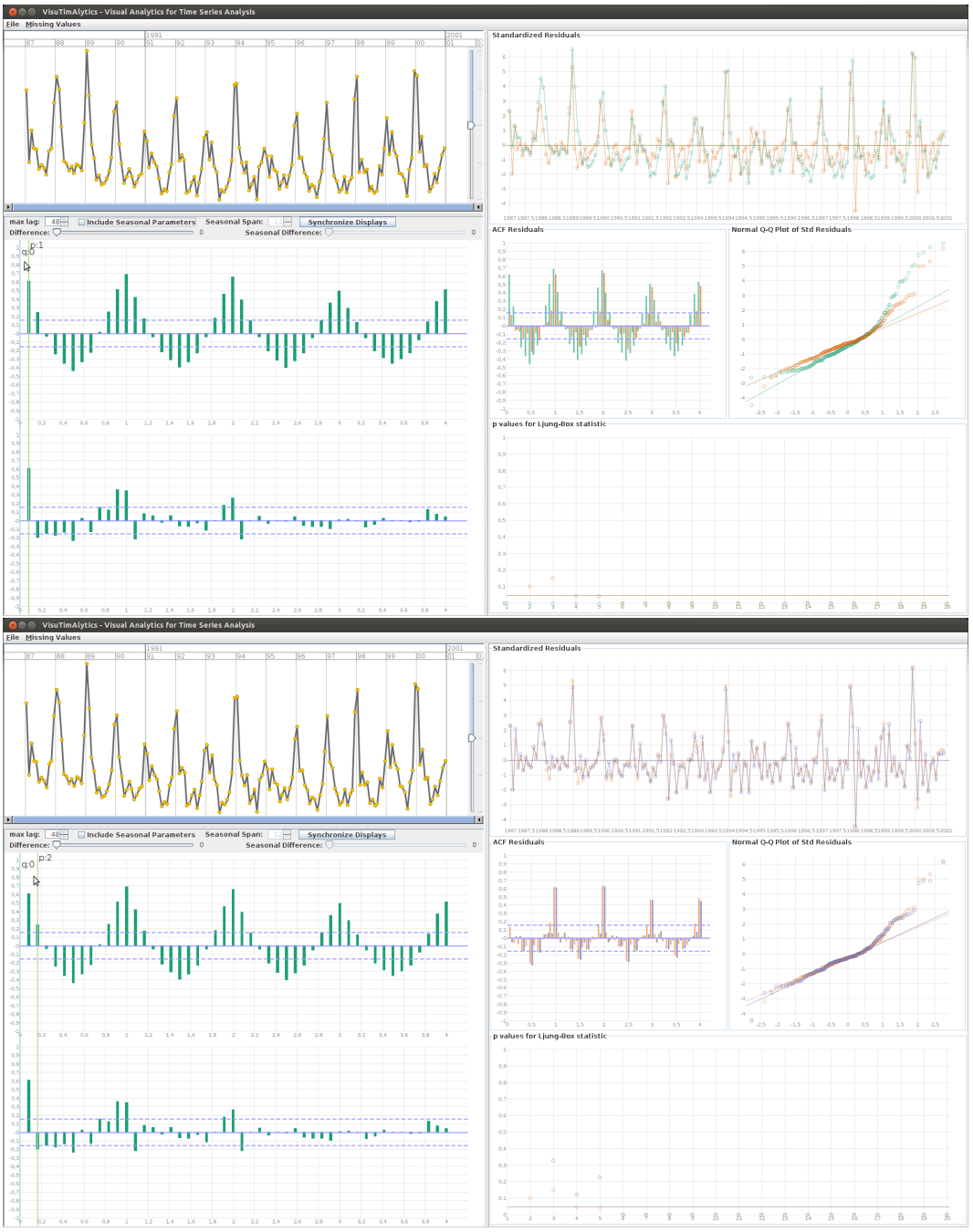

Figure 7.4: VisuTimAlytics Evaluation Slide Progress of AR Component. Fading from $p = 0$ to $p = 1$ and $p = 1$ to $p = 2$ changes the order of the autoregressive model to AR(1) and then to AR(2). The residual plots show the improvement.

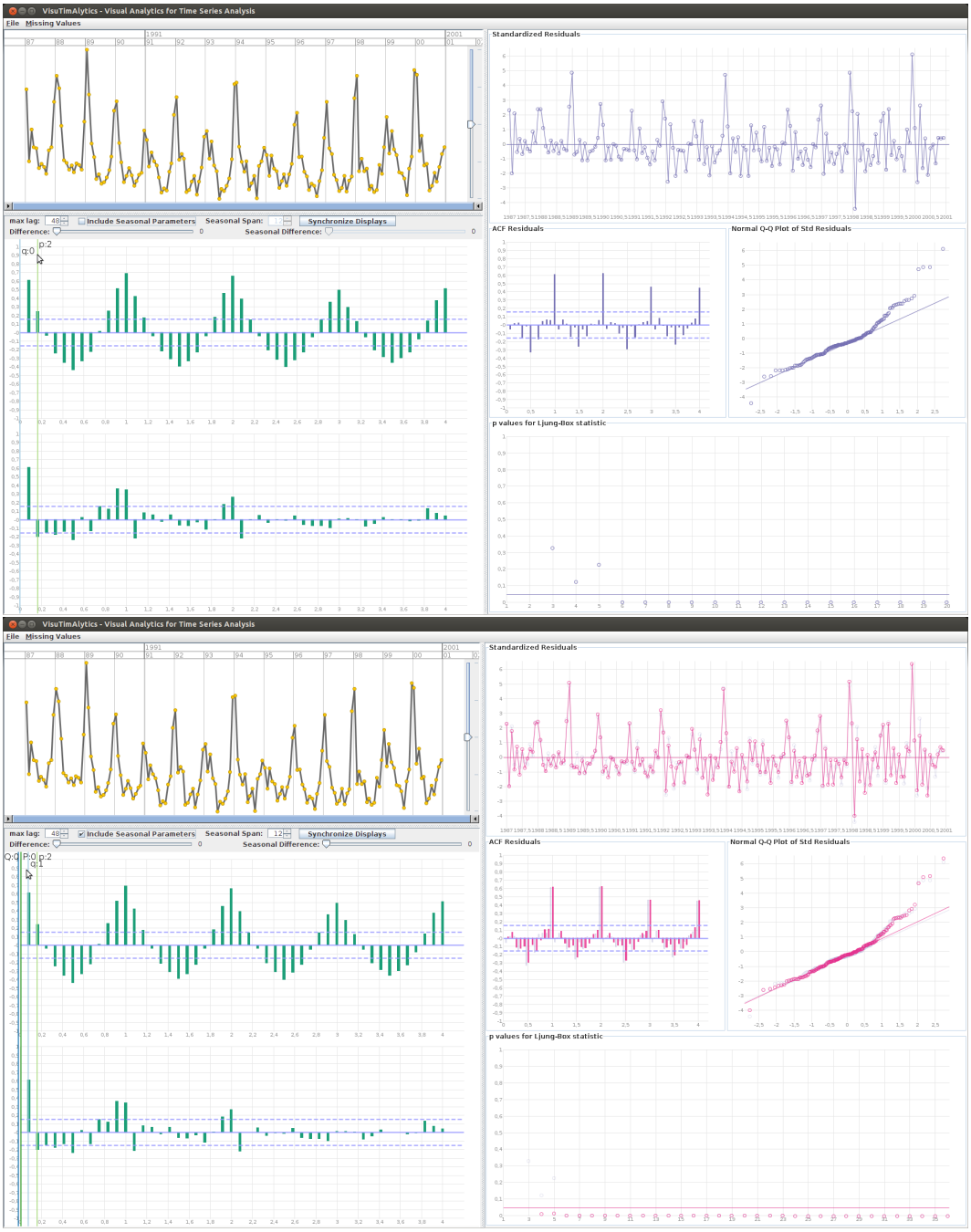

Figure 7.5: VisuTimAlytics Evaluation Slide Progress of MA Component. The upper display shows the results when the model is configured to $p = 2$, which is an AR model order 2. The lower display shows the results of adding a MA component to the model with order $q = 1$.

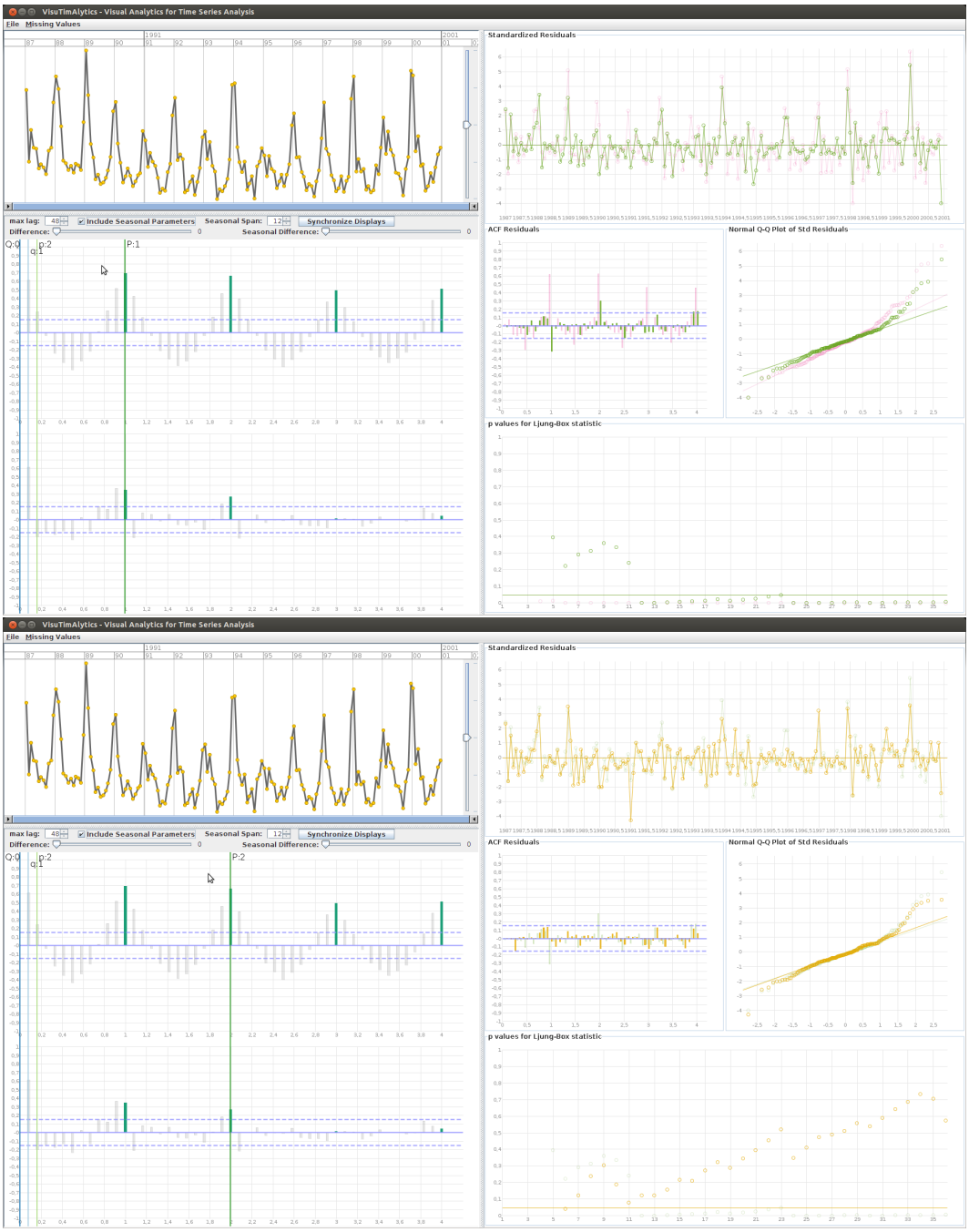

Figure 7.6: VisuTimAlytics Evaluation Slide Progress of Seasonal AR Component. The seasonal lags are highlighted to support the decision for the order of the seasonal components of the model. The upper display shows the sliding progress from $P = 0$ to $P = 1$ and the lower one the progress from $P = 1$ to $P = 2$.

to Shumway and Stoffer [2011] the BIC tends more to prefer rather a model of smaller order than the AIC and the AICc. For that reason and because the difference in the BIC is relatively small the criteria support the decision for the selected model configuration. We use the criteria in this case, to show that we found a well fitted model using the VisuTimAlytics prototype. In Section 2.5, we discussed that these criteria are used in the model specification as an additional source to judge the suitability of the model configuration. We get these values in the prototype when estimating the model parameters, but they are not integrated in the user interface of this prototype version. For future work, we are considering to include these criteria in the graphical user interface.

ARIMA$(p, d, q) \times (P, D, Q)_s$	**AIC**	**AICc**	**BIC**
ARIMA$(0, 0, 0) \times (1, 0, 0)_{12}$	10.7673	10.7801	9.8045
ARIMA$(0, 0, 0) \times (2, 0, 0)_{12}$	10.4573	10.4707	9.5131
ARIMA$(1, 0, 0) \times (1, 0, 0)_{12}$	10.5000	10.5000	9.5800
ARIMA$(1, 0, 0) \times (2, 0, 0)_{12}$	10.2016	10.2157	9.2760
ARIMA$(1, 0, 1) \times (2, 0, 0)_{12}$	10.1860	10.2010	9.2790
ARIMA$(2, 0, 1) \times (2, 0, 0)_{12}$	10.1824	10.1985	9.2940
ARIMA$(3, 0, 1) \times (2, 0, 0)_{12}$	10.1875	10.2048	9.3176

Table 7.1: VisuTimAlytics Evaluation Model Selection Criteria. The different model configurations in the first column and the corresponding model criteria, introduced in Section 2.5, in the other columns exhibit that the model configuration highlighted with the gray background is the "best" fitted model. We selected this model using the VisuTimAlytics prototype.

Range Selection

Using the VisuTimAlytics prototype enables the user to select a range of the time series like we showed in Figure 6.6. A possible use case scenario is to select a trend that starts and ends at a defined point in the time series and only consider this trend for model selection. Another use case scenario for time series containing missing values is to select a connected range to get a subset of the time series without missing values.

In the following use case scenario we use the example dataset with the introduced missing values as shown in Figure 7.7. The scenario is to solve the problem of missing values by using the range selection. We select the range starting with November 1994 and ending with the last data point in the time series. We select the model as described in the previous section. Compared to the complete time series we get a slightly different model, which is simpler and with fewer parameters. In Figure 7.7 the final step of this model selection process is shown. The upper display demonstrates the model configuration with $p = 2$ and seasonal $P = 1$. The ACF plot of the residuals indicates a remaining seasonal autocorrelation. In this case we consider a seasonal difference of $D = 1$ as a possible solution to remove this autocorrelation in the residuals. Sliding the seasonal difference fader as shown in the lower display of this figure,

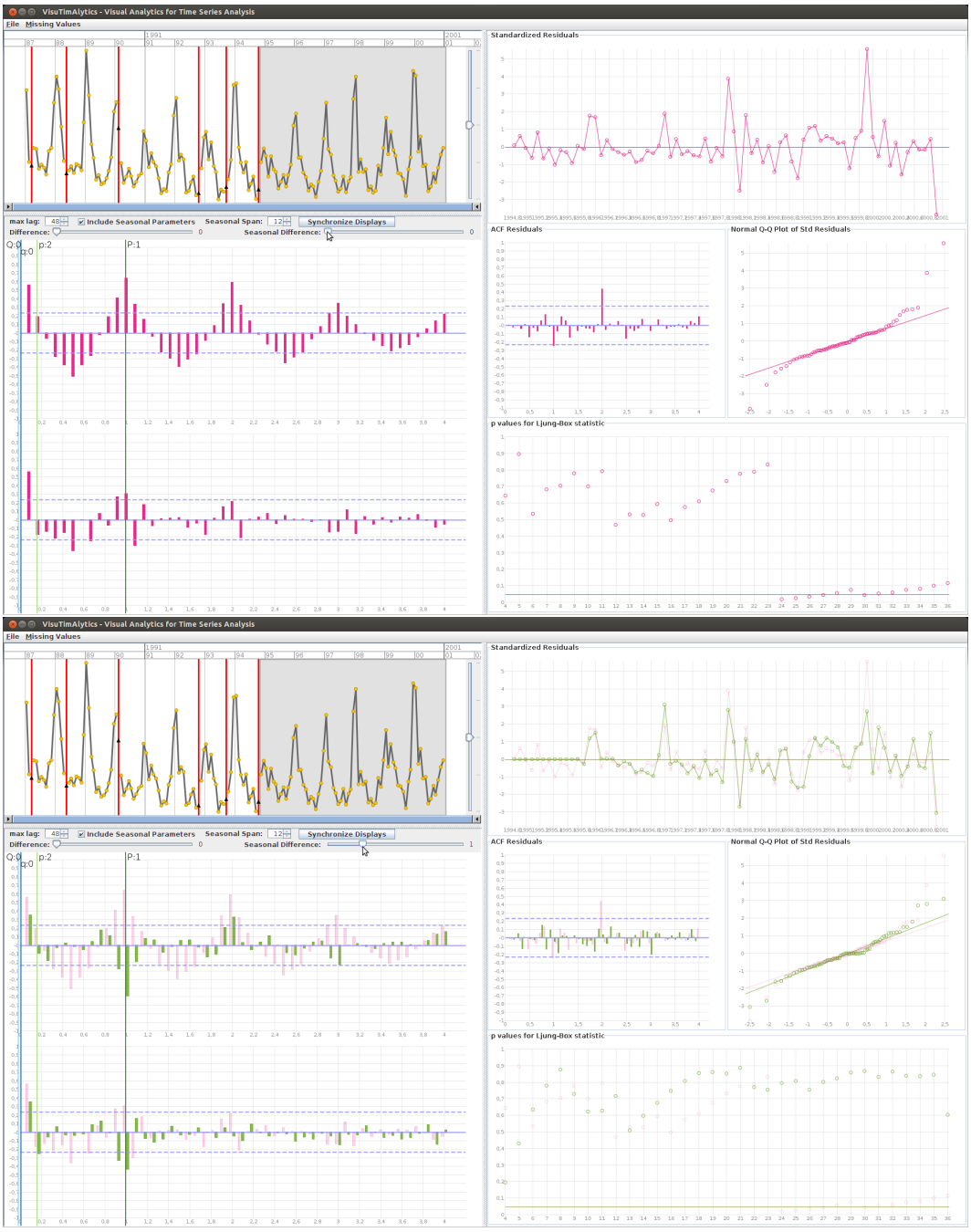

Figure 7.7: VisuTimAlytics Evaluation Range Selection. The model is selected based on a subset of the time series. Here the range highlighted in gray is selected from the time series plot. The progress of no seasonal differencing in the upper display and fading to a seasonal difference $D = 1$ in the lower display is shown.

Figure 7.8: VisuTimAlytics Evaluation Missing Values Estimation using Seasonal Kalman Filter. The figure shows the estimated values without manual adjustment and the diagnostic plots for the model configuration. The red box indicates the region where the remaining pattern in the residual time series plot is visible.

improves this configuration. Finally we get a model with the following configuration:

$$\text{ARIMA}(p = 2, d = 0, q = 0) \times (P = 1, D = 1, Q = 0)_{s=12}$$

Missing Values

Besides using a subset of the time series, we can use granularity aggregation to overcome the problems of missing values described above. In our example dataset we introduced missing values in the granularity level day. For the first use case scenario we ignore the missing values in the aggregation. In this case we overcome the problem of missing values by choosing a higher level of granularity, ending up with a time series with granularity level month without missing values.

For the second use case scenario, we choose not to ignore the missing values. As a result there are remaining missing values in the granularity level month, as shown in Figures 6.8 and 7.7. The scenario is to estimate the missing values using automated computation. The missing values in these figures are estimated using linear interpolation. In our case the estimated values are very good approximations, because the time series only has a few single missing values. On

Figure 7.9: VisuTimAlytics Evaluation Missing Values Estimation using Manual Adjustment. In this figure the previously estimated values using the seasonal Kalman Filter are adjusted to the visible seasonal pattern. With this adjusted estimated values there is no remaining pattern in the residual time series. The red box indicates the same region in the residual time series plot as in Figure 7.8 without remaining pattern.

the resulting time series including the estimated values, it is then possible to fit a time series model. Because the estimated values are approximately like the original data values, the fitted model is the same as presented in the use case scenario for the model selection.

For a third use case scenario, we introduce a connected sequence of missing values. The missing values are estimated using a seasonal Kalman filter. For the theoretical background of the Kalman filter we refer to the book by Shumway and Stoffer [2011, p. 325–335]. In Figure 7.8 the sequence of missing values and the estimated values are shown. The major difference in the model diagnostic plots is in the time series plot of the standardized residuals. Although the other plots show that the model configuration is still very good for the time series, the residual time series plot shows evidence that there is a remaining structure in the residuals that is not covered by the model see the region indicated by the red box in in the residual time series plot of Figure 7.8. This is due to the imperfect estimation of the missing values and especially due to the missing trough between the two spikes. This trough is expected to appear based on the seasonal pattern that is visible in the time series.

To continue this use case scenario, we adjust the previously estimated values manually to approximate the seasonal reoccurring cycles. We end up with adjusted values as shown in Fig-

ure 7.9. Although these adjusted values do not follow the original time series exactly, they are a very good approximation and the diagnostic plots show that there is no remaining pattern in the residuals as before. The same region is indicated by the red box in the residual time series plot.

This use case scenario shows the strength of combining human perception and automated methods. For the user it is relatively easy to adjust the values to the seasonal pattern that is visible in the time series plot. We showed that even a good time series estimator cannot achieve a comparable result to this manual adjustment.

7.3 Summary

In this chapter, we evaluated the Visual Analytics process for time series analysis and the implementation of this process in the VisuTimAlytics prototype by using the prototype to actually work on an example dataset. The example dataset that we selected for the evaluation is from environmental epidemiological research. The dataset represents the number of deaths from cardiovascular diseases in people aged 75 and older in Los Angeles for the years 1987 to 2000. We presented this dataset and the key features of the dataset. Using use case scenarios, we showed how the prototype is applied and how the user is supported in the process to solve the given tasks. We considered the final discovered time series model as a good model for the example dataset and supported this decision with further statistics. In the use case scenario to estimate missing values, we highlighted the major strengths of manual adjustment using an interactive user interface over automated methods for this time series.

CHAPTER 8

Summary and Conclusion

In Chapter 1 we gave an overview of the content of this thesis and stated that the main objective is to design a Visual Analytics process and implement this process in the domain of time series analysis. To achieve this objective we studied the details of this domain in Chapter 2. Our focus was the class of ARIMA and seasonal ARIMA models and the Box-Jenkins methodology for model selection. We also considered the challenge of missing values and unequally spaced time series. To overcome the problems stated briefly in Chapter 1, we studied the state of the art methods of Visual Analytics in Chapter 3. We investigated existing Visual Analytics processes, structures and properties of time-oriented data, as well as visualization techniques. Taking into account the findings of this research, we formulated the research question and the hypotheses for this thesis in Chapter 4. In Chapter 5 we presented the scientific approach used to apply our research findings to achieve the anticipated results and how we evaluated the prototype. In Chapter 6 we showed how our research findings helped us to solve the problems stated. The result was the definition of a tailored Visual Analytics process in Section 6.1 and the discussion about the implementation of this process as a prototype and the design of the prototype in Section 6.4. To answer the research question we formulated use case scenarios and evaluated the prototype on an example dataset in Chapter 7.

8.1 Conclusion

In Chapter 2 we studied the problem domain of time series analysis. From our findings we conclude that there is a well recognized and widely used process for the class of ARIMA and seasonal ARIMA models in time series analysis. This iterative process, named the Box-Jenkins methodology, describes how to find an appropriate model that fits the time series. We found that the separate steps of this process are implemented in the major statistical software tools we investigated, but that the overall process and especially the workflow of this process are not supported in an intuitive and user friendly way. For the discussion about the tool support, we refer to Section 2.8. We determined that unequally spaced time series or time series containing

missing values are a special challenge. The ways to handle these cases are either to use advanced specialized methods in the Box-Jenkins process or to impute the missing values and transform the unequally spaced time series to equally spaced time series. The latter enables the user to use the standard methods and to understand what happens in the separate steps of the process. We recognize that there may be other possible transformations that could be applied to a time series as a data pre-processing step. Our focus however, is on the overall process of model selection including a simple and intuitive way to overcome the complexity that is introduced by unequally spaced time series and missing values. The investigation of the separate steps of the Box-Jenkins methodology and their theoretical underpinnings unveiled that although it is possible to support the user in this process by using visual exploration of the time series, it is still necessary for the user to have a good level of domain knowledge about time series analysis. The visual representations of the time series and the residuals of the selected model need to be known by the user to decide on and adjust the model configuration. All in all we conclude that the problem domain of time series analysis, more precisely the Box-Jenkins methodology and the problem specified in Chapter 4, is an exciting target problem to apply Visual Analytics methods to and overcome the discussed challenges.

To support our discussion about the application of Visual Analytics methods to the target problem, we studied Visual Analytics and time-oriented data in Chapter 3. From the definition of the field of Visual Analytics alone, we concluded that it is a possible way to solve the problems we tried to overcome. The exploratory analysis used in time series analysis and the statistical computations discussed in Chapter 2, already demand a close intertwinedness of human reasoning and automated methods. The investigation of the generic Visual Analytics process and the more specific process for time-oriented data confirmed that conclusion. Through the analysis of the characteristic and structure of time, we discovered that the concept of time granularity is a way to use the structure of time for the aggregation of time series in time series analysis. The visualization of time-oriented data and the survey of visualization techniques supported the ideas for visualizing the data in the process of time series analysis. The related work we studied, showed that no attempt exists to apply Visual Analytics methods to the problem of model selection in time series analysis from the perspective of Visual Analytics. The related work however showed some inspirations on how to display time series data and which interaction mechanisms could possibly be applied. We concluded that Visual Analytics and the Visual Analytics processes are a possible way to overcome the challenges stated in Chapter 4 as our problem domain.

Based on these conclusions and insights we designed the Visual Analytics process for time series analysis. Along with the evolution of the process design, we formulated the requirements for the prototype. We then implemented the VisuTimAlytics prototype based on the process definition and the formulation of the requirements. These results gained from testing the prototype using tailored use case scenarios are the foundation to answer the research question and support the hypotheses. The presentation and discussion of the user interface and the visualizations used in the prototype in Section 6.4 describe very well how Visual Analytics supports the challenge in our problem domain. The application of the VisuTimAlytics prototype according to the use case scenarios on the example dataset in Chapter 7 strengthens the arguments for the hypotheses. To answer the main research question, which we stated in Chapter 4, we recap the question here

- **How can Visual Analytics support the process of model building for time series analysis and help to choose the best transformation for unequally spaced time series and missing values?**

and finally conclude that the presentation of the process definition and the prototype design, the details about the visualizations and interaction mechanisms we used, and how this all is applied to an example dataset following use case scenarios, answers this question and shows how Visual Analytics supports the user to solve the stated problems. The research hypotheses are supported in the same way as stated here for the research question.

8.2 Main Contribution

The contributions of this thesis are evident in all chapters and are included in the discussion and conclusion of Section 8.1. In the following list, we provide the main contributions in a summarized way:

- Identification and specification of the target problem in the problem domain of time series analysis
- Investigation of the relevant Visual Analytics methods and visualization techniques for the target problem
- Formulation of a Visual Analytics process for the stated problem of model selection and time series transformation
- Implementation of this process as a prototype.
- Definition of use case scenarios to evaluate the prototype
- Discussion of the application of the prototype to an example dataset.

8.3 Future Work

The current version of the prototype implementation is not fully matured. The prototype was tested with the example dataset we introduced in Section 7.1, which includes approximately 5000 data values in the granularity level of one day. For larger time series it is necessary to improve the prototype to get a good performance. Another improvement to consider is to extend the prototype to directly support more and other statistical methods that are implemented in R. One idea would be to use robust methods to prevent the influence of outliers.

The support for the feedback loop, which we discussed in the definition of our combined Visual Analytics process, is not directly implemented in the current version of the prototype. The idea is to use the fitted model to estimate the missing values to get better estimations and refit the model to the time series, to get again a better model.

The diagnostic of the time series model is at the moment limited to the diagnostic plots. For future work, it would be interesting to include the performance of the model for forecasting the diagnostic step. This could be done by using the first part of the time series and the model to calculate the forecast for the next time point then use the actual value of this next time point to forecast the next but one. This could be continued for the rest of the time series. This single–

step–ahead–forecast time series could be shown along to the actual time series, which would then be another way to decide how well the model fits the time series.

The transformation of unequally spaced time series to equally spaced time series is limited in the current implementation of the prototype to use the lowest granularity as lattice to aggregate to an equally spaced time series. The higher granularities are then calculated based on the values of the lowest granularity. It would be beneficial to be able to first select the granularity using the structure of time, which is then used as the basic lattice for the transformation.

Finally it is necessary to conduct a user study to evaluate the process and the prototype. An interesting question for this user study is how much easier and/or how much faster it is for different skilled domain experts or other users to work on the time series and fit a model to the time series.

Bibliography

Aigner, W. (2006). *Visualization of time and time-oriented information: challenges and conceptual design*. PhD thesis, Vienna University of Technology, Institute of Software Technology and Interactive Systems, Supervisors: Silvia Miksch (Vienna University of Technology), Heidrun Schumann (University of Rostock).

Aigner, W., Miksch, S., Schumann, H., and Tominski, C. (2011). *Visualization of Time-Oriented Data*. Human-Computer Interaction Series. Springer, London.

Beck, K. (2000). *Extreme Programming Explained: Embrace Change*. Addison-Wesley, Reading, MA.

Bernard, J., Ruppert, T., Goroll, O., May, T., and Kohlhammer, J. (2012). Visual-interactive preprocessing of time series data. In Kerren, A. and Seipel, S., editors, *Proceedings of SIGRAD 2012, Interactive Visual Analysis of Data, Växjö, Sweden, November 29-30, 2012*, volume 81 of *Linköping Electronic Conference Proceedings*, pages 39–48. Linköping University Electronic Press.

Berry, L. and Munzner, T. (2004). Binx: Dynamic exploration of time series datasets across aggregation levels. In *Poster Compendium of IEEE Symposium on Information Visualization (InfoVis)*, pages 5–6, Los Alamitos, CA, USA. IEEE Computer Society.

Bertini, E. and Lalanne, D. (2009). Surveying the complementary role of automatic data analysis and visualization in knowledge discovery. In *Proceedings of VAKD09*, pages 12–20, Paris, France. ACM.

Bettini, C., Jajodia, S., and Wang, S. (2000). *Time granularities in databases, data mining, and temporal reasoning*. Springer, New York, NY, USA.

Bisgaard, S. and Kulahci, M. (2011). *Time Series Analysis and Forecasting by Example*. Wiley Series in Probability and Statistics. John Wiley & Sons, Hoboken, New Jersey, USA.

Box, G. and Jenkins, G. (1970). *Time Series Analysis: Forecasting and Control*. Holden-Day, San Francisco.

Box, G., Jenkins, G., and Reinsel, G. (1976). *Time Series Analysis: Forecasting and Control*. Wiley Series in Probability and Statistics. Holden-Day, San Francisco, USA, 3rd edition. Series G.

Box, G., Jenkins, G., and Reinsel, G. (2008). *Time Series Analysis: Forecasting and Control*. Wiley Series in Probability and Statistics. John Wiley & Sons, Hoboken, New Jersey, USA, 4th edition.

Box, G. E. P. and Pierce, D. A. (1970). Distribution of residual autocorrelations in autoregressive-integrated moving average time series models. *Journal of the American Statistical Association*, 65(332):pp. 1509–1526.

Brockwell, P. and Davis, R. (2002). *Introduction to Time Series and Forecasting*. Springer, New York, NY, USA, 2nd edition.

Brockwell, P. J. (2011). Time series. In Lovric, M., editor, *International Encyclopedia of Statistical Science: SpringerReference (`www.springerreference.com`)*. Springer-Verlag Berlin Heidelberg. DOI: 10.1007/SpringerReference_205677 2011-05-09 09:18:39 UTC.

Brockwell, P. J. and Davis, R. A. (1991). *Time Series: Theory and Methods*. Springer Series in Statistics. Springer, New York, NY, USA, 2nd edition.

Buono, P., Aris, A., Plaisant, C., Khella, A., and Shneiderman, B. (2005). Interactive Pattern Search in Time Series. In *Proceedings of Conference on Visualization and Data Analysis, VDA 2005*, pages 175–186, Washington.

Buono, P., Plaisant, C., Simeone, A., Aris, A., Shneiderman, B., Shmueli, G., and Jank, W. (2007). Similarity-based forecasting with simultaneous previews: A river plot interface for time series forecasting. In *Proceedings 11th International Conference Information Visualisation (IV'07)*, pages 191–196, Zurich, Switzerland. IEEE Computer Society.

Cleveland, W. S. (1993). *Visualizing Data*. Hobart Press, Summit, New Jersey, USA.

Cohn, M. (2004). *User Stories Applied: For Agile Software Development*. Addison-Wesley, Boston.

Cohn, M. (2010). *Succeeding with Agile: Software Development Using Scrum*. The Addison-Wesley Signature Series. Addison-Wesley, Upper Saddle River, NJ.

Cowpertwait, P. S. and Metcalfe, A. V. (2009). *Introductory Time Series with R*. Use R! Springer, New York, NY, USA.

Cryer, J. D. and Chan, K.-S. (2008). *Time Series Analysis. With Applications in R*. Springer Texts in Statistics. Springer, New York, NY, USA, 2nd edition.

Dachselt, R., Frisch, M., and Weiland, M. (2008). Facetzoom: a continuous multi-scale widget for navigating hierarchical metadata. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1353–1356, Florence, Italy. ACM, New York, NY, USA.

Eckner, A. (2012). A framework for the analysis of unevenly-spaced time series data. (Working Paper, Version: November 19, 2012) `http://eckner.com/papers/unevenly_spaced_time_series_analysis.pdf` (13.01.2013).

Einstein, A. (1934). On the method of theoretical physics. *Philosophy of science*, 1(2):163–169.

Hamilton, J. D. (1994). *Time series analysis*. Princeton University Press, Princeton, New Jersey, USA.

Heer, J., Card, S. K., and Landay, J. A. (2005). prefuse: A toolkit for interactive information visualization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2005)*, pages 421–430, Portland, Oregon, USA. ACM, New York, NY, USA.

Jones, R. H. (1985). Time series analysis with unequally spaced data. In Hannan, E. J., Krishnaiah, P. R., and Rao, M. M., editors, *Handbook of statistics*, volume 5, pages 157–178. Elsevier, Amsterdam, Netherlands.

Keim, D., Ankerst, M., and Kriegel, H. (1995). Recursive pattern: A technique for visualizing very large amounts of data. In *Proceedings of the 6th conference on Visualization 95*, pages 279–286, Atlanta, GA. IEEE Computer Society.

Keim, D., Kohlhammer, J., Ellis, G., and Mansmann, F. (2010). *Mastering The Information Age - Solving Problems with Visual Analytics*. Eurographics Association, Goslar, Germany.

Keim, D. A., Mansmann, F., Schneidewind, J., Thomas, J., and Ziegler, H. (2008). Visual Analytics: Scope and Challenges. In Simoff, S., Boehlen, M. H., and Mazeika, A., editors, *Visual Data Mining: Theory, Techniques and Tools for Visual Analytics*. Springer, Berlin, Heidelberg. Lecture Notes in Computer Science (LNCS).

Kowarik, A., Meraner, A., Schopfhauser, D., and Templ, M. (2012). Interactive adjustment and outlier detection of time dependent data in R. In *Conference of European Statisticians, Work Session on Statistical Data Editing*, Oslo, Norway. United Nations - Economic Commission for Europe. `http://www.unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.44/2012/31_Austria.pdf` (18.01.2013).

Lammarsch, T. (2010). *Facets of Time: Making the Most of Time's Structure in Interactive Visualization*. PhD thesis, Vienna University of Technology, Institute of Software Technology and Interactive Systems, Supervisors: Silvia Miksch (Vienna University of Technology), Daniel Keim (University of Konstanz).

Lammarsch, T., Aigner, W., Bertone, A., Gartner, J., Mayr, E., Miksch, S., and Smuc, M. (2009). Hierarchical temporal patterns and interactive aggregated views for pixel-based visualizations. In *Proceedings of the International Conference Information Visualisation (IV)*, pages 44–49, Los Alamitos, CA, USA. IEEE.

Lammarsch, T., Aigner, W., Bertone, A., Miksch, S., and Rind, A. (2011). Towards a concept how the structure of time can support the visual analytics process. In Miksch, S. and Santucci, G., editors, *Proceedings of International Workshop on Visual Analytics (EuroVA 2011) in conjunction with EuroVis*, pages 9–12, Bergen, Norway.

Ljung, G. M. and Box, G. E. P. (1978). On a measure of lack of fit in time series models. *Biometrika*, 65(2):pp. 297–303.

Mintz, D., Fitz-Simons, T., and Wayland, M. (1997). Tracking air quality trends with SAS/GRAPH. In *Proceedings of the 22nd Annual SAS User Group International Conference (SUGI97)*, pages 807–812, Cary, NC, USA. SAS.

Müller, W. and Schumann, H. (2003). Visualization methods for time-dependent data-an overview. In *Simulation Conference, 2003. Proceedings of the 2003 Winter*, volume 1, pages 737–745, New Orleans, LA. IEEE.

Munzner, T. (2008). Process and pitfalls in writing information visualization research papers. In Kerren, A., Stasko, J., Fekete, J.-D., and North, C., editors, *Information Visualization*, volume 4950 of *LNCS*, pages 134–153, Heidelberg. Springer.

Pfaff, B. (2008). *Analysis of Integrated and Cointegrated Time Series with R*. Use R! Springer, New York, NY, USA.

R Development Core Team (2012). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Samet, J. M., Dominici, F., Zeger, S. L., Schwartz, J., and Dockery, D. W. (2000). The national morbidity, mortality, and air pollution study. part I: Methods and methodologic issues. Research Report 94, Part I, Health Effects Institute, Cambridge, MA, USA.

Shneiderman, B. (1996). The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pages 336–343, Boulder, Colorado, USA. IEEE.

Shore, J. and Warden, S. (2008). *The Art of Agile Development*. O'Reilly, Sebastopol, CA.

Shumway, R. H. and Stoffer, D. S. (2011). *Time Series Analysis and its Applications. With R examples.* Springer Texts in Statistics. Springer, New York, NY, USA, 3rd edition.

Templ, M., Alfons, A., and Filzmoser, P. (2012). Exploring incomplete data using visualization techniques. *Advances in Data Analysis and Classification*, 6(1):1–19.

Thomas, J. J. and Cook, K. A. (2005). *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. IEEE Computer Society, Los Alamitos, CA, USA.

Tominski, C. and Schumann, H. (2008). Enhanced interactive spiral display. In *Proceedings of the Annual SIGRAD Conference, Special Theme: Interactivity*, pages 53–56, Stockholm, Sweden. Linköping University Electronic Press, Linköpings universitet.

Urbanek, S. (2011). rjava: Low-level r to java interface. R package version 0.9-3.

Weber, M., Alexa, M., and Müller, W. (2001). Visualizing time-series on spirals. In *Information Visualization, 2001. INFOVIS 2001. IEEE Symposium on*, pages 7–13, San Diego, California. IEEE.