DIPLOMARBEIT

Statistical models for multivariate (compositional) count time series

Zur Erlangung des akademischen Grades

Diplom-Ingenieur

Im Rahmen des Masterstudiums

Wirtschaftsmathematik und Statistik

Ausgeführt am

Institut für Stochastik und Wirtschaftsmathematik
Fakultät für Mathematik und Geoinformation
Technische Universität Wien

Unter der Anleitung von

Univ.Prof. Dipl.-Ing. Dr.techn. Peter Filzmoser

Eingereicht von

Alexander Schwaiger, BSc

Matrikelnummer: 11775205

Wien, am 3. August 2023

Alexander Schwaiger (Verfasser) Peter Filzmoser (Betreuer)

Abstract

In this thesis, we analyse and compare two approaches for multivariate count data time series with an excessive amount of zeros. The first approach belongs to the class of generalised linear models (GLM) and fits a univariate integer-valued generalized autoregressive conditional heteroskedasticity model of order (p,q) (INGARCH(p,q) model) for each dimension. The second approach is based on compositional data analysis (CoDA) and uses the relative structure of our data to build a vectorised autoregressive (VAR) model from it. In addition, we also consider alternative options like zero-inflated models (ZIM) and integer-valued autoregressive (INAR) models. Providing the mathematical background for the INGARCH(p,q) and CoDA approach and exploring different parameter settings for them, we evaluate their performance on real world data and compare different tuning options. We then introduce an error measure for comparison and use it to compare the performance on different time series. We provide a handbook of our analysis in the statistical software R and present the used packages and functions. At last, we show the results of our analysis. All models outperform the naive random walk model, but they cannot take all three major characteristics, integer-valued, multivariate and excessive amount of zeros, simultaneously into account.

Keywords: Compositional Data Analysis, General Linear Models, INGARCH, Multivariate Count Data Time Series, R

Kurzfassung

In dieser Diplomarbeit werden zwei Ansätze für multivariate Zähldaten-Zeitreihen mit einer überproportionalen Anzahl von Nullen analysiert und verglichen. Der erste Ansatz gehört zu der Klasse der verallgemeinerten linearen Modelle (GLM). Dabei wird ein ganzzahliges verallgemeinertes autoregressives Model mit bedingter Heteroskedastizität der Ordnung (p,q) (INGARCH(p,q) Model) für jede Dimension gefitted. Der zweite Ansatz basiert auf der Analyse von Kompositionsdaten (CoDA) und nutzt die relative Struktur der Daten, um daraus ein vektorisiertes autoregressives Model (VAR) zu erstellen. Darüber hinaus betrachten wir auch alternative Optionen wie Zero-Inflation-Modelle (ZIM) und ganzzahlige autoregressive Modelle (INAR). Wir beschreiben den mathematischen Hintergrund des INGARCH(p,q)- und des CoDA-Ansatzes, untersuchen verschiedene Parametereinstellungen, vergleichen Tuning-Optionen und testen die Modelle mit realen Daten. Anschließend führen wir ein Fehlermaß zum Vergleich ein und verwenden es, um die Güte der Modelle bei verschiedenen Zeitreihen zu vergleichen. Wir stellen ein Benutzerhandbuch für unsere Analyse in der Statistiksoftware R zur Verfügung und erklären die verwendeten Pakete und Funktionen. Zuletzt präsentieren wir die Ergebnisse der Analyse. Unsere Modelle liefern bessere Ergebnisse als eine zufällige stochastische Irrfahrt, können aber nicht die drei Haupteigenschaften, Multivariabilität, ganzzahlige Werte und überproportionale Anzahl an Nullen, gleichzeitig berücksichtigen.

Schlagworte: Compositional Data Analysis, General Linear Models, INGARCH, Multivariate Count Data Time Series, R

Eidesstattliche Erklärung

Ich erkläre an Eides statt, dass ich die vorliegende Diplomarbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt bzw. die wörtlich oder sinngemäß entnommenen Stellen als solche kenntlich gemacht habe.

Wien, am 3. August 2023

Alexander Schwaiger

Acknowledgements

At this point, I would like to express my sincere gratitude to my supervisor Univ.Prof. Dipl.-Ing. Dr.techn. Peter Filzmoser for the help he provided in the form of resources, answers to my questions and in general the excellent guidance during my thesis. In this context I also have to thank Dipl.-Ing. Lukas Neubauer who worked on this project with me and supported me not only in the analysis and interpretation of the results but also with suggestions and insightful comments. I also want to thank Schrankerl GmbH for providing the data used for carrying out the analyis.

Furthermore, I would like to thank my family and especially my parents who gave me the opportunity to study and supported me the whole time. I also want to thank my friends and colleagues who accompanied me during this journey and made this an unforgettable time.

Contents

Abstract i

1 Introduction 1

1.1 Motivation . . . 1
1.2 Data Description . . . 1
1.3 Outlook . . . 3

2 Count Time Series Models 5

2.1 Motivation . . . 5
2.2 INGARCH Model . . . 7
2.2.1 Multivariate INGARCH Model . . . 9
2.3 GARCH Models . . . 12
2.3.1 Parameter Estimation and Forecasting . . . 12
2.3.2 Testing for GARCH Models . . . 13
2.3.3 Applications . . . 13
2.4 Naive Random Walk . . . 13
2.5 Zero-Inflated Models . . . 14
2.5.1 Parameter Estimation and Forecasting . . . 16
2.6 Log-Linear Models . . . 16
2.6.1 Parameter Estimation and Forecasting . . . 17
2.7 AR Models . . . 17
2.7.1 Parameter Estimation and Forecasting . . . 17
2.7.2 Testing for AR Models . . . 18
2.8 INAR(p) Models . . . 19
2.8.1 Distributional Assumptions . . . 20
2.8.2 Parameter Estimation and Forecasting . . . 20
2.8.3 Testing for INAR(1) Models . . . 21
2.8.4 Difference to AR(p) Models . . . 21

3 Compositional Data Models **23**
3.1 Motivation . . . 23
3.2 Preliminaries . . . 23
3.3 Common Transformations . . . 25
3.4 The VAR Model . . . 27
3.5 $\mathcal{T}$-Spaces . . . 28
3.6 Zero Handling . . . 31
3.6.1 Rounded Zeros . . . 31
3.6.2 Essential Zeros . . . 35

4 Application **37**
4.1 Model Specifications . . . 37
4.1.1 CoDA Specifications . . . 38
4.1.2 INGARCH Specifications . . . 39
4.2 Error Measure . . . 39
4.3 Examples of Model Application . . . 40
4.4 R-Code . . . 45
4.4.1 R-Packages . . . 45
4.4.2 Handbook . . . 45
4.5 Results . . . 54
4.5.1 Model Comparison . . . 55
4.5.2 General Specifications . . . 58
4.5.3 INGARCH Specifications Results . . . 61
4.5.4 CoDA Specifications Results . . . 63
4.5.5 Zero Handling . . . 64
4.5.6 $\mathcal{T}$-spaces . . . 66
4.5.7 One-vs-All Method . . . 69

5 Conclusion **71**

6 Summary **73**

Bibliography **75**

List of Figures **81**

List of Tables **83**

1 Introduction

1.1 Motivation

Multivariate count data is a reoccurring theme in real-world applications. While there are various methods among the classical statistical models to handle such data, there are fewer methods available to handle it in a time series context. Even more so, when there is an excessive amount of zeros or missing values present. In this thesis, we compare various models for such data and compare their predictive power. We test our models on real world data, which was kindly provided to us, and analyse their performance. In the following, we will shortly describe the general framework and objective.

This thesis is part of a bigger project being carried out at the Technical University of Vienna in cooperation with the company Schrankerl GmbH. Schrankerl GmbH operates food vending machines in offices, which are filled with food ranging from appetizers and main course to snacks and beverages. Each week the vending machines, or in the following also called fridges, are being restocked and the number of items sold in the past week is being recorded. In addition, non-sold items are being disposed of which results in monetary losses. The objective is to find a model to predict the amount the company needs to order for the upcoming week, in a bid to minimise the loss.

1.2 Data Description

In this section, we describe the structure of our data, which is essential in choosing the right model. We have several multivariate time series with integer values, with each series representing a vending machine. The dimensions represent the various categories of the food where each item is of one of the four main categories 1,2,3,4 and one of the various subcategories. We mainly analyse the time series on the aggregated level of the main categories; however, the models can also be applied to the subcategories. In this case we have a model for each main category instead of each vending machine. The values for each category represent the number of items sold. For a fridge f denote this

time series with

$$\left\{\boldsymbol{Y}_t : t = 1, \ldots, T_f; \boldsymbol{Y}_t \in \mathbb{N}_0^K\right\}_f, \tag{1.1}$$

where K stands for the number of categories, T_f denotes the total length of the time series and $\mathbb{N}_0^K = \underbrace{\mathbb{N}_0 \times \ldots \times \mathbb{N}_0}_{K-times}$. This means $\boldsymbol{Y}_t = (Y_{1t}, \ldots, Y_{Kt})^T$ with $Y_{kt} \in \mathbb{N}_0, t = 1, \ldots, T_f$ and $k = 1, \ldots, K$. Since we will sometimes not use all of our data but only a fraction of it, we will denote with T the length of the time series used

$$\left\{\boldsymbol{Y}_t : t = 1, \ldots, T; \boldsymbol{Y}_t \in \mathbb{N}_0^K\right\}_f. \tag{1.2}$$

So Equation (1.1) describes the whole time series available, while Equation (1.2) describes the time series used and it holds $T \leq T_f$. In the following we will use Equation (1.2) to indicate that we may only use a fraction of the whole time series. We will dive more into that in Section 4.1.

The data is measured on a weekly basis and hence our points in time are equidistant. One noteworthy feature of our data is the amount of 0 and NA values, which will be dived into in later sections. An additional characteristic of our data is the difference in length for various time series. While for some time series we have 70+ data points, for others we have less than 10. An example view of our data would be:

Fridge ID	Week Date	Main Category	Sub Category	Sold
111	2021-01-18	1	3	6
111	2021-01-18	1	8	7
111	2021-01-25	2	6	4
222	2022-06-06	3	15	1
222	2022-06-06	4	11	0
222	2022-06-13	1	100061	0
222	2022-06-20	2	6	30
222	2022-06-20	2	10	15

Table 1.1: Example data

As mentioned before, we mainly aggregate our data on main category level. This means that we do not differentiate between the subcategories and are only interested in the number of items sold for each main category. Our data in Table 1.1 would then change to Table 1.2:

Fridge ID	Week Date	Main Category	Sold
111	2021-01-18	1	13
111	2021-01-25	2	4
222	2022-06-06	3	1
222	2022-06-06	4	0
222	2022-06-13	1	0
222	2022-06-20	2	45

Table 1.2: Example data aggregated on main category level

1.3 Outlook

The remainder of the thesis is split in the following way. In Chapters 2 and 3, we describe our methodologies used and the reasoning why we are using them. We provide a short literature review about count data time series in Section 2. In these chapters, we also lay the mathematical groundwork for the considered methods. In Chapter 4, we explain the specification and tuning options for our models and also introduce an error measure to evaluate their performance. We show the results on some exemplary time series and then show the results of each tuning parameter. In Section 4.4, we explain the R-functions used and provide a guidebook. In Chapter 5 we summarise our findings and provide a further outlook on the topic.

2 Count Time Series Models

2.1 Motivation

In this section, we introduce the different count time series models. We begin with a short literature review about possible count data models and then provide a motivation on why we decided to focus on our models. The review is mainly based on Liboschik (2016) and Heinen (2003) and a more detailed review can be found in MacDonald and Zucchini (1997). Later, we define the models themselves and list some of their properties.

Since our data can be seen as a discrete time series with count data, we want a model which is able to take these properties into account. Hence, common features of count data, like autocorrelation and overdispersion, should not be neglected and instead be modelled properly.

One common way to deal with count data are Markov chains. In Markov chains, the dependent variable can take on all possible values in the so called state space and the probability of changing states is then modelled as a transition probability. A limitation is the fact that these models become cumbersome if the state space gets too big and the model loses tractability, see Heinen (2003). As an extension to the basic Markov chains models, Hidden Markov chains are proposed by MacDonald and Zucchini (1997). In this case, one assumes that the observations follow a discrete distribution, i.e. the Poisson distribution. However, instead of assuming that the parameter of this distribution is fixed, it is assumed that it follows a Markov chain with finite state space. This makes it possible to account for serial correlation, as well as overdispersion, see MacDonald and Zucchini (1997). But, since there is no generally accepted way to determine the order of the Markov chain, it can cause problems if the data structure does not provide intuitive ways to do it. Another issue is that the number of parameters which needs to be estimated gets big quickly, especially if the order of the model is big, see Heinen (2003).

Other common models for time series data are the ARMA models and their discrete version, the discrete autoregressive moving average (DARMA) models. They can be

defined as a mixture of discrete probability distributions and a suitable chosen marginal probability function, see Biswas and Song (2009). While there have been various applications, for example in Chang, Delleur, and Kavvas (1987), there seem to be difficulties in their estimation, see Heinen (2003).

State space models with conjugated priors are proposed by Harvey and Fernandes (1989). Here, one assumes that the observations are drawn from a Poisson distribution whose mean itself follows a Gamma distribution. The parameters of the Gamma distribution, which are seen as latent variables, are chosen in such a way that the mean is constant, but the variance is increasing. While there are ways proposed by Zeger (1988) to handle overdispersion, their models have the weakness of needing further assumptions to handle zeros while also having more complicated model specifications, see Heinen (2003).

We decide to focus on the class of generalised linear models (GLM) and in particular on the INGARCH(p,q) and log-linear model. For those models, the observations are modelled conditionally on the past and follow a discrete distribution. The conditional mean is then connected with a link function to the past observations and conditional means. A covariate vector can be included in the model to factor in additional, external information. While being easy to use and estimate, they still provide a good amount of flexibility and additionally, a wide array of tools is available for various tests and forecasts, see Liboschik (2016). We also introduce an extension of the INGARCH(p,q) model to multivariate data. However, since to our knowledge there is currently no R-package available to fit these models, we stay with the univariate version. The INGARCH(p,q) and log-linear model will be discussed in detail in Sections 2.2 and 2.6 respectively.

Since our data features many zero values, we also investigate zero-inflated models (ZIM) with the focus on a zero-inflated version of the INGARCH(p,q) model. The structure of this model follows an INGARCH(p,q) model but with a zero-inflated Poisson distribution as the conditional distribution. However, due to a lack of appropriate R-packages, we use a slightly different version of the ZIM introduced by Lambert (1992). This model is basically a generalised linear regression model with a logit link where the data is assumed to follow a zero-inflated Poisson distribution. More details can be found in Section 2.5.

Another popular approach for count time series are the integer-valued autoregressive (INAR) models presented in Section 2.8.1. These models are based on a thinning operator and a parameter α. The dependent variable y_t is modelled as the sum of an error term and the sum of y_{t-1} draws from an integer-valued distribution with mean

α and finite variance. They are attractive since they have a linear-like structure and a similar correlation structure to AR or ARMA models and hence can be seen as a discrete counterpart, see Heinen (2003).

The simple naive random walk, defined in Section 2.4, is the simplest and most basic approach. Since this is the model that is currently used for forecasting, it is ideal as a benchmark. We will use it to compare the performance of the models with the help of a new error measure in Section 4.2.

Since the INGARCH and the INAR model are based on their real-valued counterparts, the GARCH and AR model, we will also provide a short review for them for better comparison and clearness on why we choose the integer-valued versions. However, we will consider neither the GARCH nor the AR model in our analysis.

2.2 INGARCH Model

We base this whole section on Liboschik (2016) and construct the INGARCH(p,q) model like the author. Take again our time series $\left\{\boldsymbol{Y}_t : t = 1, \ldots, T; \boldsymbol{Y}_t \in \mathbb{N}_0^K\right\}_f$ for fridge f and denote the univariate time series for category k with $\{Y_{kt} : t = 1, \ldots, T; Y_{kt} \in \mathbb{N}_0\}_f$ for $k = 1, \ldots, K$. This means $\boldsymbol{Y}_t = (Y_{1t}, \ldots, Y_{Kt})^T$. Denote an r-dimensional time varying covariate vector with $\mathbf{X}_{kt} = (X_{t1}^k, \ldots, X_{tr}^k)^T$. Let the conditional mean be $\lambda_{kt} = \mathbb{E}\left[Y_{kt}|\mathcal{F}_{k,t-1}\right]$ where $\mathcal{F}_{k,t-1}$ is the σ-field generated by Y_{kt} and λ_l for $l < t$, $\mathcal{F}_{k,t-1} = \sigma(Y_{k1}, \ldots, Y_{kl}, \lambda_1, \ldots, \lambda_l)$. Therefore, the conditional mean of the time series is dependent on its combined history of the past conditional means and its past values. With this, we can define the integer-valued generalized autoregressive conditional heteroskedasticity model of order (p,q) (INGARCH(p,q) model) for category $k = 1, \ldots, K$ as

$$\begin{aligned} Y_{kt}|\mathcal{F}_{k,t-1} &\sim P(\lambda_{kt}); \forall t \in \mathbb{N}, \\ \mathbb{E}\left[Y_{kt}|\mathcal{F}_{k,t-1}\right] = \lambda_{kt} &= \beta_0 + \sum_{i=1}^{p} \beta_i Y_{k,t-i} + \sum_{j=1}^{q} \alpha_j \lambda_{k,t-j}, \end{aligned} \tag{2.1}$$

where $p, q \in \mathbb{N}$ and $P(\lambda_{kt})$ is a Poisson distribution with mean λ_{kt}. The integer p defines the number of past values to regress on, whereas q does the same for the past conditional means. In order to account for external effects as well, we can add the covariate vector $\mathbf{X}_{kt}$

$$
\begin{aligned}
Y_{kt}|\mathcal{F}_{k,t-1} &\sim P(\lambda_{kt}); \forall t \in \mathbb{N}, \\
\mathbb{E}\left[Y_{kt}|\mathcal{F}_{k,t-1}\right] = \lambda_{kt} &= \beta_0 + \sum_{i=1}^{p} \beta_i Y_{k,t-i} + \sum_{j=1}^{q} \alpha_j \lambda_{k,t-j} + \boldsymbol{\eta}^T \mathbf{X}_{kt},
\end{aligned} \tag{2.2}
$$

where $\boldsymbol{\eta}$ is the parameter for the covariates such that $\boldsymbol{\eta}^T \mathbf{X}_{kt} \geq 0$. In this case $\mathcal{F}_{k,t}$ also includes the past information from $\mathbf{X}_{kt}$ up to time $t+1$. From the distributional assumption $Y_{kt}|\mathcal{F}_{k,t-1} \sim P(\lambda_{kt})$ it follows

$$
p_{kt}(y; \boldsymbol{\theta}) = \mathbb{P}(Y_{kt} = y|\mathcal{F}_{k,t-1}) = \frac{\lambda_{kt}^y \exp(-\lambda_{kt})}{y!}, \quad y \in \mathbb{N}_0. \tag{2.3}
$$

Furthermore, it can be shown that conditionally on the past history $\mathcal{F}_{k,t-1}$, the model is equidispersed, i.e. it holds $\lambda_{kt} = \mathbb{E}\left[Y_{kt}|\mathcal{F}_{k,t-1}\right] = \mathbb{V}\left[Y_{kt}|\mathcal{F}_{k,t-1}\right]$. However, unconditionally the model exhibits overdispersion. In that case it holds $\mathbb{E}\left[Y_{kt}\right] \leq \mathbb{V}\left[Y_{kt}\right]$, see Liboschik (2016), Fokianos and Tjøstheim (2011), and Heinen (2003).

Parameter Estimation and Forecasting

We summarise the estimation of the INGARCH(p,q) model as described in Liboschik (2016). The model is estimated for each category $k = 1, \ldots, K$ separately.

The parameter space for the INGARCH(p,q) model with external effects, Model (2.2), is given by

$$
\Theta = \left\{ \boldsymbol{\theta} \in \mathbb{R}^{p+q+r+1} : \beta_0 > 0; \beta_1, \ldots, \beta_p, \alpha_1, \ldots, \alpha_q, \eta_1, \ldots, \eta_r \geq 0; \sum_{i=1}^{p} \beta_i + \sum_{j=1}^{q} \alpha_j < 1 \right\}. \tag{2.4}
$$

To ensure positivity of the conditional mean λ_{kt}, the intercept β_0 must be positive while all other parameters must be non-negative. The upper bound of the sum ensures that the model has a stationary and ergodic solution with moments of any order, see Ferland, Latour, and Oraichi (2006), Fokianos, Rahbek, and Tjøstheim (2009), and Doukhan, Fokianos, and Tjøstheim (2012). A quasi maximum likelihood approach is used to estimate the parameters $\boldsymbol{\theta}$. For observations $\mathbf{y}_k = (y_{k1}, \ldots, y_{kT})^T$ for category $k = 1, \ldots, K$, the conditional quasi log-likelihood function, up to a constant, is given by

$$
\ell_k(\boldsymbol{\theta}) = \sum_{t=1}^{T} \log p_{kt}(y_{kt}; \boldsymbol{\theta}) = \sum_{t=1}^{T} \left(y_{kt} \log(\lambda_{kt}(\boldsymbol{\theta})) - \lambda_{kt}(\boldsymbol{\theta}) \right), \tag{2.5}
$$

where $p_{kt}(y_{kt}; \boldsymbol{\theta})$ is the probability density function defined in Equation (2.3). The

conditional mean is seen as a function $\lambda_{kt} : \Theta \to \mathbb{R}^+$. The conditional score function is given by

$$S_{kT}(\boldsymbol{\theta}) = \frac{\partial \ell_k(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \sum_{t=1}^{T} \left(\frac{y_{kt}}{\lambda_{kt}(\boldsymbol{\theta})} - 1 \right) \frac{\partial \lambda_{kt}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}. \tag{2.6}$$

The vector $\frac{\partial \lambda_{kt}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}$ can be computed recursively. The conditional information matrix is given by

$$\begin{aligned} G_{kT}(\boldsymbol{\theta}; \sigma_k^2) &= \sum_{t=1}^{T} Cov \left(\frac{\partial \ell_k(\boldsymbol{\theta}; Y_{kt})}{\partial \boldsymbol{\theta}} \middle| \mathcal{F}_{k,t-1} \right) \\ &= \sum_{t=1}^{T} \left(\frac{1}{\lambda_{kt}(\boldsymbol{\theta})} + \sigma_k^2 \right) \left(\frac{\partial \lambda_{kt}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right) \left(\frac{\partial \lambda_{kt}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right)^T. \end{aligned} \tag{2.7}$$

In the case of the Poisson distribution, we have $\sigma_k^2 = 0$. If the Negative Binomial distribution is used, see Section 4.1.2, then we have $\sigma_k^2 = \frac{1}{\phi_k}$ where ϕ_k is the dispersion parameter of the Negative Binomial distribution. Finally, assuming that the quasi maximum likelihood estimator (QMLE) $\hat{\boldsymbol{\theta}}_T$ of $\boldsymbol{\theta}$ exists, it is the solution to

$$\hat{\boldsymbol{\theta}} = \hat{\boldsymbol{\theta}}_T = \arg\max_{\boldsymbol{\theta} \in \Theta} (\ell_k(\boldsymbol{\theta})). \tag{2.8}$$

The dispersion parameter can be estimated using

$$\sum_{t=1}^{T} \frac{(Y_{kt} - \hat{\lambda}_{kt})^2}{\hat{\lambda}_{kt} + \hat{\lambda}_{kt}^2 / \hat{\phi}_k} = T - (p + q + r + 1), \tag{2.9}$$

where $\hat{\lambda}_{kt} = \lambda_{kt}(\hat{\boldsymbol{\theta}})$ is the fitted value. The variance is then $\hat{\sigma}_k^2 = \frac{1}{\hat{\phi}_k}$, see Christou and Fokianos (2014), Liboschik (2016).

The optimal one-step ahead forecast with regard to the mean squared error is the conditional expectation $\lambda_{k,t+1} = \mathbb{E}[Y_{k,t+1} | \mathcal{F}_{kt}]$. For $h > 1$, the h-step ahead prediction is calculated iteratively with the one-step ahead predictions of $Y_{k,t+1}, Y_{k,t+2}, \ldots$, see Liboschik (2016).

2.2.1 Multivariate INGARCH Model

Since we have multivariate data, we also investigate multivariate versions of the INGARCH model. There have been various approaches in literature to expend the univariate INGARCH model to more dimensions. For example, bivariate models have been proposed by Liu (2012) and extended by Cui and Zhu (2018).

The authors in Fokianos, Støve, et al. (2020) and Fokianos (2021) introduce and review the multivariate INGARCH model on the basis of a data generating process. Since the extension from the univariate Poisson assumption to the multivariate case is quite complex, the authors focus on constructing a joint distribution, such that the components are marginally Poisson distributed but not the joint distribution itself.

Let $\boldsymbol{Y}_t = (Y_{1t}, \ldots, Y_{Kt})^T$, $\boldsymbol{\lambda}_t = \mathbb{E}[\boldsymbol{Y}_t|\mathcal{F}_t]$ where $\boldsymbol{\lambda}_t = (\lambda_{1t}, \ldots, \lambda_{Kt})^T$ and $\mathcal{F}_t$ is the σ-field generated by $\{\boldsymbol{Y}_0, \ldots, \boldsymbol{Y}_t, \boldsymbol{\lambda}_0\}$. Then for each $k = 1, \ldots, K$ we assume

$$\begin{aligned} Y_{kt}|\mathcal{F}_{t-1} &\sim P(\lambda_{kt}), \\ \boldsymbol{\lambda}_t &= \boldsymbol{d} + \boldsymbol{A}\boldsymbol{\lambda}_{t-1} + \boldsymbol{B}\boldsymbol{Y}_{t-1}, \end{aligned} \tag{2.10}$$

where $\boldsymbol{d}$ is a K-dimensional vector and $\boldsymbol{A}, \boldsymbol{B}$ are $K \times K$ matrices. The elements of $\boldsymbol{d}, \boldsymbol{A}, \boldsymbol{B}$ are assumed to be positive such that $\boldsymbol{\lambda}_t > 0$. Therefore, Equation (2.10) implies, that $\{\boldsymbol{Y}_t\}$ is marginally a Poisson process. However, the joint distribution is not necessarily assumed to be a multivariate Poisson distribution. Instead, the joint distribution is constructed using a copula structure. This allows for arbitrary dependence between the components. The joint distribution is constructed with the help of the following process. Take a starting value $\boldsymbol{\lambda_0} = (\lambda_{1,0}, \ldots, \lambda_{K,0})^T$ and then

1. Let $\boldsymbol{U}_l = (U_{1,l}, \ldots, U_{K,l})$ for $l = 1, \ldots, m$ be a sample from a K-dimensional copula $C(u_1, \ldots, u_K)$. Then by definition of a copula, U_{il} follows marginally the uniform distribution on $(0, 1)$ for $i = 1, \ldots, K$ and $l = 1, \ldots, m$.

2. Define the transformation $X_{il} = -\log(\frac{U_{il}}{\lambda_{i,0}})$. Then the marginal distribution of X_{il} is exponential with parameter $\lambda_{i,0}$.

3. For $\boldsymbol{Y}_0 = (Y_{1,0}, \ldots, Y_{K,0})^T$ and m large enough, define $Y_{i,0} = \max\limits_{1 \le j \le m} (\sum_{l=1}^{j} X_{il}) \le 1$. Then $\boldsymbol{Y}_0$ is marginally a set of starting values of a Poisson process with parameter $\boldsymbol{\lambda}_0$.

4. Use Model (2.10) to obtain $\boldsymbol{\lambda_1}$.

5. Go back to step 1 to obtain $\boldsymbol{Y}_1$ and so on.

see Fokianos, Støve, et al. (2020).

This construction of the joint distribution imposes the dependence among the components of the process $\{\boldsymbol{Y}_t\}$. This approach can be extended to other marginal count processes if they can be generated by continuous arrival times, see Fokianos, Støve, et al. (2020).

We can then define the multivariate INGARCH model as

$$\begin{aligned} \boldsymbol{Y}_t &= \boldsymbol{N}_t(\boldsymbol{\lambda}_t), \\ \boldsymbol{\lambda}_t &= \boldsymbol{d} + \boldsymbol{A}\boldsymbol{\lambda}_{t-1} + \boldsymbol{B}\boldsymbol{Y}_{t-1}, \end{aligned} \tag{2.11}$$

where $\{\boldsymbol{N}_t\}$ is a sequence of independent K-variate copula-Poisson processes that counts the number of events in $[0, \lambda_{1t}]\times, \ldots, \times[0, \lambda_{Kt}]$, see Fokianos, Støve, et al. (2020).

Another approach is taken by S. Lee, D. Kim, and B. Kim (2023). Instead of constructing a joint distribution for the multivariate vector $\boldsymbol{Y}_t$, they fit a one-parameter exponential family conditional distribution to each component Y_{kt}

$$p_k(y|\nu) = \exp(\nu y - A_k(\nu))h_k(y), \quad y \in \mathbb{N}_0, \tag{2.12}$$

where A_k and h_k are known functions and ν is the natural parameter. Both A_k and $B_k(\nu) = \frac{dA_k(\nu)}{d\nu}$ are strictly increasing. The multivariate INGARCH model is then given for each $k = 1, \ldots, K$ by

$$\begin{aligned} Y_{kt}|\mathcal{F}_{t-1} &\sim p_k(y|\nu_{kt}), \\ \boldsymbol{\lambda}_t &= \mathbb{E}[\boldsymbol{Y}_t|\mathcal{F}_{t-1}] = f_{\boldsymbol{\theta}}(\boldsymbol{\lambda}_{t-1}, \boldsymbol{Y}_{t-1}), \end{aligned} \tag{2.13}$$

where $\mathcal{F}_{t-1}$ is the σ-field generated by $\{\boldsymbol{Y}_{t-1}, \boldsymbol{Y}_{t-2}, \ldots\}$, $B_k(\nu_{kt}) = \lambda_{kt}$, and $f_{\boldsymbol{\theta}}$ is a non-negative function on $[0, \infty)^K \times \mathbb{N}_0^K$, depending on the parameter $\boldsymbol{\theta} \in \Theta \subset \mathbb{R}^d$ with $d \in \mathbb{N}$. So for each component Y_{kt}, a univariate INGARCH model is fit, but the components are connected by the conditional mean process. A popular choice of $f_{\boldsymbol{\theta}}$ results in a linear relationship. Take a K-dimensional vector $\boldsymbol{W}$ with positive entries and $K \times K$ matrices $\boldsymbol{A}, \boldsymbol{B}$ with non-negative entries satisfying either

$$\sup_{\boldsymbol{\theta}\in\Theta} \left(\sum_{j=1}^{K} (a_{ij} + b_{ij}) \right) < 1, \quad i = 1, \ldots, K, \tag{2.14}$$

for a compact set $\Theta \subseteq R^{K+2K^2}$ and the vectorization of the K-dimensional vector $\boldsymbol{W}$ and the $K \times K$ matrices $\boldsymbol{A}, \boldsymbol{B}$, $\boldsymbol{\theta} = vec(\boldsymbol{W}, \boldsymbol{A}, \boldsymbol{B})$, or

$$\sup_{\boldsymbol{\theta}\in\Theta} \left(\max_{1\le j\le K} (\sum_{i=1}^{K} a_{ij}) + \max_{1\le j\le K} \left(\sum_{i=1}^{K} b_{ij} \right) \right) < 1. \tag{2.15}$$

Then Model (2.13) becomes

$$
\begin{aligned}
Y_{kt}|\mathcal{F}_{t-1} &\sim p_k(y|\nu_{kt}),\\
\boldsymbol{\lambda}_t = \mathbb{E}[\boldsymbol{Y}_t|\mathcal{F}_{t-1}] &= \boldsymbol{W} + \boldsymbol{A}\boldsymbol{\lambda}_{t-1} + \boldsymbol{B}\boldsymbol{Y}_{t-1},
\end{aligned}
\tag{2.16}
$$

see S. Lee, D. Kim, and B. Kim (2023).

2.3 GARCH Models

INGARCH models are structurally derived from the generalised autoregressive conditional heteroscedasticity (GARCH) models, which themselves are generalisations of the autoregressive conditional heteroscedasticity (ARCH) model. ARCH models, which were first developed by Engle, see Engle (1982), in an economic context, model the variance conditional on past values. Since we no longer assume our data to be integer-valued, our time series has the form $\{Y_{kt} : t = 1, \dots T; Y_{kt} \in \mathbb{R}\}_f$. Denote again with $\mathcal{F}_{k,t}$ the information available at time t. Then the ARCH(q) model is given by

$$
\begin{aligned}
Y_{kt}|\mathcal{F}_{k,t-1} &\sim N(0, h_{kt}),\\
h_{kt} &= a_0 + a_1 Y_{k,t-1}^2 + \ldots + a_q Y_{k,t-q}^2,
\end{aligned}
\tag{2.17}
$$

with $a_0 > 0$, $a_i \geq 0, i = 1, \dots, q$, see Bera and Higgins (1993).

The GARCH model generalises this approach by adding the past variances as another source of information. The GARCH(p,q) model for non-negative parameters $a_0 > 0$, $\boldsymbol{a} = (a_1, \dots, a_q)^T \geq 0$ and $\boldsymbol{b} = (b_1, \dots, b_p)^T \geq 0$ with $p, q \in \mathbb{N}$, $p \geq 0, q > 0$ is given by

$$
\begin{aligned}
Y_{kt}|\mathcal{F}_{k,t-1} &\sim N(0, h_{kt}); \forall t \in \mathbb{N},\\
\mathbb{V}[Y_{kt}|\mathcal{F}_{k,t-1}] = h_{kt} &= a_0 + \sum_{i=1}^{q} a_i Y_{k,t-i}^2 + \sum_{j=1}^{p} b_j h_{k,t-j}; \forall t \in \mathbb{N}.
\end{aligned}
\tag{2.18}
$$

Other distributions than the normal distributions can be taken as well, see Bollerslev (1986).

2.3.1 Parameter Estimation and Forecasting

Estimation of the parameters can be done with maximum likelihood and an iterative algorithm. First, the model is rewritten and the logarithm of the likelihood function is taken. Second, after differentiation with respect to its variance and mean parameters, the Berndt, Hall Hall and Hausman algorithm, see Berndt et al. (1974), is used to

obtain the maximum likelihood estimates, see Bollerslev (1986). Further details and assumptions can be found in Bollerslev (1986).

If one is interested in forecasting $\{Y_{kt}\}$, then the minimum mean squared one-step error forecast is $\mathbb{E}[Y_{k,t+1}|\mathcal{F}_{k,t}] = 0$ where $\mathcal{F}_{k,t}$ is the information available at time t. One should note, that the forecast is independent of the model parameters. If the conditional variance should be forecasted, the parameters are estimated and the known values are plugged in. For $h > 1$, h-step predictions are computed recursively with plugging in the forecasts for $h-1, h-2, \ldots$ in the model, see Zivot (2009).

2.3.2 Testing for GARCH Models

To decide whether to use a GARCH model, one can test for volatility or the validity of GARCH models in general. In the original paper of Bollerslev (1986), the author suggests a Lagrange multiplier test. Other popular tests include the Box–Pierce–Lung-type portmanteau tests and residual-based diagnostics, see Hong and Y.-J. Lee (2017). The authors in Hong and Y.-J. Lee (2017) present further methods.

2.3.3 Applications

The introduction of ARCH and subsequently GARCH models in the 1980s has been revolutionary. ARCH models have originally been introduced for modelling macroeconomic key figures, such as inflation rates, but since then have been used in a variety of fields. GARCH models generalised the ARCH model approach to allow the modelling of a more flexible lag structure. They have found wide applications in finance mathematical problems, especially for the modelling of a changing variance and volatility in financial markets. They are often used to estimate volatility of various financial instruments, see Bollerslev (1986) and Kreiß and Neuhaus (2006).

Since the ARCH and GARCH models are used to model and forecast volatility or the conditional variance, but not values, we will not use it in our application. In addition, the INGARCH model also accounts for the discrete nature of our data, which makes it the preferred choice.

2.4 Naive Random Walk

The Naive Random Walk model is one of the simplest and most comprehensive forecasting models, which makes it a popular benchmark model. In addition, it is what

is currently employed, so using it enables us to directly see if our models outperform the current model. It assumes that the one-step difference between two values is i.i.d distributed with mean 0. We again no longer assume our data to be integer-valued $\{Y_{kt} : t = 1, \dots T; Y_{kt} \in \mathbb{R}_0\}_f$. Then the Naive Random Walk model is given as

$$Y_{k,t+1} = Y_{kt} + \epsilon_{kt}, \tag{2.19}$$

where $\{\epsilon_{kt}\} \sim WN(\sigma_k^2)$ is a white noise process with variance $\sigma_k^2 \in \mathbb{R}_+$. It can be shown easily, that the optimal one-step ahead forecast with regard to the mean squared error (MSE) is given by

$$\hat{Y}_{k,t+1} = Y_{kt}, \tag{2.20}$$

where $\hat{Y}_{k,t+1}$ is the predicted value at time $t+1$. In other words, the predicted value is the last known value, see Nau (2014) and Deistler and Scherrer (2018).

2.5 Zero-Inflated Models

Since we encounter a large number of zeros, we also consider zero-inflated models. Zero inflation means that the proportion of observed zeros is bigger than that of the underlying distribution and hence would not be expected. The idea of zero-inflated models is to add a degenerated distribution with mass at zero to the probability mass function, which enables one to explain the large amount of zero values. The probability mass function of a $ZIP(\lambda, \omega)$ distribution for a random variable Y is defined as

$$\mathbb{P}(Y = y) = \omega\delta_{y,0} + (1-\omega)\frac{\lambda^y \exp(-\lambda)}{y!}, \quad y \in \mathbb{N}_0, \tag{2.21}$$

where $0 < \omega < 1$ is the zero-inflation parameter, λ is the Poisson parameter and $\delta_{y,0}$ is the Kronecker delta for which $\delta_{y,0} = 1$ if $y = 0$ and $\delta_{y,0} = 0$ else. This way our zeros can come from two different sources. The first part of Equation (2.21) $\delta_{y,0}$ is the degenerated point mass distribution, see Zhu (2012).

Let $\mathcal{F}_{k,t-1}$ be the σ-field generated by $\{Y_{k,t-1}, Y_{k,t-2}, \dots\}$. Assume, conditionally on $\mathcal{F}_{k,t-1}$, that $Y_{k,1}, \dots, Y_{k,T}$ are independent. Now we can define the zero-inflated Poisson (ZIP) INGARCH(p,q) as

$$
\begin{aligned}
Y_{kt}|\mathcal{F}_{k,t-1} &\sim ZIP(\lambda_{kt}, \omega_k); \forall t \in \mathbb{N}, \\
\mathbb{E}\left[Y_{kt}|\mathcal{F}_{k,t-1}\right] = \lambda_{kt} &= \beta_0 + \sum_{i=1}^{p} \beta_i Y_{k,t-i} + \sum_{j=1}^{q} \alpha_j \lambda_{k,t-j},
\end{aligned}
\tag{2.22}
$$

with $0 < \omega_k < 1$, $\beta_0 > 0$, $\beta_i \geq 0$, $\alpha_j \geq 0$ for $i = 1, \ldots, p$, $j = 1, \ldots, q$, $p \geq 1$, $q \geq 0$. If $\omega_k = 0$ then we get the standard INGARCH(p,q) model discussed above. It can be shown that the conditional mean and variance are given by

$$
\mathbb{E}[Y_{kt}|\mathcal{F}_{k,t-1}] = (1-\omega_k)\lambda_{kt}, \qquad \mathbb{V}[Y_{kt}|\mathcal{F}_{k,t-1}] = (1-\omega_k)\lambda_{kt}(1+\omega_k\lambda_{kt}), \tag{2.23}
$$

which implies $\mathbb{V}[Y_{kt}|\mathcal{F}_{k,t-1}] > \mathbb{E}[Y_{kt}|\mathcal{F}_{k,t-1}]$. This means that Model (2.22) can handle overdispersion in our data, see Zhu (2012). More details about zero-inflated models and especially the zero-inflated INGARCH(p,q) model can be found in Zhu (2012).

However, due to a lack of available R-packages for zero-inflated Poisson INGARCH models, we use a zero-inflated Poisson autoregressive model. We assume that we have discrete count data $\{Y_{kt}\}$ which is conditionally $ZIP(\lambda_{kt}, \omega_{kt})$ distributed. For the parameters λ_{kt} and ω_{kt}, the ZIP autoregressive model is given by

$$
\begin{aligned}
Y_{kt}|\mathcal{F}_{k,t-1} &\sim ZIP(\lambda_{kt}, \omega_{kt}); \forall t \in \mathbb{N}, \\
\log(\lambda_{kt}) &= \sum_{i=1}^{p} \beta_i b_{t-1,i}^k, \\
\log\left(\frac{\omega_{kt}}{1-\omega_{kt}}\right) &= \sum_{i=1}^{q} \gamma_i z_{t-1,i}^k,
\end{aligned}
\tag{2.24}
$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^T$ and $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_q)^T$ are the parameters to be estimated and the vectors $\boldsymbol{B}_t^k = (b_{ti}^k)$ and $\boldsymbol{Z}_t^k = (z_{ti}^k)$ are the explanatory covariates. In Model (2.24) a logit link function has been used although, it can be replaced with other link functions like the probit or log link, see Yang (2012).

In our case we regress on the past values of our time series. In that case Model (2.24) becomes

$$
\begin{aligned}
\log(\lambda_{kt}) &= \beta_1 + \beta_2 Y_{k,t-1}, \\
\log\left(\frac{\omega_{kt}}{1-\omega_{kt}}\right) &= 1 \cdot \gamma.
\end{aligned}
\tag{2.25}
$$

2.5.1 Parameter Estimation and Forecasting

Parameter estimation for Model (2.25) is done with the maximum partial likelihood estimate. However, since there exists no closed form solution, iterative algorithms like the Expectation-Maximisation (EM), Newton-Raphson (NR), or Fisher Scoring (FS) can be used, see Yang (2012). Further details can be found in Yang (2012).

The one-step ahead predictor is again given by $\mathbb{E}[Y_{kt}|\mathcal{F}_{k,t-1}] = (1-\omega_{kt})\lambda_{kt}$ with the estimated coefficients plugged in.

2.6 Log-Linear Models

As mentioned in Section 2.1, we also investigate log-linear models. These models are structurally very similar to the normal INGARCH(p,q) model, only with a logarithmic link function. Under the same assumptions as for Model (2.1), they have the form

$$\begin{aligned} Y_{kt}|\mathcal{F}_{k,t-1} &\sim P(\lambda_{kt}); \forall t \in \mathbb{N}, \\ \nu_{kt} = \log(\lambda_{kt}) &= \beta_0 + \sum_{i=1}^{p} \beta_i \log(Y_{k,t-i}+1) + \sum_{j=1}^{q} \alpha_j \nu_{k,t-j}. \end{aligned} \tag{2.26}$$

The past values get transformed by $h(x) = \log(x+1)$ to get them on the same scale as ν_{kt} and avoid zero values in the logarithm, see Liboschik (2016) and Fokianos and Tjøstheim (2011).

We consider the log-linear model because it provides solutions to at least two drawbacks from the INGARCH(p,q) model. First, Model (2.1) only allows positive serial correlation. Second, when we include covariates, they can only have a non-negative regression term because otherwise the mean λ_{kt} can become negative. However, in the log-linear case we can extend this to

$$\begin{aligned} Y_{kt}|\mathcal{F}_{k,t-1} &\sim P(\lambda_{kt}); \forall t \in \mathbb{N}, \\ \nu_{kt} = \log(\lambda_{kt}) &= \beta_0 + \sum_{i=1}^{p} \beta_i \log(Y_{k,t-i}+1) + \sum_{j=1}^{q} \alpha_j \nu_{k,t-j} + \boldsymbol{\eta}^T \mathbf{X}_{kt}. \end{aligned} \tag{2.27}$$

with $\boldsymbol{\eta} \in \mathbb{R}^r$. Additionally, because of the updated definition of the parameter space, see Equation (2.28), it also allows for negative autocorrelation, see Liboschik (2016) and Fokianos and Tjøstheim (2011).

2.6.1 Parameter Estimation and Forecasting

The parameter estimation for the log-linear model is done similarly to the INGARCH model in Section 2.2. Only the parameter space Θ is different

$$\Theta = \left\{ \boldsymbol{\theta} \in \mathbb{R}^{p+q+r+1} : |\beta_1|, \ldots, |\beta_p|, |\alpha_1|, \ldots, |\alpha_q| < 1, |\sum_{i=1}^{p} \beta_i + \sum_{j=1}^{q} \alpha_j| < 1 \right\}. \tag{2.28}$$

Just like parameter estimation, forecasting is also performed in the same way as the INGARCH model. The optimal one-step ahead prediction with regards to the mean squared error is given by the conditional expectation $\lambda_{k,t+1} = \mathbb{E}\left[Y_{k,t+1} | \mathcal{F}_{kt}\right]$. The h-step ahead predictions for $h > 1$ are calculated iteratively again, see Liboschik (2016).

Log-Linear models are further discussed in Fokianos and Tjøstheim (2011), Woodard, Matteson, and Henderson (2011), and Douc, Doukhan, and Moulines (2013).

2.7 AR Models

Autoregressive models of order p (AR(p)) are one of the most simple time series models, which makes them very popular. Assuming again our time series is real-valued $\{Y_{kt} : t = 1, \ldots T; Y_{kt} \in \mathbb{R}\}_f$, taking coefficients $\boldsymbol{a} = (a_1, \ldots, a_p)^T \in \mathbb{R}^p$, and a white noise process $\{\epsilon_{kt}\} \sim WN(\sigma_k^2)$, called innovations, they are defined as

$$Y_{kt} = a_1 Y_{k,t-1} + \ldots + a_p Y_{k,t-p} + \epsilon_{kt}. \tag{2.29}$$

If $\{Y_{kt}\}$ is stationary, it is an autoregressive process of order p (AR(p) process). The multivariate version for $\left\{ \boldsymbol{Y}_t : t = 1, \ldots, T; \boldsymbol{Y}_t \in \mathbb{R}_0^K \right\}_f$, also called vectorised autoregressive model (VAR(p)), is defined as

$$\boldsymbol{Y}_t = \boldsymbol{a_1 Y_{t-1}} + \ldots + \boldsymbol{a_p Y_{t-p}} + \boldsymbol{\epsilon}_t, \tag{2.30}$$

where $\boldsymbol{a}_j \in \mathbb{R}^{K \times K}$ are the coefficient matrices and $\{\boldsymbol{\epsilon}_t\} \sim WN(\boldsymbol{\Sigma})$ is a multivariate white noise process with covariance matrix $\boldsymbol{\Sigma}$, see Deistler and Scherrer (2018).

2.7.1 Parameter Estimation and Forecasting

The simplicity of AR models makes parameter estimation and forecasting easy. There are various ways to estimate the parameters in Model (2.30) such as the Yule-Walker

equations, the ordinary least squares (OLS) estimator, and if the innovations $\{\boldsymbol{\epsilon}_t\}$ are multivariate normal distributed, then the maximum likelihood estimator can be used as well, see Deistler and Scherrer (2018) and Hamilton (1994).

Like parameter estimation, forecasting is also straightforward in the AR model. We use the mean squared error as an error measure and only consider affine forecasts with $m \geq p, m \in \mathbb{N}$ past values, hence the forecast has the form

$$\hat{\boldsymbol{Y}}_{t+h} = \boldsymbol{c}_1 \boldsymbol{Y}_t + \ldots + \boldsymbol{c}_m \boldsymbol{Y}_{t-m+1}, \tag{2.31}$$

with coefficients $\boldsymbol{c}_i \in \mathbb{R}^{K \times K}$. Under some additional assumptions for $\{\boldsymbol{Y}_t\}$, which can be found in chapter 5.1 and 5.2 of Deistler and Scherrer (2018), it turns out that the optimal one-step ahead prediction is simply

$$\hat{\boldsymbol{Y}}_{t+1} = \boldsymbol{a}_1 \boldsymbol{Y}_t + \ldots + \boldsymbol{a}_p \boldsymbol{Y}_{t-p+1}. \tag{2.32}$$

So the optimal one-step ahead forecast consists exactly of the coefficients of the AR model. For $h > 1$, one simply continues recursively, using $\hat{\boldsymbol{Y}}_{t+1}$, $\hat{\boldsymbol{Y}}_{t+2}, \ldots$. Since we normally do not know the exact coefficients $\boldsymbol{a}_i$, their estimations $\hat{\boldsymbol{a}}_i$ can be plugged into Equation (2.32), see Deistler and Scherrer (2018).

2.7.2 Testing for AR Models

To test whether or not a time series follows an AR(p) process, the estimates of the white noise process $\{\hat{\boldsymbol{\epsilon}}_t\}$ can be used. These estimates should follow a white noise process and hence should show no signs of serial correlation. Popular tests are the Portmanteau and the Breusch-Godfrey Test, see Lütkepohl (2007).

The Portmanteau Test tests the null hypothesis $H_0 : \mathbb{E}[\boldsymbol{\epsilon}_t \boldsymbol{\epsilon}_{t-m}^T] = \boldsymbol{0}$, i.e. if the estimated innovations are uncorrelated. Under the assumption that $\{\boldsymbol{Y}_t\}$ is an AR(p) process and an AR(p) model has been fit, the used test statistic converges against a chi-squared distribution, see Lütkepohl (2007).

The Breusch-Godfrey Test tests if the coefficients $(d_1, \ldots, d_h)$ in the model

$$\boldsymbol{\epsilon}_t = d_1 \boldsymbol{\epsilon}_{t-1} + \ldots d_h \boldsymbol{\epsilon}_{t-h} + \boldsymbol{\eta}_t, \tag{2.33}$$

are zero. The process $\{\boldsymbol{\eta}_t\}$ is a white noise process and hence we test if $\{\boldsymbol{\epsilon}_t\}$ follows an AR(h) structure or not. Under the null hypothesis the test statistic follows a chi-squared distribution again, see Lütkepohl (2007).

2.8 INAR(p) Models

Integer-valued autoregressive models of order p (INAR(p)) are another option to handle univariate count data. To define them, we first need to define the generalised thinning operator. Take an integer-valued, non-negative random variable X and $\alpha \in [0, 1]$. Further, take a sequence of i.i.d. integer-valued, non-negative random variables $(Z_i)_{i=1}^{X}$ with finite mean α and variance $\sigma^2 < \infty$ which are independent of X. Then the generalised thinning operator $\circ$ is defined as

$$\alpha \circ X = \sum_{i=1}^{X} Z_i. \tag{2.34}$$

The sequence $\{Z_i\}_{i=1}^{X}$ is called the counting series of X, see I. Silva et al. (2005).

We can then define the INAR(p) model for a positive integer-valued time series $\{X_t\}$ as

$$X_t = \alpha_1 \circ X_{t-1} + \alpha_2 \circ X_{t-2} + \ldots + \alpha_p X_{t-p} + \epsilon_t, \tag{2.35}$$

where

1. $\{\epsilon_t\}$ is a sequence of integer-valued i.i.d. random variables, called innovations, with finite first, second, and third moment,
2. $\alpha_i \circ X_{t-i} = \sum_{j=1}^{X_{t-i}} Z_{i,j}$ for $i = 1, \ldots, p$ and for each i the counting series $\{Z_{i,j}\}$ with $j = 1, \ldots, X_{t-i}$ are mutually independent, independent of $\{\epsilon_t\}$, and it holds $\mathbb{E}[Z_{i,j}] = \alpha_i$, as well as $\mathbb{V}[Z_{i,j}] = \sigma_i^2$ and $\mathbb{E}[Z_{i,j}^3] = \gamma_i$,
3. $\alpha_i \in (0, 1]$ for $i = 1, \ldots, p-1$ and $0 < \alpha_p < 1$,
4. $\sum_{j=1}^{p} \alpha_j < 1$.

The last condition ensures the existence and stationary of the process, see I. Silva et al. (2005).

Let $\{Y_{kt} : t = 1, \ldots T; Y_{kt} \in \mathbb{N}_0\}_f$ be again the univariate integer-valued time series for category k for $k = 1, \ldots, K$ and fridge f. Then the INAR(p) model is given by

$$Y_{kt} = \alpha_{k1} \circ Y_{k,t-1} + \alpha_{k2} \circ Y_{k,t-2} + \ldots + \alpha_{kp} \circ Y_{k,t-p} + \epsilon_{kt}. \tag{2.36}$$

For simplicity, we will consider INAR(1) models, although the optimal choice of the lag is something that could be further investigated.

2.8.1 Distributional Assumptions

While we will mainly assume that the innovations $\{\epsilon_t\}$ follow a Poisson distribution, they can also follow other distributions. One interesting option is, that one can choose a zero-inflated distribution, see Garay et al. (2022). This could make the model adequate for our data.

2.8.2 Parameter Estimation and Forecasting

Parameter estimation can be done in several ways. Possible methods are: moment based estimators (MM), regression based or conditional least squares (CLS), and maximum likelihood (ML) based estimators. Especially for the Poisson model, those methods have been studied in detail in literature, see I. Silva et al. (2005) for more details.

The authors in I. Silva et al. (2005) present two types of forecasting methods for INAR(1) models. The first approach is a classical method for performing predictions in a time series context and makes use of the conditional expectation. It was obtained by Brännäs (1993) and Freeland and McCabe (2004). Assuming that $\{\epsilon_{kt}\} \sim_{i.i.d} P(\lambda_k)$ is Poisson distributed with parameter λ_k, the h-step ahead predictor, for $h \in \mathbb{N}$, based on n past observations $\boldsymbol{Y}_k = (Y_{k1}, \ldots, Y_{kn})$ is given by

$$\hat{Y}_{k,n+h} = \mathbb{E}[Y_{k,n+h}|\boldsymbol{Y}_k] = \alpha_k^h \left[Y_{kn} - \frac{\lambda_k}{1-\alpha_k} \right] + \frac{\lambda_k}{1-\alpha_k}. \tag{2.37}$$

However, this forecast hardly ever produces integer values. One option to counter this problem, is to take the value which minimises $\mathbb{E}[|Y_{k,n+h} - \hat{Y}_{k,n+h}||Y_{k,n}]$, the absolute expected error, instead of the MSE. This turns out to be the median $\hat{m}_{n+h}$ of the h-step ahead conditional distribution of $Y_{k,n+h}|Y_{k,n}$. Another option is a Bayesian approach presented in I. Silva et al. (2005). It is based on the assumption that both, the future prediction $Y_{k,n+h}$ and the vector of unknown parameters $\boldsymbol{\theta} = (\alpha_k, \lambda_k)$ are random. Since the complexity of the posterior probability density function makes it difficult to work with it directly, a sampling algorithm can be deployed for estimation. The details are given in I. Silva et al. (2005). The estimator for the conditional expectation is then given by

$$\hat{Y}_{k,n+h} = Y_{kn} \left(\frac{1}{m} \sum_{i=1}^{m} \alpha_{ki}^h \right) + \left(\frac{1}{m} \sum_{i=1}^{m} \frac{1-\alpha_{ki}^h}{1-\alpha_{ki}} \lambda_{ki} \right), \tag{2.38}$$

where m is the sampling size and the pairs $(\alpha_{ki}, \lambda_{ki})$ for $i = 1, \ldots, m$ and a fixed k with $k = 1, \ldots, K$ are the sampled parameters, see I. Silva et al. (2005) and Freeland and

McCabe (2004).

2.8.3 Testing for INAR(1) Models

To test the adequacy of the INAR(1) model, there are again various options.

Parametric re-sampling is a popular method. The idea is to generate data with the help of the fitted model, construct the empirical distribution of the functional of interest and check if the original sample is a reasonable point within that empirical distribution, see I. Silva et al. (2005).

Residual based methods are based on the Pearson residuals defined by

$$r_{kt} = \frac{Y_{kt} - \mathbb{E}[Y_{kt}|Y_{k,t-1}]}{\mathbb{V}[Y_{kt}|Y_{k,t-1}]^{1/2}}, \tag{2.39}$$

where estimated quantities are plugged in. If the model is specified correctly, the residuals should have mean zero, variance one and no significant serial correlation, see I. Silva et al. (2005).

Another option is based on predictive distributions where an adjusted probability integral transform (PIT) is used. Further details can be found in I. Silva et al. (2005).

2.8.4 Difference to AR(p) Models

Depending on the definition of the INAR(p) model, the degree of similarity varies. If one follows the definition of Jin-Guan and Yuan (1991), which is the one given in Equation (2.35), then the autocorrelation function follows that of an AR(p) process, see M. Silva and Oliveira (2005). However, the authors in Alzaid and Al-Osh (1990) propose a different definition. In their work, given $Y_{tk} = y_{tk}$, the conditional distribution of $(\alpha_1 \circ Y_{tk}, \ldots, \alpha_p \circ Y_{tk})$ is multinomial with parameters $(\alpha_1, \ldots, \alpha_p, y_{tk})$ and is independent of the history of the process. Under those assumptions, the components $\alpha_i \circ Y_{t,k}$ of $Y_{t,k}$ for $i = 1, 2, \ldots, p$ have a stronger mutual dependence structure than the corresponding AR(p) process and induce a moving-average structure, see Alzaid and Al-Osh (1990). Because of this additional dependence, it can be shown that the autocorrelation behaves more like a standard ARMA(p,p-1) process, see Alzaid and Al-Osh (1990).

3 Compositional Data Models

3.1 Motivation

Another way to see our data is as a compositional time series. Compositional data, which is by nature multivariate, describes relations between the parts instead of absolute values. We assume that for each point in time t, our K-dimensional data $(Y_{1t}, \ldots, Y_{Kt})^T$ is part of a $(K-1)$-dimensional simplex defined in Equation (3.1) with total sum κ_t. Hence, the value of each category can be seen as the relative share at the current time. With the use of a transformation, we map the data to the Euclidean vector space, fit a time series model there, predict the relative share of the category for the next point in time and then back transform the result into the original space. Since we are ultimately interested in the absolute value, we also investigate the inclusion of the total sum of all categories as an additional variable and predict it as well. We use the predicted shares and the predicted total value to calculate the absolute values of each part. This is modelled as the so-called $\mathcal{T}$-Space, which will be explained in further detail in Section 3.5. For the actual modelling, we choose VAR models. Their easiness to estimate and interpret, as well as other beneficial properties with our choice of transformation, make them desirable, see Kynčlová, Filzmoser, and Hron (2015).

3.2 Preliminaries

The basis of this section is given by Kynčlová, Filzmoser, and Hron (2015), Egozcue, Pawlowsky-Glahn, et al. (2003) and Filzmoser and Hron (2020).

CoDA, which is short for "Compositional Data Analysis", works with compositional data. The key to compositional data analysis is the fact that only the relative relation of the parts to each other is important. To define compositional data, we first need to

define the $(D-1)$-dimensional simplex

$$\mathbb{S}^D = \left\{ (x_1, \ldots, x_D)^T : x_i > 0, i = 1, \ldots, D; \sum_{i=1}^{D} x_i = \kappa \right\}, \tag{3.1}$$

where κ is a positive constant. The choice of κ is not relevant, as the relative information in the compositional parts stays the same. A D-dimensional vector $\mathbf{x} = (x_1, \ldots, x_D)^T$ is said to be compositional if it is part of $\mathbb{S}^D$. Next, we can induce a $(D-1)$-dimensional vector space on $\mathbb{S}^D$ by perturbation and power transformation. For $\mathbf{x}, \mathbf{z} \in \mathbb{S}^D$ and $b \in \mathbb{R}$ they are defined respectively as

$$\mathbf{x} \oplus_a \mathbf{z} = \mathcal{C}(x_1 z_1, x_2 z_2, \ldots, x_D z_D)^T, \quad b \odot_a \mathbf{x} = \mathcal{C}(x_1^b, x_2^b, \ldots, x_D^b)^T. \tag{3.2}$$

Since we will use these operations to induce a geometry called the Aitchison geometry, we will mark them with the subscript a. Here $\mathcal{C}$ is the closure operation that maps each compositional vector from the real value space $\mathbb{R}_+^D$ into its representation in $\mathbb{S}^D$

$$\mathcal{C}(\boldsymbol{x}) = \left(\frac{\kappa x_1}{\sum_{i=1}^{D} x_i}, \ldots, \frac{\kappa x_D}{\sum_{i=1}^{D} x_i} \right)^T. \tag{3.3}$$

Using $\boldsymbol{z}^{-1} = \mathcal{C}(z_1^{-1}, z_2^{-1}, \ldots, z_D^{-1})$, the inverse perturbation can be defined as

$$\mathbf{x} \ominus_a \mathbf{z} = \mathbf{x} \oplus_a \mathbf{z}^{-1}. \tag{3.4}$$

Now we further define an inner product in order to have an inner product space over the simplex $\mathbb{S}^D$. For two compositions $\mathbf{x}, \mathbf{z} \in \mathbb{S}^D$ define the Aitchison inner product as

$$\langle \mathbf{x}, \mathbf{z} \rangle_a = \frac{1}{2D} \sum_{i=1}^{D} \sum_{j=1}^{D} \log(\frac{x_i}{x_j}) \log(\frac{z_i}{z_j}). \tag{3.5}$$

In addition, a norm and distance measure can be defined

$$\|\mathbf{x}\|_a^2 = \langle \mathbf{x}, \mathbf{x} \rangle_a, \quad d_a(\mathbf{x}, \mathbf{z}) = \|\mathbf{x} \ominus_a \mathbf{z}\|_a. \tag{3.6}$$

With this, we induced the Aitchison geometry and it allows us to express a composition $\mathbf{x} \in \mathbb{S}^D$ as a perturbation-linear combination of a basis of $\mathbb{S}^D$, see Filzmoser and Hron (2020) and Kynčlová, Filzmoser, and Hron (2015).

However, in order to use standard statistical tools, it is desirable to move from this geometry to the Euclidean real space, see Filzmoser and Hron (2020). There are various ways to map the data from the simplex $\mathbb{S}^D$ to the Euclidean real space. A review of the

most common transformations is provided in the following section.

3.3 Common Transformations

Let $\mathbf{x}, \mathbf{z} \in \mathbb{S}^D$ be D-part compositions.

alr Coordinates

The additive log-ratio (alr) Coordinates are defined as

$$\mathbf{z}^{(k)} = alr_k(\mathbf{x}) = \left(\log\left(\frac{x_1}{x_k}\right), \ldots, \log\left(\frac{x_{k-1}}{x_k}\right), \log\left(\frac{x_{k+1}}{x_k}\right), \ldots, \log\left(\frac{x_D}{x_k}\right)\right)^T, \tag{3.7}$$

and map the composition $\mathbf{x}$ to the real space $\mathbb{R}^{D-1}$, see Kynčlová, Filzmoser, and Hron (2015). They are mainly mentioned for historic purposes since they are an intuitive way of transformation. However, limitations are posed by their dependence on the choice of the denominator x_k and the fact that they are not orthogonal to each other, see Filzmoser and Hron (2020).

clr Coefficients

Let $g(\mathbf{x})$ be the geometric mean of $\mathbf{x}$. The centered log-ratio coefficients are then defined as

$$\mathbf{w} = (w_1, \ldots, w_D)^T = clr(\mathbf{x}) = \left(\log\left(\frac{x_1}{g(\mathbf{x})}\right), \ldots, \log\left(\frac{x_D}{g(\mathbf{x})}\right)\right)^T, \tag{3.8}$$

see Kynčlová, Filzmoser, and Hron (2015). This transformation maps $\mathbf{x}$ into the hyperplane $V = \left\{\mathbf{w} \in \mathbb{R}^D : \sum_{i=1}^D w_i = 0\right\} \subset \mathbb{R}^D$. Hence, the transformed data is constrained, which is emphasised by the term 'coefficient' instead of 'coordinates', see Filzmoser and Hron (2020). It can be shown that the *clr* transformation is an isometry, see Egozcue, Pawlowsky-Glahn, et al. (2003). Therefore it holds

$$\langle \mathbf{x}, \mathbf{z} \rangle_a = \langle clr(\mathbf{x}), clr(\mathbf{z}) \rangle, \tag{3.9}$$

$$d_a(\mathbf{x}, \mathbf{z}) = d(clr(\mathbf{x}), clr(\mathbf{z})). \tag{3.10}$$

ilr Coordinates

The isometric log-ratio (ilr) are closely related to the clr coefficients. Assume the inverse clr transformation is isometric. Let $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{D-1}\}$ be an orthonormal base in the hyperplane V. Then $\mathbf{e}_i = clr^{-1}(\boldsymbol{v}_i), i = 1, \ldots, D-1$ is an orthonormal basis in the simplex $\mathbb{S}^D$. For $\mathbf{x} \in \mathbb{S}^D$, the ilr transformation can then be defined as

$$\mathbf{u} = ilr(\mathbf{x}) = \left(\langle \mathbf{x}, \mathbf{e}_1 \rangle_a, \ldots, \langle \mathbf{x}, \mathbf{e}_{D-1} \rangle_a\right)^T. \tag{3.11}$$

In addition to being isometric, the ilr transformation is also isomorph. Let $\mathbf{x}, \mathbf{z}$ be two compositions and $a, b \in \mathbb{R}$. Then

$$ilr(a \odot_a \mathbf{x} \oplus_a b \odot_a \mathbf{z}) = a \cdot ilr(\mathbf{x}) + b \cdot ilr(\mathbf{z}), \tag{3.12}$$

as well as,

$$\langle \mathbf{x}, \mathbf{z} \rangle_a = \langle ilr(\mathbf{x}), ilr(\mathbf{z}) \rangle, \tag{3.13}$$

$$d_a(\mathbf{x}, \mathbf{z}) = d(ilr(\mathbf{x}), ilr(\mathbf{z})), \tag{3.14}$$

$$\|\boldsymbol{x}\|_a = \|ilr(\boldsymbol{x})\|, \tag{3.15}$$

see Kynčlová, Filzmoser, and Hron (2015). From the definition of the ilr coordinates it can be seen, that they can be expressed as a linear combination of the basis induced by the clr coefficients as seen above. Let $\mathbf{V}$ be a $D \times (D-1)$ matrix with columns $\mathbf{v}_i = clr(\mathbf{e}_i)$. For a composition $\mathbf{x}$, the vector of ilr coordinates associated with $\mathbf{V}$ is given by

$$\mathbf{u}_\mathbf{V} = ilr_\mathbf{V}(\mathbf{x}) = \mathbf{V}^T clr(\mathbf{x}) = \mathbf{V}^T \log(\mathbf{x}). \tag{3.16}$$

The matrix $\mathbf{V}$ is the contrast matrix with the orthonormal basis $(\mathbf{e}_i)_{i=1}^{D-1}$, see Egozcue, Pawlowsky-Glahn, et al. (2003). A special choice of orthogonal coordinates leads to the coordinates

$$\begin{gathered} ilr(\mathbf{x}) = (u_1, \ldots, u_{D-1})^T, \\ u_j = \sqrt{\frac{D-j}{D-j+1}} \log\left(\frac{x_j}{\sqrt[D-j]{\prod_{l=j+1}^{D} x_l}}\right), \quad j = 1, \ldots, D-1. \end{gathered} \tag{3.17}$$

With this choice, the problem of interpretation, which arises from the relative nature of the compositional data and the dimension of the simplex, can be solved. The part x_1 is only contained in z_1 and therefore contains all relative information of x_1, see Kynčlová, Filzmoser, and Hron (2015) and Filzmoser and Hron (2020).

Since the *ilr* transformation is defined as a one-to-one mapping, we can transform the data back in the simplex. The inverse transformation is given by

$$x_1 = \exp\left(\sqrt{\frac{D-1}{D}}u_1\right), \tag{3.18}$$

$$x_i = \exp\left(-\sum_{j=1}^{i-1}\frac{1}{\sqrt{(D-j+1)(D-j)}}u_j + \sqrt{\frac{D-i}{D-i+1}}u_i\right), \quad i = 2,\ldots,D-1, \tag{3.19}$$

$$x_D = \exp\left(-\sum_{j=1}^{D-1}\frac{1}{\sqrt{(D-j+1)(D-j)}}u_j\right), \tag{3.20}$$

see Kynčlová, Filzmoser, and Hron (2015).

3.4 The VAR Model

Since we have established the basic setting, we can now introduce compositional time series (CTS). A CTS $\{\boldsymbol{x}_t\}_{t=1}^n$ can be defined as a series where $\boldsymbol{x}_t = (x_{1t},\ldots,x_{Dt})^T \in \mathbb{S}^D$. They are thus characterised by their positive components which sum up to a constant κ_t for each point in time $t = 1,\ldots,n$

$$\sum_{i=1}^{D} x_{it} = \kappa_t, \qquad x_i > 0, i = 1,\ldots D\,; t = 1,\ldots,n. \tag{3.21}$$

Since we do not assume our data to be integer-valued, our time series has the form $\left\{\boldsymbol{Y}_t : t = 1,\ldots,T; \boldsymbol{Y}_t \in \mathbb{R}_0^K\right\}_f$ for fridge f. Further, assume that $\boldsymbol{Y}_t = (Y_{1t},\ldots,Y_{Kt})^T$ is a K-dimensional compositional vector measured at time $t = 1,\ldots,T$ and let $\mathbf{u}_t = ilr_{\boldsymbol{V}}(\boldsymbol{Y}_t)$ be its *ilr* transformation determined by the matrix $\mathbf{V}$. Then the VAR model with lag order p is given by

$$\mathbf{u}_t = \mathbf{c_V} + \mathbf{A}_{\mathbf{V}}^{(1)}\mathbf{u}_{t-1} + \mathbf{A}_{\mathbf{V}}^{(2)}\mathbf{u}_{t-2} + \ldots + \mathbf{A}_{\mathbf{V}}^{(p)}\mathbf{u}_{t-p} + \boldsymbol{\epsilon}_t, \tag{3.22}$$

where $\mathbf{c_V} \in \mathbb{R}^{K-1}$ is a real vector, $\mathbf{A}_{\mathbf{V}}^{(i)} \in \mathbb{R}^{(K-1)\times(K-1)}$ are parameter matrices, and $\{\boldsymbol{\epsilon}_t\}$ is a white noise process with covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}$. The observation $\mathbf{u}_t$ therefore

depends on the p past observations $\mathbf{u}_{t-1}, \ldots, \mathbf{u}_{t-p}$, see Kynčlová, Filzmoser, and Hron (2015).

It can be shown, that two VAR(p) models resulting from different *ilr* transformations are compositionally equivalent, which means that the same predictions are obtained, see Kynčlová, Filzmoser, and Hron (2015).

Estimation and forecasting of Model (3.22) can be done with the methods described in Section 2.7.1.

3.5 $\mathcal{T}$-Spaces

As we have seen, the focus in compositional data analysis lies in the relative information encoded in the observations. However, as is often the case in practice, the absolute information is of interest as well. To retain this information, usually two practices are used. First, for a vector $\boldsymbol{x} \in \mathbb{R}_+^D$ the component wise logarithm $\log(\boldsymbol{x})$ is considered. Second, the total sum, or some other function, of $\boldsymbol{x}$ is added as an additional variable. Here, we will dive deeper into the second method mentioned. For more details see Pawlowsky-Glahn, Egozcue, and Lovell (2013).

Let $\boldsymbol{x} \in \mathbb{R}_+^D$ be a positive vector and $\mathcal{C}(\boldsymbol{x})$ the projection onto $\mathbb{S}^D$. Further, take a function $t : \mathbb{R}_+^D \longrightarrow \mathbb{R}_+$ (i.e. the sum, product,...). Then define the product space $\mathcal{T} = \mathbb{R}_+ \times \mathbb{S}^D$ as the space of all possible elements $(t(\boldsymbol{x}), \mathcal{C}(\boldsymbol{x}))^T$. To define a D-dimensional Euclidean vector space structure on $\mathcal{T}$, we define an Abelian inner group operation, an external multiplication, and an inner product. However, first we need to induce the Euclidean structure on $\mathbb{R}_+^D$ with the same operations. For $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}_+^D$ and $\alpha \in \mathbb{R}$ define the Abelian inner group operation, the external multiplication, and an inner product respectively as

$$\boldsymbol{x} \oplus_+ \boldsymbol{y} = (x_1 \cdot y_1, \ldots, x_D \cdot y_D)^T, \tag{3.23}$$

$$\alpha \odot_+ \boldsymbol{x} = (x_1^\alpha, \ldots, x_D^\alpha)^T, \tag{3.24}$$

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle_+ = \langle \log(\boldsymbol{x}), \log(\boldsymbol{y}) \rangle . \tag{3.25}$$

Here, $\langle , \rangle$ denotes the usual Euclidean inner product on $\mathbb{R}^D$, see Pawlowsky-Glahn, Egozcue, and Lovell (2013).

Now we can define for $\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}} \in \mathcal{T}$ and $\alpha \in \mathbb{R}$ the Abelian inner group operation as

$$\tilde{\boldsymbol{x}} \oplus_T \tilde{\boldsymbol{y}} = (t(\boldsymbol{x}) \oplus_+ t(\boldsymbol{y}), \boldsymbol{x} \oplus_a \boldsymbol{y})^T = (t(\boldsymbol{x}) \cdot t(\boldsymbol{y}), \mathcal{C}(\tilde{x}_1\tilde{y}_1, \ldots, \tilde{x}_D\tilde{y}_D))^T, \tag{3.26}$$

and the external multiplication as

$$\alpha \odot_T \tilde{\boldsymbol{x}} = (\alpha \odot_+ t(\boldsymbol{x}), \alpha \odot_a \boldsymbol{x})^T = (t(\boldsymbol{x})^\alpha, \mathcal{C}(\tilde{x}_1^\alpha, \ldots, \tilde{x}_D^\alpha))^T, \tag{3.27}$$

where $\oplus_a$ and $\odot_a$ are the perturbation and power transformation defined in Equation (3.2) and $\oplus_+$ and $\odot_+$ the respective operations defined for $\mathbb{R}_+$ (3.23)(3.24), see Pawlowsky-Glahn, Egozcue, and Lovell (2013).

The inner product is defined as

$$\langle \tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}} \rangle_T = \langle t(\boldsymbol{x}), t(\boldsymbol{y}) \rangle_+ + \langle \mathcal{C}(\boldsymbol{x}), \mathcal{C}(\boldsymbol{y}) \rangle_a , \tag{3.28}$$

where $\langle , \rangle_+$ is the inner product in $\mathbb{R}_+$ and $\langle , \rangle_a$ is the Aitchison inner product defined in (3.5), see Pawlowsky-Glahn, Egozcue, and Lovell (2013).

Further we can define a distance on $\mathcal{T}$ with

$$d_T^2(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}) = d_+^2(t(\boldsymbol{x}), t(\boldsymbol{y})) + d_a^2(\mathcal{C}(\boldsymbol{x}), \mathcal{C}(\boldsymbol{y})), \tag{3.29}$$

with $d_+^2(\boldsymbol{x}, \boldsymbol{y}) = d(\log(\boldsymbol{x}), \log(\boldsymbol{y}))$ and d is the Euclidean distance, see Pawlowsky-Glahn, Egozcue, and Lovell (2013).

To ensure that the operations performed on $\mathcal{C}(\boldsymbol{x})$ are compatible with the ones performed on $\mathcal{T}$ we need to impose some conditions on the function $h : \mathbb{R}_+^D \to \mathcal{T}$, $h(\boldsymbol{x}) = (t(\boldsymbol{x}), \mathcal{C}(\boldsymbol{x}))^T$. First, the function h needs to be a one-to-one function since otherwise information could be lost by applying h or h^{-1}. Since we can write $\boldsymbol{x} \in \mathbb{R}_+^D$ as $\boldsymbol{x} = \frac{\sum_{i=1}^D x_i}{\kappa} \cdot \mathcal{C}(\boldsymbol{x})$, the function t must be related to the sum of the components. This allows the reconstruction of $\boldsymbol{x}$ from the composition and total. To see this, write $\frac{\sum_{i=1}^D x_i}{\kappa} \cdot \mathcal{C}(\boldsymbol{x}) = h^{-1}((t(\boldsymbol{x}), \mathcal{C}(\boldsymbol{x}))^T)$, see Pawlowsky-Glahn, Egozcue, and Lovell (2013).

The second condition is the preservation of the vector space properties in $\mathbb{R}_+^D$ and $\mathcal{T}$

$$h(\boldsymbol{x} \oplus_+ \boldsymbol{y}) = h(\boldsymbol{x}) \oplus_T h(\boldsymbol{y}), \tag{3.30}$$

$$h(\alpha \odot_+ \boldsymbol{x}) = \alpha \odot_T h(\boldsymbol{x}). \tag{3.31}$$

This means for the function t that

$$t(\boldsymbol{x} \oplus_+ \boldsymbol{y}) = t(\boldsymbol{x}) \cdot t(\boldsymbol{y}), \tag{3.32}$$

$$t(\alpha \odot_+ \boldsymbol{x}) = (t(\boldsymbol{x}))^{\alpha}. \tag{3.33}$$

In Pawlowsky-Glahn, Egozcue, and Lovell (2013) it is shown that $h_s = ((t_s(\boldsymbol{x}), \mathcal{C}(\boldsymbol{x}))^T)$ with $t_s(\boldsymbol{x}) = \sum_{i=1}^{D} x_i$ is a one-to-one function, but not compatible with $\oplus_+, \odot_+$ and $\oplus_T, \odot_T$. However, as h_s is a one-to-one function between $\mathbb{R}_+^D$ and $\mathcal{T}$, there exists a Euclidean structure in $\mathbb{R}_+^D$ that is isometric to the one in $\mathcal{T}$. The vector space operations can be defined as

$$\boldsymbol{x} \oplus_{+s} \boldsymbol{y} = h_s^{-1}(\tilde{\boldsymbol{x}}) \oplus_T h_s^{-1}(\tilde{\boldsymbol{y}}), \tag{3.34}$$

$$\alpha \odot_{+s} \boldsymbol{x} = \alpha \odot_T h_s^{-1}(\tilde{\boldsymbol{x}}), \tag{3.35}$$

$$d_{+s}^2(\boldsymbol{x}, \boldsymbol{y}) = d_T^2(h_s(\boldsymbol{x}, \boldsymbol{y})), \tag{3.36}$$

where $\oplus_{+s}$ and $\odot_{+s}$ are the new operations in $\mathbb{R}_+^D$ that are compatible with the operations in $\mathcal{T}$ and d^2 is the squared distance in $\mathcal{T}$, see Pawlowsky-Glahn, Egozcue, and Lovell (2013).

With the structure established, we can model the relative structure and total sum in one model. We have again $\boldsymbol{Y}_t = (Y_{1t}, \ldots, Y_{Kt})^T$ and hence $\mathcal{T} = \mathbb{R}_+ \times \mathbb{S}^K$. So $\tilde{\boldsymbol{Y}}_t = h(\boldsymbol{Y}_t) = (t(\boldsymbol{Y}_t), \mathcal{C}(\boldsymbol{Y}_t))^T$ with $t(\boldsymbol{Y}_t) = \sum_{k=1}^{K} Y_{kt}$. For $\boldsymbol{w}_t = (t(\boldsymbol{Y}_t), ilr_{\boldsymbol{V}}(\boldsymbol{Y}_t))^T$ take the *ilr* transformation determined by matrix $\boldsymbol{V}$. Further, let $\boldsymbol{c_V} \in \mathbb{R}^K$ be a real vector, $\mathbf{A}_{\mathbf{V}}^{(i)} \in \mathbb{R}^{K \times K}$ parameter matrices, and $\{\boldsymbol{\epsilon}_t\}$ be a white noise process with covariance matrix $\boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}$. Then we get the model

$$\mathbf{w}_t = \mathbf{c}_{\mathbf{V}} + \mathbf{A}_{\mathbf{V}}^{(1)} \mathbf{w}_{t-1} + \mathbf{A}_{\mathbf{V}}^{(2)} \mathbf{w}_{t-2} + \ldots + \mathbf{A}_{\mathbf{V}}^{(p)} \mathbf{w}_{t-p} + \boldsymbol{\epsilon}_t. \tag{3.37}$$

In our application, we will use $t(\boldsymbol{Y}_t) = \sum_{k=1}^{K} Y_{kt}$ or $t(\boldsymbol{Y}_t) = \log(\sum_{k=1}^{K} Y_{kt})$ since we are interested in the total sum at time t. The logarithmic sum is a popular choice in the time series context as it prevents the sum of being too big, see Kynčlová, Filzmoser, and Hron (2015). The estimation of Model (3.37) is carried out analogous to 2.7.1.

3.6 Zero Handling

As we can see in the definition of the simplex (3.1), a compositional vector can only consist of positive parts and since we have a considerable amount of zeros in our data, we need to take care of them. There have been various methods proposed in literature to handle zero values in compositional data but first, a distinction must be made in the type of zeros present. One can differentiate between two types of zeros. The first type of zeros is called structural zeros or essential zeros. Those values are truly zero. The second type is called rounded zeros or count zeros. They appear due to imprecision when measuring data or if the detected value is below the detection limit. Those values are not truly zero and hence it makes sense to replace them to perform compositional data analysis. In our data we have essential zeros. In the following we summarise the methods presented in Lubbe, Filzmoser, and Templ (2021) and J. A. Martín-Fernández, Barceló-Vidal, and Pawlowsky-Glahn (2003).

3.6.1 Rounded Zeros

Some basic methods for handling rounded zeros are proposed by J. A. Martín-Fernández, Barceló-Vidal, and Pawlowsky-Glahn (2003), while more advanced methods are discussed in Palarea-Albaladejo and J. A. Martín-Fernández (2015). A common feature of most of these methods is the assumption of the existence of a detection limit (DL). Due to technical or other limitations, observations below this limit cannot be detected and hence are missing. We start with the methods in J. A. Martín-Fernández, Barceló-Vidal, and Pawlowsky-Glahn (2003) and later move on to those in Palarea-Albaladejo and J. A. Martín-Fernández (2015).

Let $\boldsymbol{x} \in \mathbb{S}^D$ be a compositional vector and assume it has m zeros. Further take $\boldsymbol{r} \in \mathbb{S}^D$ as its zero free replacement. Let $\boldsymbol{S}$ be the selection matrix of the non-zero components and define a sub composition as $\boldsymbol{x}_s = \mathcal{C}(\boldsymbol{S}\boldsymbol{x})$. If we have rounded zeros, a simple method proposed in J. A. Martín-Fernández, Barceló-Vidal, and Pawlowsky-Glahn (2003) is to replace zero values with $DL \cdot 0.65$ where DL is the detection limit and 0.65 was found to be optimal to minimise the distortion in the covariance structure, see Lubbe, Filzmoser, and Templ (2021). This means $\boldsymbol{r}$ has the form

$$r_j = \begin{cases} 0.65 \cdot DL, & \text{if } x_j = 0, \\ x_j, & \text{if } x_j > 0. \end{cases} \tag{3.38}$$

Additionally J. A. Martín-Fernández, Barceló-Vidal, and Pawlowsky-Glahn (2003) mention three other methods.

First, the Additive Replacement Strategy, which was first introduced by Aitchison in Aitchison (1986), and is given by

$$r_j = \begin{cases} \frac{\delta(m+1)(D-m)}{D^2}, & \text{if } x_j = 0, \\ x_j - \frac{\delta(m+1)m}{D^2}, & \text{if } x_j > 0, \end{cases} \tag{3.39}$$

where D is the dimension and m the amount of zeros in $\boldsymbol{x}$. As we can see in (3.39), both zero and non-zero values are modified. In addition, this rule can be extended by using a different δ_j for each component x_j. However, the additive replacement strategy is additive for non-zero values and hence not coherent with the basic operations of $\mathbb{S}^D$. Other properties include:

1. The replacement value r_j depends on both, the amount of zeros m and the dimension D.

2. For two vectors $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{S}^D$ with common zeros, i.e. $x_j = 0 \leftrightarrow y_j = 0, j = 1, \ldots, D$, their sub compositions $\boldsymbol{x}_s, \boldsymbol{y}_s$ on their non-zero parts, and their replacements $\boldsymbol{r}^x, \boldsymbol{r}^y$, the Aitchison distance is not preserved $d_a(\boldsymbol{r}^x, \boldsymbol{r}^y) \neq d_a(\boldsymbol{x}_s, \boldsymbol{y}_s)$.

3. Ratios are not preserved. If $\boldsymbol{x}$ has more than one zero, then $\frac{r_j}{r_k} \neq \frac{x_j}{x_k}$ for $x_j, x_k > 0$.

4. The value $\frac{r_j}{r_k}$ depends on δ. Therefore, the covariance structure of the sub compositions of the non-zero parts is not preserved,

see J. A. Martín-Fernández, Barceló-Vidal, and Pawlowsky-Glahn (2003).

Second, the Simple Replacement Strategy, which formalises the procedure of replacing the zeros in $\boldsymbol{x}$ with a small positive value δ, obtaining a strictly positive vector $\boldsymbol{w} \in \mathbb{R}_+$ and applying the closure operation $\boldsymbol{r} = \mathcal{C}(\boldsymbol{w})$

$$r_j = \begin{cases} \frac{\kappa}{\kappa + \sum_{i|x_i=0} \delta_i} \delta_j, & \text{if } x_j = 0, \\ \frac{\kappa}{\kappa + \sum_{i|x_i=0} \delta_i} x_j, & \text{if } x_j > 0. \end{cases} \tag{3.40}$$

This method depends again on δ_j and the number of zeros m, see J. A. Martín-Fernández, Barceló-Vidal, and Pawlowsky-Glahn (2003).

Third is the multiplicative replacement strategy, which is the main result of J. A. Martín-Fernández, Barceló-Vidal, and Pawlowsky-Glahn (2003). The proposed replacement is

$$r_j = \begin{cases} \delta_j, & \text{if } x_j = 0, \\ \left(1 - \frac{\sum_{i|x_i=0} \delta_i}{\kappa}\right) x_j, & \text{if } x_j > 0, \end{cases} \tag{3.41}$$

where δ_j is the imputed value. It has the following properties

1. It is a more intuitive approach. If δ_j is close to the actual censored value, then $\boldsymbol{r}$ recovers the true composition. Further it does not depend on the number of zeros m or the dimension D.

2. It is compatible with the Simplex vector space structure. For $\boldsymbol{x} \in \mathbb{S}^D$, its non-zero version $\boldsymbol{r}$ and their sub compositions $\boldsymbol{x}_s = \mathcal{C}(\boldsymbol{S}\boldsymbol{x}), \boldsymbol{r}_s = \mathcal{C}(\boldsymbol{S}\boldsymbol{r})$, it holds
   - Subcomposition Invariance: $\boldsymbol{x}_s = \boldsymbol{r}_s$,
   - Perturbation Invariance: $\forall \boldsymbol{y} \in \mathbb{S}^D : (\boldsymbol{y} \oplus_a \boldsymbol{r})_s = (\boldsymbol{y} \oplus_a \boldsymbol{x})_s$,
   - Power transformation Invariance: $\forall \alpha \in \mathbb{R} : (\alpha \odot_a \boldsymbol{r})_s = (\alpha \odot_a \boldsymbol{x})_s$.

3. Ratios are preserved, which implies that the covariance structure for non-zero components is preserved. For $x_j, x_k > 0$ it holds $\frac{r_j}{r_k} = \frac{x_j}{x_k}$.

4. Let again $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{S}^D$ be two vectors with common zeros and their replacements $\boldsymbol{r}^x, \boldsymbol{r}^y$ which were obtained with the same imputation δ_j. Then it holds $\frac{r_j^x}{r_j^y} = \frac{x_j}{y_j}$ for $x_j, y_j > 0$ and $d_a(\boldsymbol{r}^x, \boldsymbol{r}^y)$ does not depend on the imputed values,

see J. A. Martín-Fernández, Barceló-Vidal, and Pawlowsky-Glahn (2003).

Another method proposed in Lubbe, Filzmoser, and Templ (2021) is to replace rounded zeros with values drawn from a continuous uniform distribution $U(0.1 \cdot DL, DL)$. Setting the lower limit to $0.1 \cdot DL$ makes sure that the values are not getting too close to zero and not using a constant prevents underestimation of the variability. They further present the R-package *zCompositions* by Palarea-Albaladejo and J. A. Martín-Fernández (2015).

The authors in Palarea-Albaladejo and J. A. Martín-Fernández (2015) focus on the case of rounded zeros which can be seen as left censored data. Their package includes some more advanced methods which are based on Markov Chain Monte Carlo (MCMC), the EM-algorithm or multiple imputation to perform imputation. They assume the data is left-censored, or Type 1 censored, and follows a multivariate normal distribution in $\mathbb{R}^D$. We review some of their methods presented and refer for more details to Palarea-Albaladejo and J. A. Martín-Fernández (2015).

EM-based algorithm

The Expectation-Maximisation (EM) algorithm of Dempster, Laird, and Rubin (1977) is a widely used method in imputation. In the setting of multivariate compositional data, it uses the information in the covariance structure to conditionally estimate the censored values. Define a censoring pattern, as the indices of the missing components for a composition $\boldsymbol{x}$. Denote the missing components for a composition with $\boldsymbol{x}_{non}$ and the observed components with $\boldsymbol{x}_{obs}$. Let r be the number of observed components and m the number of non observed components. Then the EM-algorithm consists of two steps with the t-th iteration given by

1. E-Step: Given a parameter estimate $\hat{\boldsymbol{\theta}}^{(t)}$, compute $\mathbb{E}[\boldsymbol{x}_{non}|\boldsymbol{x}_{obs}, \boldsymbol{x}_{non} < DL; \hat{\boldsymbol{\theta}}^{(t)}]$.
2. M-Step: Compute a new estimate $\hat{\boldsymbol{\theta}}^{(t+1)}$ based on $[\hat{\boldsymbol{x}}_{non}, \boldsymbol{x}_{obs}]$.

Here, DL is the mapped censoring threshold. Assuming a multivariate normal distribution, the conditional expected value of $\boldsymbol{x}_{non} = (x_{non_1}, \ldots, x_{non_m})^T$ at step t is given by

$$\hat{\boldsymbol{x}}^{(t)}_{non_j} = \hat{\beta}_{1j} + \sum_{i=1}^{r} x_{obs_i}\hat{\beta}_{(i+1),j} - \hat{\sigma}_j\hat{\lambda}_j \quad \text{for } j = 1, \ldots, m, \tag{3.42}$$

where $\hat{\lambda}^{(t)}_j = \frac{\phi((DL-(1,\boldsymbol{x}^T_{obs})\hat{\boldsymbol{\beta}}^{(t)}_j)/\hat{\sigma}^{(t)}_j)}{\Phi((DL-(1,\boldsymbol{x}^T_{obs})\hat{\boldsymbol{\beta}}^{(t)}_j)/\hat{\sigma}^{(t)}_j)}$ is the inverse Mills ratio. The function ϕ denotes the standard normal density and Φ is its distribution. The parameter $\hat{\boldsymbol{B}} = (\hat{\boldsymbol{\beta}}_1, \ldots, \hat{\boldsymbol{\beta}}_m) \in \mathbb{R}^{(r+1)\times m}$ is the ML estimate of the regression parameters and $\hat{\sigma}^2_j$ is the ML estimate of the conditional variance for $j = 1, \ldots, m$, see Palarea-Albaladejo and J. A. Martín-Fernández (2015). As seen, an initial estimation is required to kick start the iteration. This can be done by either using a subset of the data which was fully observed or by using other imputation methods, see Palarea-Albaladejo and J. A. Martín-Fernández (2015).

MCMC data augmentation

The Markov Chain Monte Carlo (MCMC) algorithm can be seen as the Bayesian counter part to the EM-algorithm. While with the use of priors, external information can be incorporated, in general, non-informative priors are used. With the same notation as above, the algorithm consists of two steps again

1. Imputation-Step: Given $\hat{\boldsymbol{\theta}}^t$, simulate from $\mathbb{P}(\boldsymbol{x}_{non}|\boldsymbol{x}_{obs}, \boldsymbol{x}_{non} < DL; \hat{\boldsymbol{\theta}}^t)$.

2. Posterior-Step: Generate $\hat{\boldsymbol{\theta}}^{t+1}$ by simulating from $\mathbb{P}(\boldsymbol{\theta}|\hat{\boldsymbol{x}}_{non}, \boldsymbol{x}_{obs})$.

In the imputation step, for $\boldsymbol{x}_{non} = (\boldsymbol{x}_{non_1}, \ldots, \boldsymbol{x}_{non_m})^T$, the value $\hat{\boldsymbol{x}}_{non_j}$ is drawn from the conditional, right-truncated normal distribution with estimated mean $(1, \boldsymbol{x}_{obs}^T)\hat{\boldsymbol{\beta}}_j^{(t)}$, variance $\hat{\sigma}_j^2$, and truncation point given by DL, for $j = 1, \ldots, m$. The posterior step simulates the parameter $\boldsymbol{\theta}$ from a normal inverted-Wishart distribution with non-informative priors. This generates a Markov Chain with the posterior distribution of the transformed censored data as the stationary distribution. After enough iterations, suitable random values can then be drawn from the chain as a replacement, see Palarea-Albaladejo and J. A. Martín-Fernández (2015).

Bayesian-multiplicative replacement

A method for count data is the Bayesian-multiplicative replacement. For multivariate count data one often assumes, that a vector $\boldsymbol{x}$ is a realisation from a multinomial distribution with parameters $[n, \pi_1, \ldots, \pi_D]$ where π_j is the probability of belonging to category j. For the prior distribution of $\boldsymbol{\pi} = [\pi_1, \ldots, \pi_D]$, an imprecise Dirichlet model with parameter s and $\boldsymbol{t} = [t_1, \ldots, t_D]$ with $\sum_k t_k = 1$ and expectation $\mathbb{E}[\pi_j] = t_j$ is considered. The posterior expectation is then given by

$$\mathbb{E}[\pi_j|x_j = 0] = t_j \frac{s}{n+s}. \tag{3.43}$$

Depending on the settings for s and $\boldsymbol{t}$ and based on (3.43), the imputation can be performed by geometric Bayesian multiplicative (BM), square root BM, or Bayes–Laplace, see Palarea-Albaladejo and J. A. Martín-Fernández (2015). The details of those methods can be found in J.-A. Martín-Fernández et al. (2015).

3.6.2 Essential Zeros

The case of essential zeros is not as straightforward because zero is the true value of the observation. In Aitchison and Kay (2003), the authors question the experimental design in case of many essential zeros. They point out to overly fine division of the data or the insignificance of the category as possible design faults. A solution in that case would be the amalgamation of categories with low counts. Further, they also introduce a two stage model. The first stage models the appearance of essential zeros, while in the second stage the non-zero components are generated. The maximum likelihood estimates of the parameters are suggested to be done via an MCMC algorithm. After

this is done, hypothesis testing and statistical analysis can be performed, see Aitchison and Kay (2003).

A vector space approach for the simplex is presented in Boogaart, Tolosana-Delgado, and Bren (2006) and extended in the R-package *compositions*, see van den Boogaart, Tolosana-Delgado, and Bren (2023). The idea is based on the clr coefficients (3.8) and the spanned subspace. Let M contain the indices of the missing parts. Then according to Egozcue and Pawlowsky-Glahn (2005), a subcomposition can be seen as a projection of the *clr* transformed composition into the orthogonal complement of the vectors $\{\boldsymbol{w}_i : i \in M\}$, with $\boldsymbol{w}_i = \boldsymbol{e}_i - \frac{1}{D}\mathbf{1} \in V$ and $V = \left\{\mathbf{w} \in \mathbb{R}^D : \sum_{i=1}^D w_i = 0\right\} \subset \mathbb{R}^D$ where $\boldsymbol{e}_i$ are the unit vectors and $\mathbf{1} = (1, \ldots, 1)^T \in \mathbb{R}^D$. Hence, one only observes a projection of the true composition. Let P_M be the projection onto the orthogonal complement of $\{\boldsymbol{w}_i : i \in M\}$ and $\boldsymbol{x}$ a composition with zeros. Then the idea is to represent the information of $\boldsymbol{x}$ by the projected values $P_M(clr(\boldsymbol{x}))$ and P_M itself. If M^C denotes the complement of M, so the indices of the non-zero parts, and $\boldsymbol{x}_s$ is the sub composition of $\boldsymbol{x}$ of M^C then for this sub composition it holds

$$P_M(clr(\boldsymbol{x}))_i = \begin{cases} clr(\boldsymbol{x}_s)_i, & \text{if } i \notin M \\ 0, & \text{if } i \in M. \end{cases} \tag{3.44}$$

The subsequent *ilr* transformation is then based on this modified approach with $ilr_{\mathbf{V}}(\mathbf{x}) = \mathbf{V}^T P_M(clr(\mathbf{x}))$, see van den Boogaart, Tolosana-Delgado, and Bren (2023).

In Leininger et al. (2013) they provide a review of other possible methods for handling essential zeros. They also introduce a model themselves, which allows zeros by modifying the *alr* transformation with the help of latent variables. Assuming a category with no zero values for all observations and taking it as the baseline component, they allow for transformation into a lower dimensional space where they can perform regression, see Leininger et al. (2013).

4 Application

4.1 Model Specifications

As our data has a specific structure, some transformations can be made to increase performance and stability. The most prominent characteristic of our data is its amount of 0 or null values. As CoDA can't handle 0 values, we have to accommodate for this. The concrete way to do this will be described in the following subsections.

Another varying factor is the history. We define the history h as the proportion of the length of the time series used for our model. While at first it may seem obvious to use as much data as possible, it may actually not always result in a better model. Older values may contain outdated information, which influences the estimation the of parameters. Therefore we compare the performance of the models with various history lengths. So instead of using T_f points in time, we will only use $T = h \cdot T_F$ with $0 < h \leq 1$.

Closely related to the length of the history, is the shape of the window used. The window determines which values are used to estimate the parameters at each point in time. The shape includes both the initial length of the window and the way new values are handled. As the different time series vary in length, we choose the possible window length as a fraction of the time series history. Hence, we define the initial window length as $w_f = w \cdot T$ with $0 < w \leq 1$. For the way how new values are handled, we focus on two different approaches. The first one uses a fixed window length. This means when a new time point is available, it will be included in the estimation while simultaneously the oldest time point will be removed from the estimation. This has the advantage of only using the most recent and relevant information. The second approach extends the window at each point in time. When a new value is available, it is included in the estimation of the parameter. With this approach we have more data available at each step and combined with the varying history length we don't have to rely on information that is too old.

The optimal one-step ahead prediction for the different models is given in their respective theoretical sections. However, since none of the models return integer-valued

results, we round the predicted values to the nearest integer.

4.1.1 CoDA Specifications

As mentioned above, the CoDA model must not include any zero values, since in this context, a value of zero is not defined. In order to keep things simple, we consider two options. The first one adds 0.5 to all time series values. The second one only replaces zero values with a chosen value δ, which is the simple replacement strategy in (3.40). Due to the fact that we have essential zeros and want to use the specific ilr coordinates, we opt for these options.

Another way to handle the zero values and the low values for some categories is a method we will call in the following one-vs-all. The principle is the following. A category k is chosen as the pivot category k_{pivot}. For all the chosen time points, at each point, the values of the other categories get summed up

$$Y_{other,t} = \sum_{\substack{k=1 \\ k \neq k_{pivot}}}^{K} Y_{kt}. \tag{4.1}$$

Together with the pivot category, the sum of the other categories are then transformed as usual and the VAR model is calculated

$$\boldsymbol{u}_t = ilr([Y_{other,t}, Y_{k_{pivot},t}]). \tag{4.2}$$

All categories are chosen as a pivot category at one point and the predicted values of the pivot groups are then used as the final result. This method is basically an implementation of the suggestions made in Aitchison and Kay (2003). We amalgamate all but one category and therefore change the experimental design.

As already hinted in the description of the methodology, we consider the use of $\mathcal{T}$-Spaces. For this, at each time point, we calculate the total amount and include it as an additional variable in the model. In addition, we can choose to take the logarithm of the sum. This means we have

$$\boldsymbol{w}_t = [ilr(\boldsymbol{Y}_t), t(\boldsymbol{Y}_t)], \tag{4.3}$$

with $t(\boldsymbol{Y}_t) = \sum_{k=1}^{K} Y_{kt}$ or $t(\boldsymbol{Y}_t) = \log\left(\sum_{k=1}^{K} Y_{kt}\right)$ and we get Model (3.37).

4.1.2 INGARCH Specifications

As an alternative to the Poisson distribution in (2.3), a Negative Binomial distribution can be used as well. This would change (2.3) to

$$p_{kt}(y;\boldsymbol{\theta}) = \mathbb{P}(Y_{kt} = y|\mathcal{F}_{k,t-1}) = \frac{\Gamma(\phi_k + y)}{\Gamma(y+1)\Gamma(\phi_k)} \left(\frac{\phi_k}{\phi_k + \lambda_{kt}}\right)^{\phi_k} \left(\frac{\lambda_{kt}}{\phi_k + \lambda_{kt}}\right)^{y}, \quad y \in \mathbb{N}_0. \tag{4.4}$$

where ϕ_k is the dispersion parameter, λ_{kt} the mean, and Γ is the gamma function. With the Negative Binomial Distribution, the conditional variance is larger than the conditional mean $\lambda_{kt} = \mathbb{V}[Y_{kt}|\mathcal{F}_{k,t-1}] > \mathbb{E}[Y_{kt}|\mathcal{F}_{k,t-1}]$.

As seen in the Model (2.2), we can also choose to include external factors. However, as our data is of the structure where we don't have information about $\boldsymbol{X}_t$ at time t, we cannot make use of it.

The values p and q are also varying parameters which have to be chosen. One could use the AIC or some other criteria to get the optimal lag order. However, this is not in the scope of this thesis and hence will be left as a future extension.

4.2 Error Measure

In order to compare the results of the methods with each other, we will introduce a new error measure. The goal of this measure is to get a performance indicator for each fridge, which can be used for comparison and summarisation. Since the scales of the fridges vary, the measure should be scale independent but because our data contains many zeros, we cannot use a percentage error measure. In addition, we want to penalise big absolute differences between the predicted values and actual values. These requirements lead us to the following measure.

For a fridge f, let $t = 1, \ldots, T$ denote the point in time and $k = 1, \ldots, K$ the category. Then y_{ftk} is the t-th measured value of the time series for category k and fridge f, $\hat{y}_{ftk}$ the predicted value, and $\hat{y}_{naive_{ftk}}$ the value predicted by the naive random walk model (2.4). Then we define our measure as

$$E_f = \frac{\sum_{k=1}^{K}\sum_{t=1}^{T}(y_{ftk} - \hat{y}_{ftk})^2}{\sum_{k=1}^{K}\sum_{t=1}^{T}(y_{ftk} - \hat{y}_{naive_{ftk}})^2}. \tag{4.5}$$

With the use of the squared difference, we penalise big deviations from the measured value. By taking the naive random walk model as a benchmark, we achieve scale in-

dependence and are able to compare the performance of our model over different time series. This error measure is basically the ratio of the mean MSEs for the chosen model and the naive random walk model

$$E_f = \frac{\frac{1}{K}\sum_{k=1}^{K} MSE_{fk}}{\frac{1}{K}\sum_{k=1}^{K} MSE_{naive_{fk}}}. \tag{4.6}$$

If the ratio is below 1, the mean of the MSEs of our method is lower than that of the naive method and vice versa. This provides a performance indicator for our models.

Extension of the Error Measure

The measure in (4.5) can be further extended. For example, by allowing to use a subset of all possible categories instead of all. Let $G_K \subset \{1, \ldots, K\}$ then

$$E_f^{GK} = \frac{\sum_{k \in G_K}\sum_{t=1}^{T}(y_{ftk} - \hat{y}_{ftk})^2}{\sum_{k \in G_K}\sum_{t=1}^{T}(y_{ftk} - \hat{y}_{naive_{ftk}})^2}. \tag{4.7}$$

This allows us to compare the performance on the subset of categories over various fridges.

Another possible extension is to take the square root

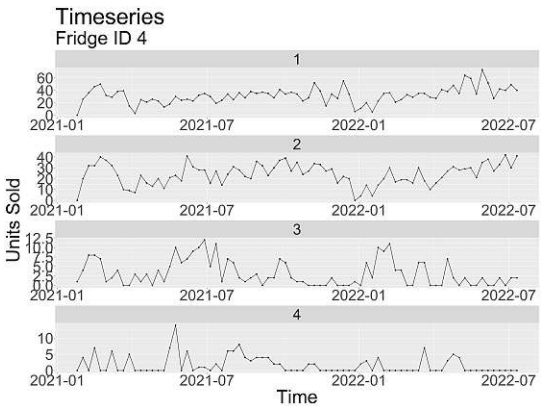
$$\widetilde{E}_f = \frac{\sum_{k=1}^{K}\sqrt{\sum_{t=1}^{T}(y_{ftk} - \hat{y}_{ftk})^2}}{\sum_{k=1}^{K}\sqrt{\sum_{t=1}^{T}(y_{ftk} - \hat{y}_{naive_{ftk}})^2}}. \tag{4.8}$$

One future extension which can be investigated is the introduction of weights. This could be used for example when the performance of the model in one category should be put more into focus.

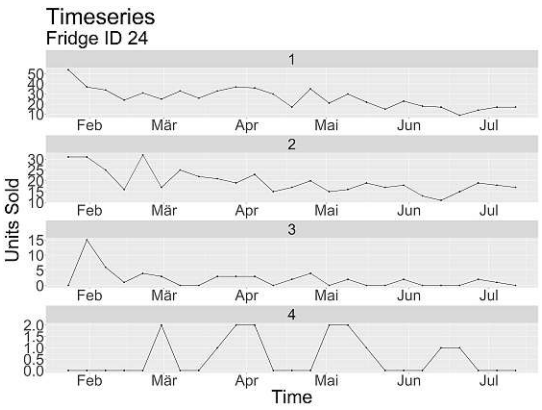
4.3 Examples of Model Application

To improve understanding of our data and the models, we show some application of the models on some exemplary fridges. We choose fridges 4 and 24 so $f \in \{4, 24\}$. Furthermore we start with analysing the aggregated 4 main categories which means $K = 4$.

We first begin with plotting the values of time series. The x-axis shows the time and the y-axis the number of units sold. Since we have four main categories for each fridge, we have four subplots.



(a) Fridge 4 with all four main categories



(b) Fridge 24 with all four main categories

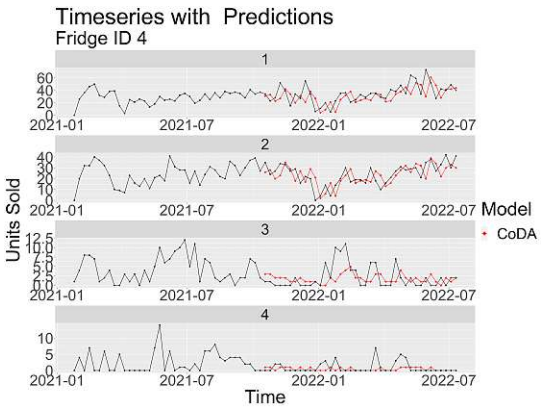
Figure 4.1: Time series for two fridges

The two plots in 4.1 are good examples of the composition of our data. The scales of the sold units within a fridge vary widely. For example in Figure 4.1b the values for category 1 vary from above 50 to as low as 10, while for category 4 we only have values in the range of 0 to 2. In both Subfigures 4.1 for category 4, we can see the excessive amount of zero values in our data, which makes the previously mentioned zero handling necessary.

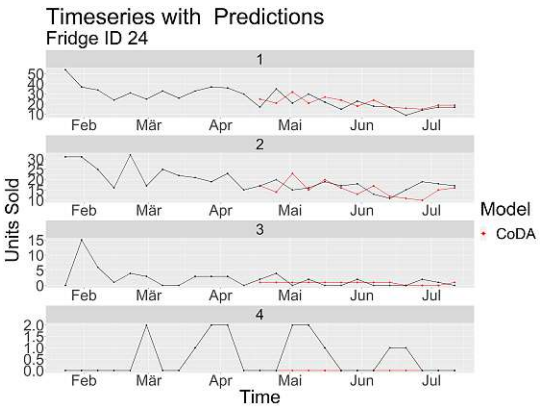
Next, in Figure 4.2, we add the predictions of the CoDA model. For this model we used the whole history $h = 1$ and half of the data for the window length $w = 0.5$. In addition, we extend the window at every time point, use the simple replacement strategy with $\delta = 0.1$, use no $\mathcal{T}-$Spaces, and use the one-vs-all method. We can see that this captures the general trend well however, struggles with unexpected high peaks. In addition, it is able to handle the difference in scales as seen in Figure 4.2a. Both, categories 1 and 2 with bigger values and categories 3 and 4 with lower values, are in general modelled well. Also for time series with less data available, as in fridge 24 4.2b, the model works well. Especially category 3 with its low values is predicted well.

In Figure 4.3 we apply the INGARCH model to the time series. For this, we used the whole history $h = 1$, half of the data for the initial window length $w = 0.5$, extend the window at every time point, add nothing to the zero values, and use the Poisson distribution. We use no external factors and set $p = 1, q = 1$ in Model (2.2). The general trend is again captured well and in the instance of Figure 4.3a it seems to be more reactive to sudden peaks, since often the value predicted after such a peak is heavily influenced by it.

To directly compare both models, we plot the predictions in one Figure 4.4. The

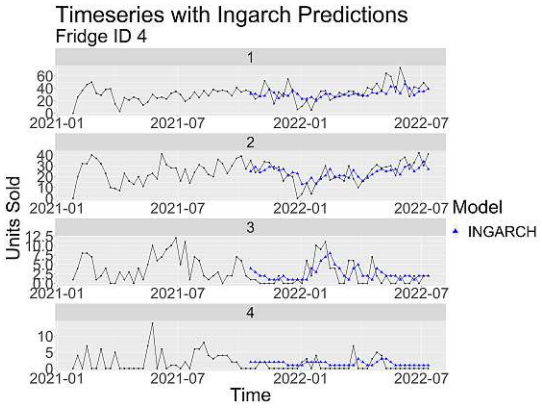


(a) Fridge 4 with the CoDA model

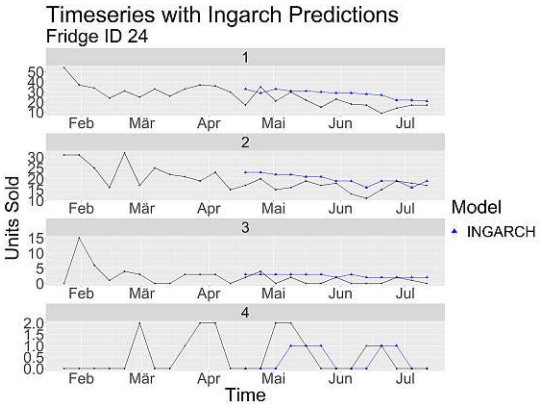


(b) Fridge 24 with the CoDA model

Figure 4.2: Time series with the CoDA model



(a) Fridge 4 with the INGARCH model

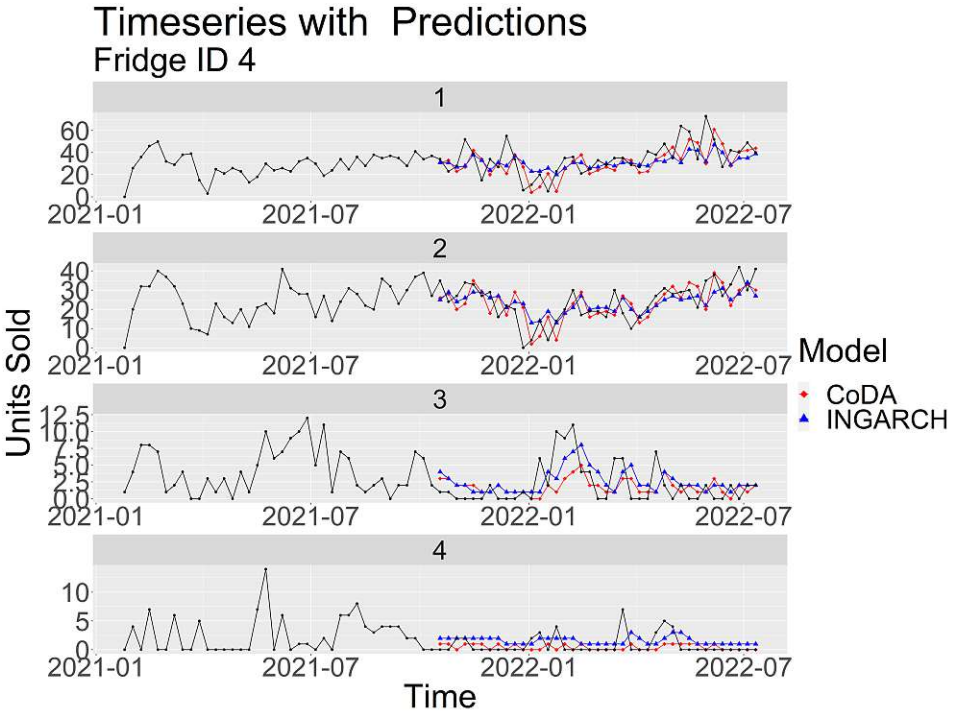


(b) Fridge 24 with the INGARCH model

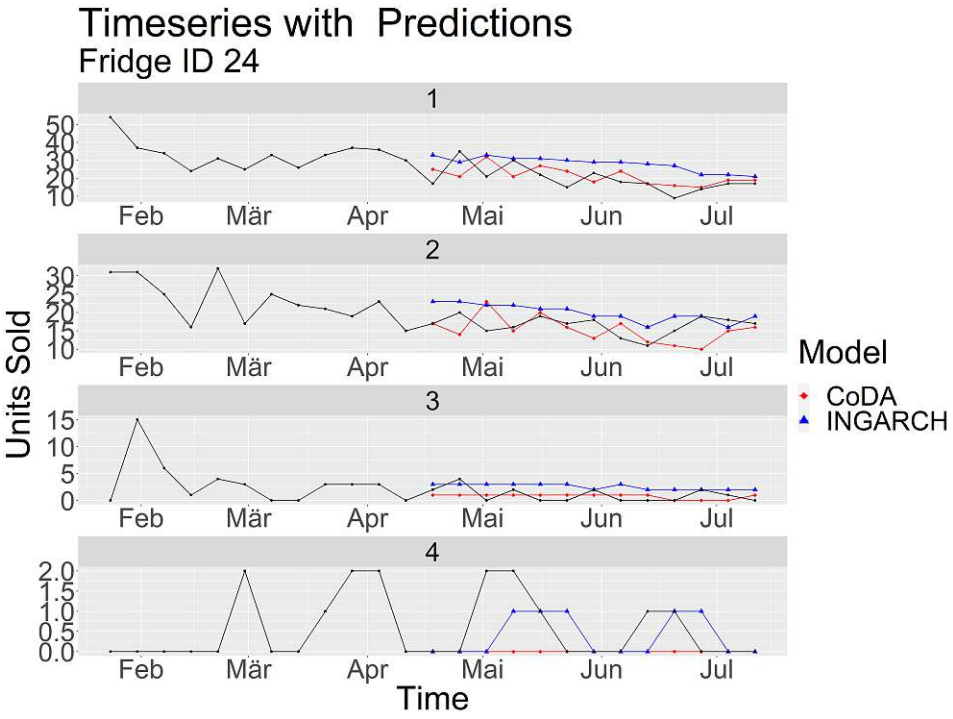
Figure 4.3: Time series with the INGARCH model

model specifications are the same as above. We can see that the models produce similar results to each other. In these instances it appears that INGARCH predicts slightly higher values than CoDA.

To get some further insight in the accuracy of our predictions, we add 95 % prediction intervals in Figure 4.5. Here, we can see some differences between the intervals. While for categories with bigger values the bands are quite similar in width, for categories with lower values, CoDA has much wider bands. This is especially visible in Figure 4.5a for category 3 and 4. However, most data points are covered by both bands.

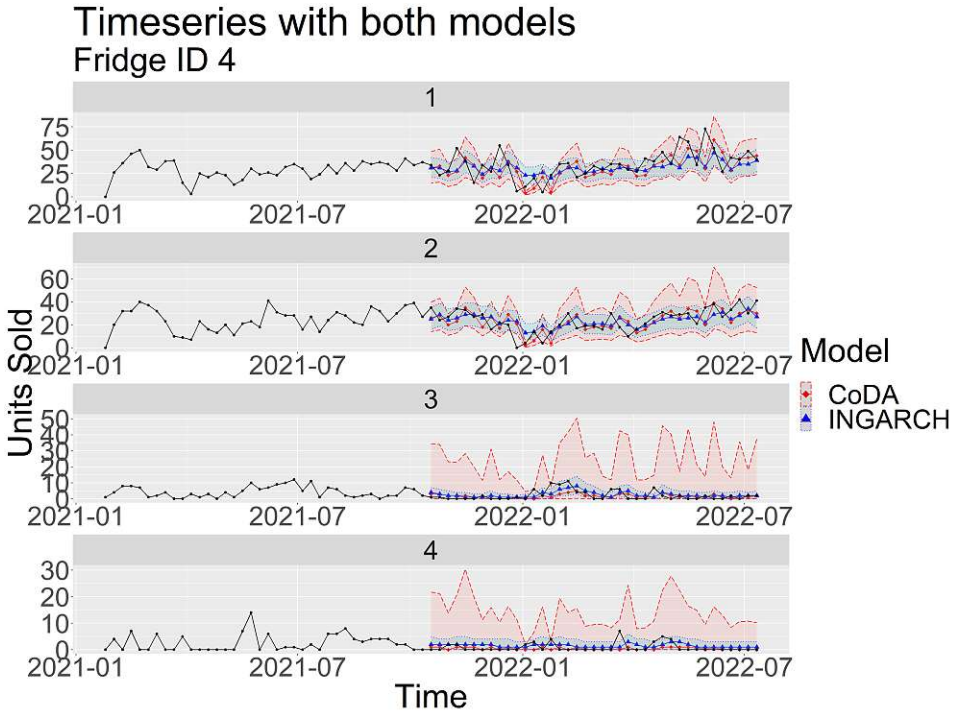


(a) Fridge 4 with both models

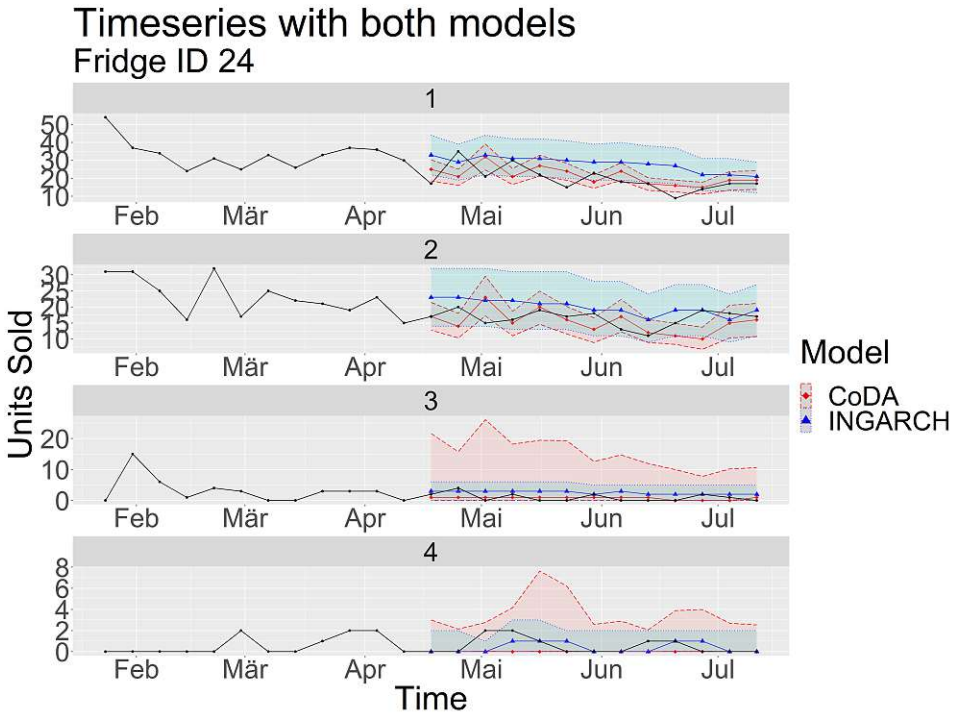


(b) Fridge 24 with both models

Figure 4.4: Time series with both models



(a) Fridge 4 with both models and their prediction intervals



(b) Fridge 24 with both models and their prediction intervals

Figure 4.5: Time series with both models and their prediction intervals

4.4 R-Code

4.4.1 R-Packages

We conduct our analysis in the statistical software R, see R Core Team (2022). For our data cleansing, data handling, and plotting we use the *tidyverse* package of Wickham et al. (2019). Further we use the packages *here* of Müller (2020), *miceadds* of Robitzsch and Grund (2023), and *parallel*, which is part of core R, to facilitate our analysis.

For building our CoDA model we use the packages *vars* of Pfaff (2008b) and Pfaff (2008a) and *robCompositions* of Templ, Hron, and Filzmoser (2011) and Filzmoser, Hron, and Templ (2018). Especially the functions `pivotCoord`, which performs the *ilr* transformation described in Section 3.2,`VAR`, which builds the VAR model described in Section 3.4, and `pivotCoordInv`, which performs the necessary back transformation to get predictions in the desired space. The INGARCH analysis is mainly done with the package *tscount*, see Liboschik, Fokianos, and Fried (2017) and Liboschik, Fried, et al. (2020). The core function used is `tsglm`, which we use to fit the INGARCH(p,q) model as well as the log-linear model. The zero-inflated models were fitted using the function `zeroinfl` from the package *pscl* of Zeileis, Kleiber, and Jackman (2008). To fit the INAR model we use two packages. First, *ZINAp* to calculate our predictions with the Bayesian approach. The function `estimate_zinarp` is used to estimate the coefficients and the values are then calculated according to the formula in N. Silva, Pereira, and M. Silva (2009). Second, the classical approach is done using the function `EST_ZINAR` from the package *ZINA1*.

In general, all other functions can be grouped into three categories: general, count data model specific, and CoDA specific. General functions are used for both, the count data models and the CoDA model. Count data models and CoDA specific functions are only used for their respective methods.

4.4.2 Handbook

In this handbook, we will describe the use and results of the most important functions used for our analysis. The code for them can be found in the GitHub repository, see Schwaiger (2023).

Data.Window

The function `Data.Window` splits the time series in the specified windows and the value to be predicted. The models are then fitted on these windows and the prediction result can be compared with the actual value.

Arguments:

- Timeseries: The time series to be split up in windows.
- Frame: The window length to split the time series into.
- Method: How the time series should be split up. For example if the windows should be extended at each step or be kept at a fixed length.
- PredictionStep: The future prediction step.

Values:
A list of all windows is returned. A window is a list with the following elements:

- timeSeriesValue_window: The values of the window.
- timeSeriesValue_future: The value which should be predicted by this window.

Data.Preparation

The function `Data.Preparation` transforms the data in the right format, replaces missing values with 0, and accounts for the length of the history chosen. In addition, for CoDA it also transforms the data into the right format needed for the one-vs-all method.

Arguments:

- Data_Raw: The data to be transformed in the right format.
- OneVsAll: When TRUE, then the one-vs-all method is used.
- PivotGroup: When one-vs-all is used, this specifies the pivot category.
- Category: The categories to consider for the transformation.
- NA_to: The value with which NA values should be replaced with.
- HistoryLength: The length of the history. Can be an absolute number or a ratio $0 < h \leq 1$.

- TakeSubCategory: When TRUE, we transform the data for the subcategories instead of the main categories.

Values:
A tibble with two or more columns is returned, depending on the number of categories:

- week_date: The dates of the recorded values of the window.
- *Name of Category 1*: The number of sold items belonging to *Name of Category 1*.

When the argument OneVsAll is TRUE, then a tibble with three columns is returned.

- week_date: The dates of the recorded values of the window.
- PivotGroup: The amount of sold items belonging to the pivot group.
- other: The amount of all other sold items belonging to the other categories.

Model.Error

The function `Model.Error` calculates the specified error measure for each time series and category. Since we want to compare the performance of a method with the naive model in Section 2.4, we calculate the errors for this model as well.

Arguments:

- Model_Result: The result of `Coda.Analysis` or `CountModel.Analysis`.
- Fnct: The error function to be used. Currently the MSE and RMSE are implemented.
- Category: The categories for which the errors should be caluculated.

Values:
A tibble with the columns is returned:

- id: The id of the fridge.
- category: The category for which the error was calculated.
- error: The error calculated according to the error function in the Fnct argument.
- error_naive: The value of the error function for the naive random walk model.
- model: The used model.

Model.ErrorOverall

The function `Model.ErrorOverall` is closely related to `Model.Error`. This function calculates the error measure defined in Section 4.2. One can decide if the error measure should be calculated over all categories or if they should be split up in subsets as in Subsection 4.2.

Arguments:

- Error_Result: The result of the function `Model.Error`.
- Fnct: Function to summarise the errors. This enables one to use different methods like the mean or median.
- SplitByGroup: When TRUE, then the errors are split by groups defined in the *Groups* argument.
- Groups: The grouped categories over which the error should be calculated.
- Category: The categories for which the error should be calculated for.

Values:
The result is a tibble with the columns:

- id: The id of the fridge.
- error: The error calculated according to the error function in the Fnct argument.
- model: The used model.
- group: The subsets of categories as defined in Subsection 4.2.

CountModel.DataPreparation

The function `CountModel.DataPreparation` transforms the data into the right format needed to fit the count data models. At its core it uses the `Data.Preparation` function but adds the additional option to replace zero values with 1.

Arguments:

- Data: The data to be transformed.
- ZeroHandling: Method for zero handling. Currently there is no treatment or them being replaced with 1.

- HistoryLength: The length of the history. Can be an absolute number or a ratio $0 < h \leq 1$.
- TakeSubCategory: When TRUE, we transform the data for the subcategories instead of the main categories.

Values:
A tibble with two or more columns is returned, depending on the number of categories:

- week_date: The dates of the recorded values of the window.
- *Name of Category 1*: The number of sold items belonging to *Name of Category 1*.

CountModel.Prediction

The function `CountModel.Prediction` is the function where the model is fit and the predicted value is calculated. It uses the corresponding functions mentioned in Section 4.4.1 to fit the INGARCH, INAR, or ZIM model for each window and predicts the next value.

Arguments:

- Data_Window: The data divided into the different windows by the `Data.Window` function.
- Data_WindowNoTransform: The data without zero handling divided into the different windows by the `Data.Window` function.
- Category: The category to predict.
- PredictionStep: The prediction step.
- Frame: The window length.
- Distribution: The distribution chosen for the model. Care has to be taken, since every model can choose from a different list of distributions and its name has to be specified correctly (i.e. "'Po"' for INAR but "'poisson"' for ZIM).
- Plot: For the INGARCH model, diagnostic plots can be generated. Currently not implemented.
- WindowMethod: Method for splitting up the time series. For example if the windows are extended at each step or kept at a fixed length.

- External: For INGARCH. When TRUE, external factors as in Equation (2.2) are used.
- PastOb: For INGARCH. How many past observations should be used. Equals p in Equation (2.1).
- PastMean: For INGARCH. How many past means should be used. Equals q in Equation (2.1).
- ModelType: Model to be fit.

Values:
It returns a list with two elements:

- prediction: A data.frame with the predicted values and some additional information.
- model: A list of all the models fitted for each window.

CountModel.Analysis

The function `CountModel.Analysis` acts as a wrapper function to streamline and facilitate the analysis. The previously mentioned model specifications can be chosen here as well as various other options. This is the sole function which has to be used by the user. The other functions are mainly for internal use.

Arguments:

- Data_Raw: The raw data as extracted from the data base.
- Id: The ids of the fridges to be analysed.
- PredictionStep: The future prediction step.
- Distribution: The distribution chosen for the model. Care has to be taken, since every model can choose from a different list of distributions and its name has to be specified correctly (i.e. "'Po"' for INAR but "'poisson"' for ZIM).
- ModelType: Model to be fit.
- Plot: For the INGARCH model, diagnostic plots can be generated. Currently not implemented.

- Category_Main: The main categories to choose.
- TakeSubCategory: When TRUE, then we transform the data for the subcategories instead of the main categories.
- Category_Sub: The sub categories to choose.
- Frame: The window length.
- WindowMethod: Method for splitting up the time series. For example if the windows are extended at each step or kept at a fixed length.
- ZeroHandling: Method for zero handling. Currently there is no treatment or them being replaced with 1.
- PastOb: For INGARCH. How many past observations should be used. Equals p in Equation (2.1).
- PastMean: For INGARCH. How many past means should be used. Equals q in Equation (2.1).
- External: For INGARCH. When TRUE, external factors as in Equation (2.2) are used.
- HistoryLength: The length of the history. Can be an absolute number or a ratio $0 < h \leq 1$.
- Multicore: When TRUE, then calculations are done on multiple cores to improve performance. Internally the parallelisation takes place across the different categories to be calculated.
- NCores: The number of cores to be used for parallelisation.

Values:
This function returns a list with two values:

- result: The analysis result in the form of a data.frame.
- model: A nested list with all models, fitted for each id, category and window.

Coda.DataPreparation

The function `Coda.DataPreparation` is analog to CountModel.Preparation

Arguments:

- Data: The data to be transformed.
- ZeroHandling: Method for zero handling. Currently there is no treatment, the simple replacement strategy or adding 0.5 to all values.
- TSpace: When TRUE, then $\mathcal{T}$-Spaces are used.
- Log: When TRUE, then the logarithm of the total sum is used in the $\mathcal{T}$-Space.
- OneVsAll: When TRUE, then the one-vs-all method is used.
- PivotGroup: When one-vs-all is used, this specifies the pivot category.
- HistoryLength: The length of the history. Can be an absolute number or a ratio $0 < h \leq 1$.
- DL: The value δ for the simple replacement strategy in (3.40).

Values:

The result is a tibble. The columns are the ilr transformed data and hence the number of columns depends on the dimension of the data:

- week_date: The dates of the recorded values of the window.
- *Name of ilr transformed category 1*: The ilr transformed values.

If $\mathcal{T}$-Spaces are used then an additional column with the sum or log-sum is added:

- week_date: The dates of the recorded values of the window.
- *Name of ilr transformed category 1*: The ilr transformed values.
- tsum: Either the total sum or log-sum.

Coda.Prediction

The function `Coda.Prediction` acts like its respective count model counterpart.

Arguments:

- Data_Window: The data divided into the different windows by the `Data.Window` function.
- Data_WindowNoTransform: The data without zero handling and no transformation divided into the different windows by the `Data.Window` function.
- Data_NoTransform: The data without zero handling and no transformation.
- PredictionStep: The future prediction step.
- OneVsAll: When TRUE, then the one-vs-all method is used.
- TSpace: When TRUE, then $\mathcal{T}$-Spaces are used.
- Log: When TRUE, then the logarithm of the total sum is used in the $\mathcal{T}$-Space.
- PivotGroup: When one-vs-all is used, this specifies the pivot category.
- Frame: The window length.

Values:

It returns a list with two elements:

- prediction: A data.frame with the predicted values and some additional information.
- model: A list of all the models fitted for each window.

Coda.Analysis

Again `Coda.Analysis` is the wrapper function. This is again the only function which needs to be used by the user to fit models for the specified time series.

Arguments:

- Data_Raw: The raw data as extracted from the data base.
- Id: The ids of the fridges to be analysed.

- Frame: The window length.
- ZeroHandling: Method for zero handling.
- PredictionStep: The future prediction step.
- Log: When TRUE, then the logarithm of the total sum is used in the $\mathcal{T}$-Space.
- TSpace: When TRUE, then $\mathcal{T}$-Spaces are used.
- OneVsAll: When TRUE, then the one-vs-all method is used.
- PivotGroup: When one-vs-all is used, this specifies the pivot category.
- HistoryLength: The length of the history. Can be an absolute number or a ratio $0 < h \leq 1$.
- ModelType: Model to be fit. Currently only "coda" and "coda_OneVsAll" can be chosen.
- WindowMethod: Method for splitting up the time series. For example if the windows are extended at each step or kept at a fixed length.
- DL: The value δ for the simple replacement strategy in (3.40).

Values:
This function returns a list with two values:

- result: The analysis result in the form of a data.frame.
- model: A nested list with all models, fitted for each id, category and window.

4.5 Results

In this section, we present and describe the results of our methods with their variations. For this we use the previously introduced error measure, calculate it for all available fridges and summarise the results. We show the results as graphics for easier interpretation. For a general comparison, we use the CoDA, INGARCH, ZIM, and INAR models. Since we focus on the CoDA and INGARCH models, we additionally analyse their results in more detail as well.

4.5.1 Model Comparison

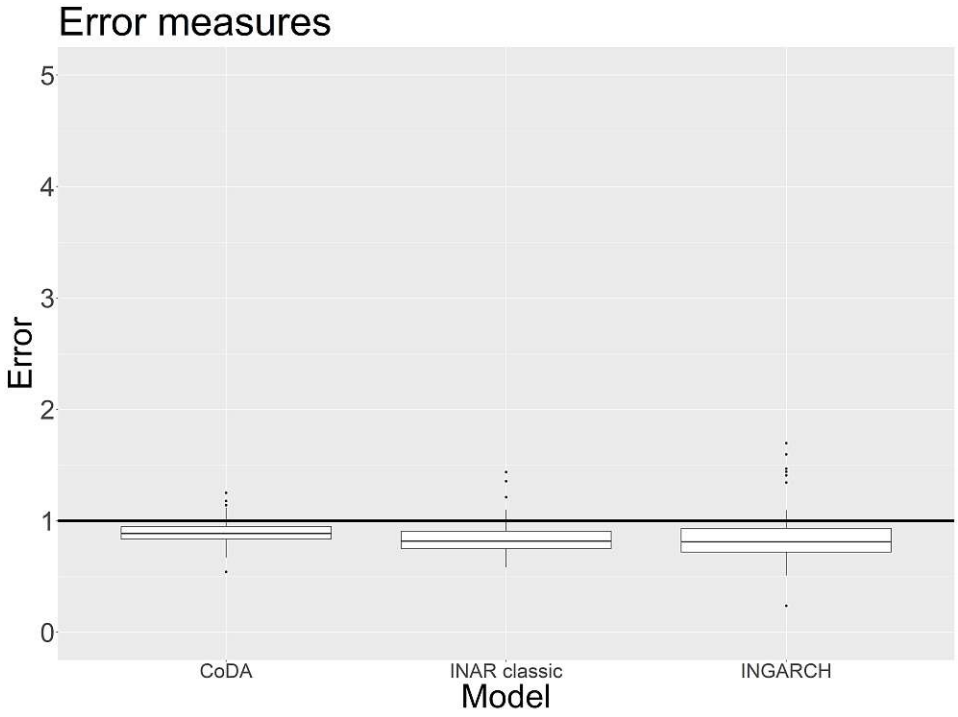
Here we compare the INGARCH(1,1) model with the CoDA and INAR(1) model. For all three models we use the same parameter values, namely a window factor of $w = 0.5$, the whole history $h = 1$, and extending windows. For CoDA, the settings are no $\mathcal{T}$-Space, one-vs-all method, and the simple replacement strategy with $\delta = 0.1$. For INGARCH(p,q) we use $p = q = 1$, assume it is Poisson distributed, use no external factors, and have no zero handling. For INAR(1) we use the classical forecasting method described in Section 2.8.2. When we speak of standard settings or values in the following, we mean these settings.

In Figure 4.6 we see a boxplot and quantile plot. In the boxplot the error measure is calculated for all groups and all fridges for each model. The result is then shown in a boxplot. In the quantile plot, the error measure for each fridge, each model, and each category is calculated and sorted according to their size. The dot size indicates the length of the respective time series and the vertical lines are the 0%,25%,50%,75%, and 100% quantile.

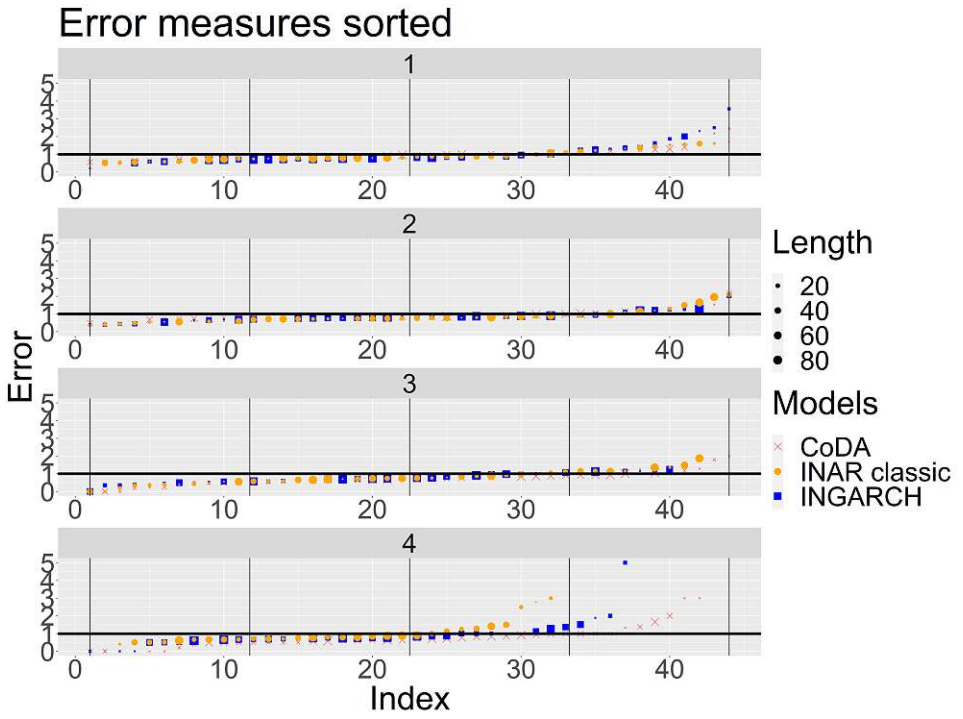
In the Boxplot 4.6a we can see that their performance is pretty similar. They all seem to outperform the naive random walk model, which is especially true for the count data models. In the Quantile Plot 4.6b we see the error measure split up by category. While for category 1 and 2 all models perform reasonably well and similar, differences emerge for category 3 and 4. CoDA seems to be the clear favourite in category 4, followed by INGARCH and then INAR. However, for all models there are time series with errors that are either too high to be shown, or that couldn't get calculated at all.

In Figure 4.7 we also include the ZIM model. One drawback about the ZIM model is, that it needs to have zero values in the fitted window. Because of the lack of them in category 1 and 2, we couldn't manage to fit it. Hence the Models in 4.7 were only fitted on categories 3 and 4.

In 4.7a we again see the boxplot for the summarised error. The ZIM is close to INGARCH, but all three models still lag behind CoDA. In 4.7b we see again the error measure for each category. While the models perform similar for category 3, the CoDA still outperforms all models for category 4. Here it is worth mentioning, that category 4 is the main category with the most amount of zeros in our data. It should be pointed out again, that all models don't predict integer values, but rather real values which then get rounded to the nearest integer value. Especially CoDA cannot predict zero values but only small positive values which get rounded to zero.

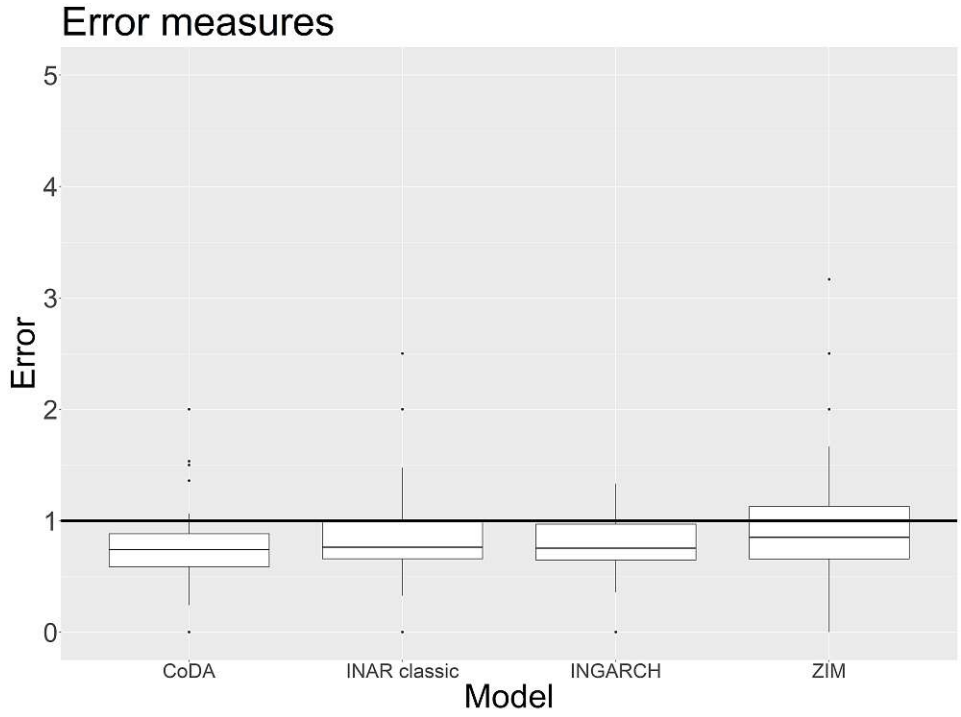


(a) Boxplot for the different models

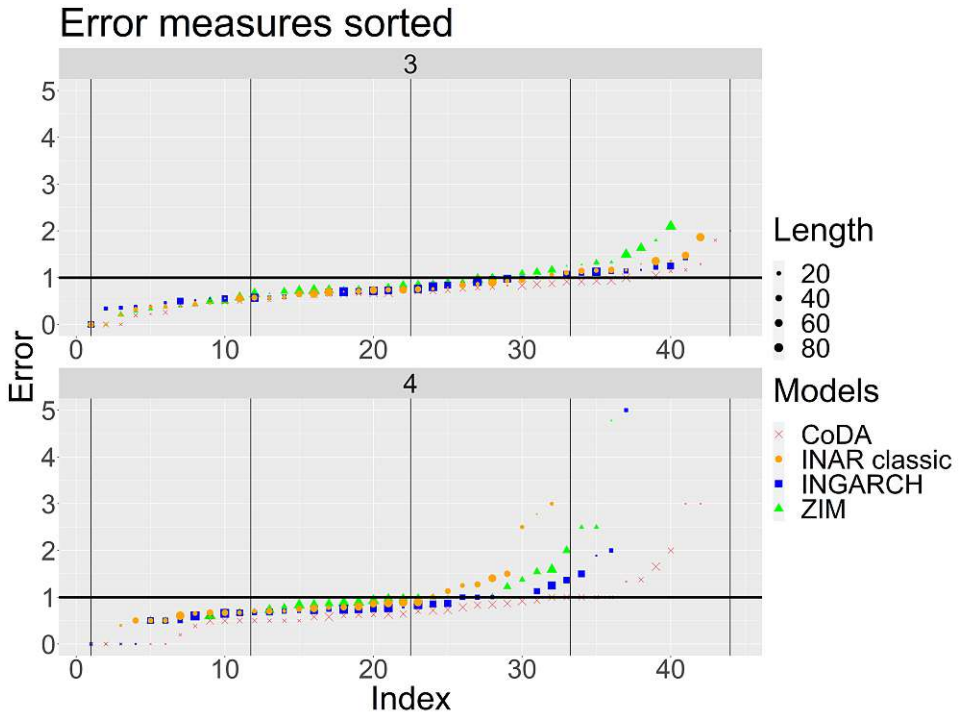


(b) Quantiles for the different models

Figure 4.6: Comparison of the different models for all main categories



(a) Boxplot for the different models



(b) Quantiles for the different models

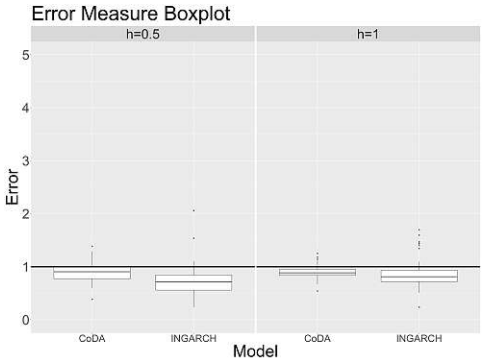
Figure 4.7: Comparison of the different models for main categories 3 and 4

4.5.2 General Specifications

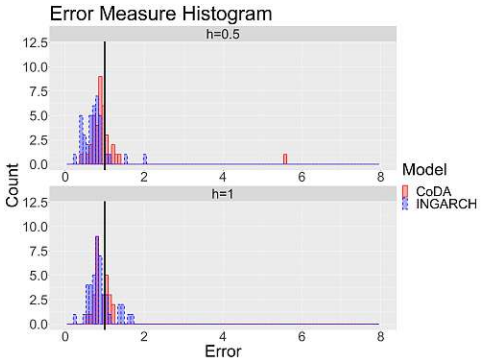
First, we start with specifications which can be chosen for both CoDA and INGARCH. We will always vary one parameter, while using the respective standard values for the other parameters.

History

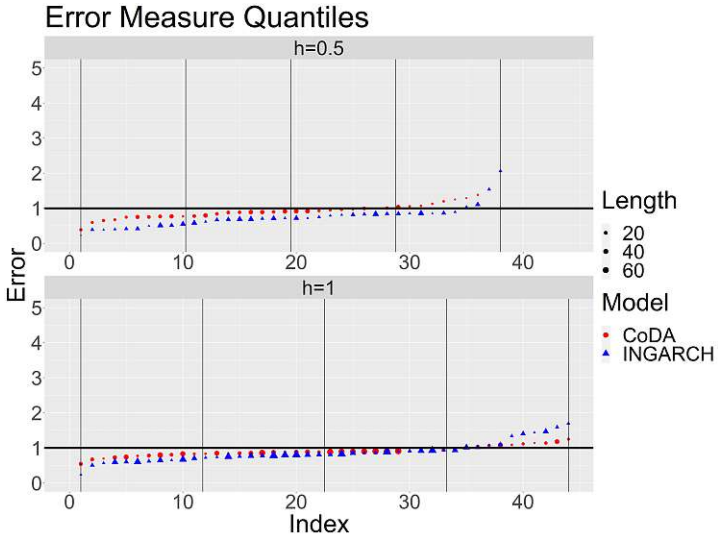
As mentioned various times throughout this thesis, the history is one of the parameters which can be adjusted. In Figure 4.8 we visualise the results as a boxplot, a quantile plot, and additionally a histogram to get a feeling for the error distribution.



(a) Boxplot for different h



(b) Histogram for different h



(c) Quantiles for different h

Figure 4.8: Comparison of different h

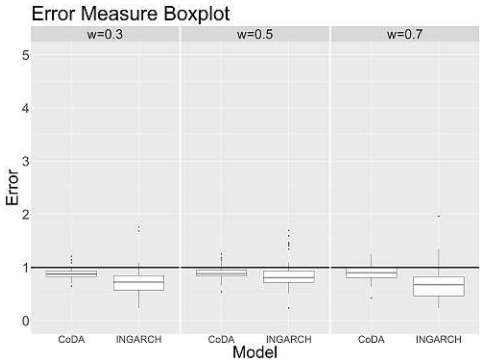
In Figure 4.8 we can see that the results for CoDA do not vary too much for the different histories. However, one can see in the Quantile Plot 4.8c that we have 8 less

values for $h = 0.5$ than for $h = 1$. This probably results from the fact, that if we only take half of the history for an already short time series, then we have too little values for estimation.

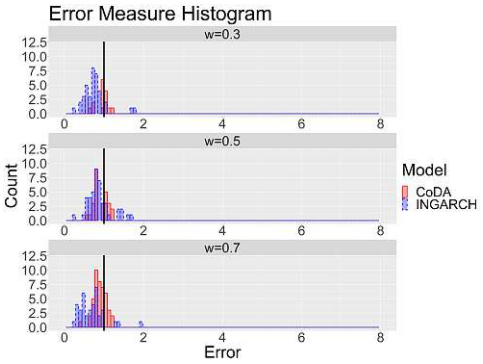
For INGARCH, the results are similar as well. For $h = 1$ we get slightly higher values for the error measure as seen in 4.8a. But again in 4.8c we see that we have less values for the shorter history for the same reason as above.

Frame

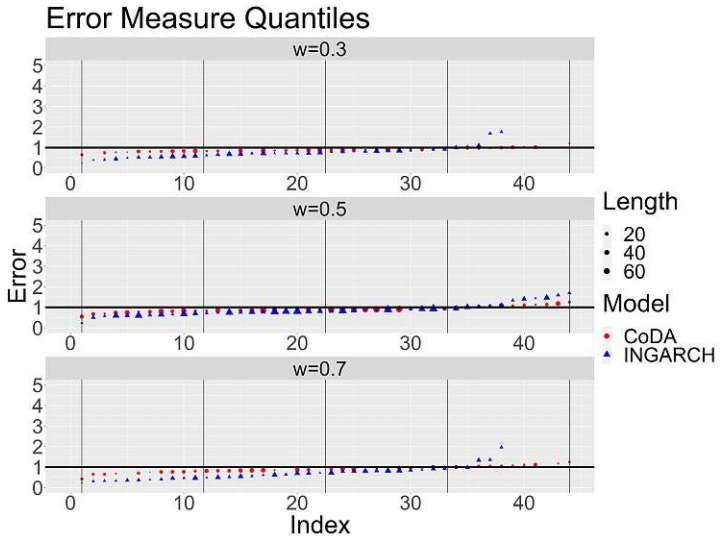
Next, we vary the initial frame length w_f. We choose to extend the frame with each new data point. For this we vary the value w in $w_f = w \cdot T$. The results are portrayed in 4.9. In general, there is not much difference between the different frames. INGARCH seems to perform better for all three values.



(a) Boxplot for different w



(b) Histogram for different w



(c) Quantiles for different w

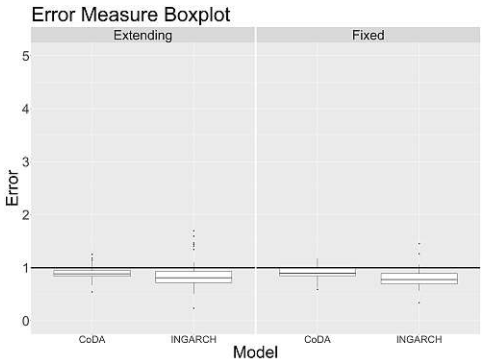
Figure 4.9: Comparison of different w

In the Boxplot 4.9a it looks like INGARCH performs worst for $w = 0.5$. However, in the Quantile Plot 4.9c we can see that for $w = 0.3, 0.7$ the last errors are not included in the plot. This could either be a result of them being too high, or that the model couldn't be fit on those fridges.

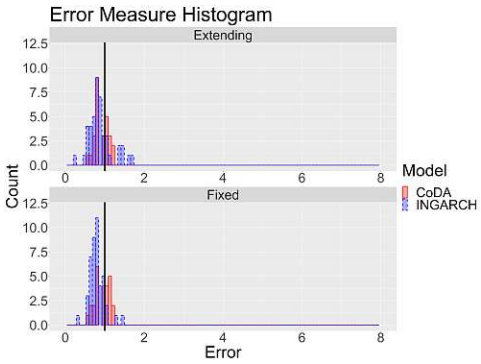
For CoDA there seems not to be much difference. The best results are obtained with $w = 0.3$, but the differences are only marginal.

Window Shape

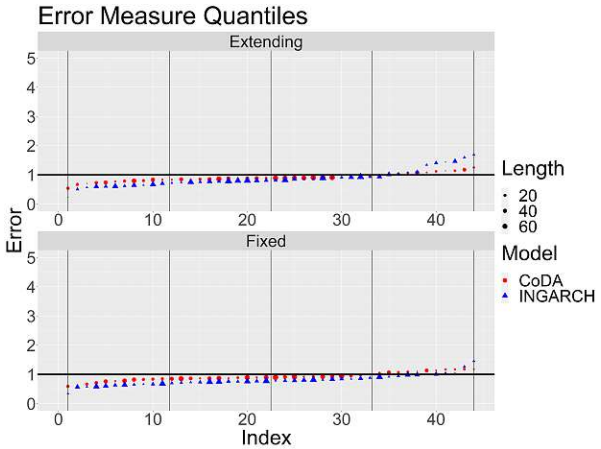
We also vary the shape of the window. As explained in Section 4.1 we either use a fixed amount of points and add and remove points as time goes on, or we continuously add points to the window. The results are in Figure 4.10. We can see that there are no big differences between the methods. For both, CoDA and INGARCH, there are no notable differences.



(a) Boxplot for different window shapes



(b) Histogram for different window shapes



(c) Quantiles for different window shapes

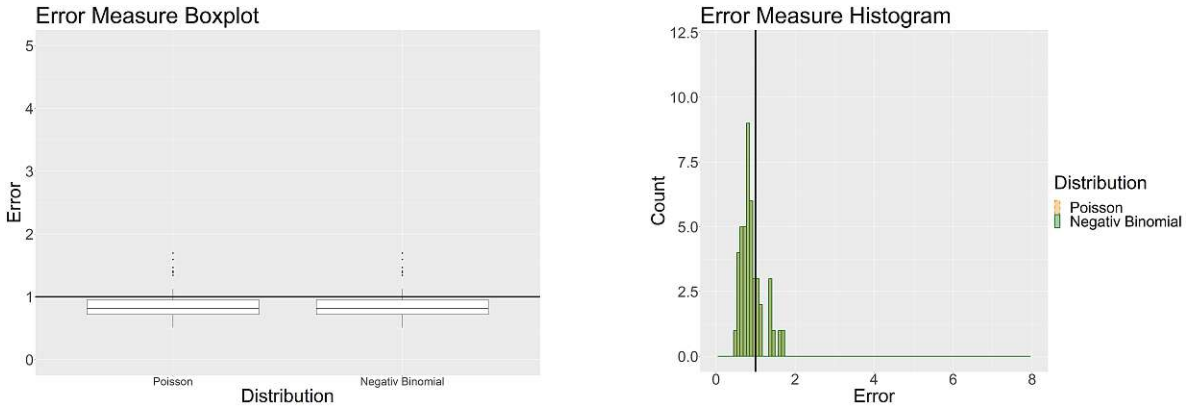
Figure 4.10: Comparison of different window shapes

4.5.3 INGARCH Specifications Results

Next we will investigate the INGARCH specific options. As before, we use the standard settings for the INGARCH(1,1) model and always vary one parameter.

Distribution

As mentioned in Section 4.1.2 we can replace the Poisson distribution with a Negative Binomial Distribution in Equation (2.3). The results are shown in Figure 4.11.



(a) Boxplot for different distributions

(b) Histogram for different distributions

Figure 4.11: Comparison of different distributions

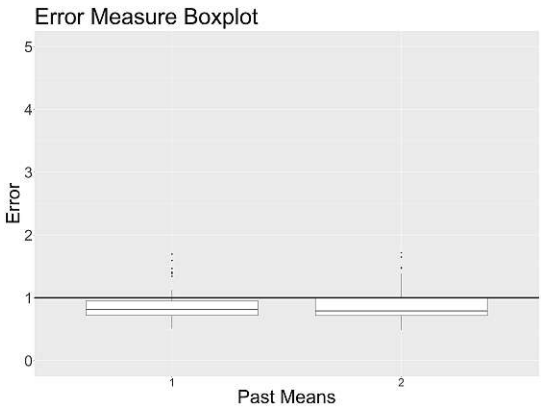
As we can see, we get the exactly the same results for both distributions. However, as mentioned in Section 2.2, we round the predicted conditional mean to the next integer. Hence, we could get slightly different results for the different distributions. Nevertheless, the difference is still negligible.

Number of Past Means and Observations

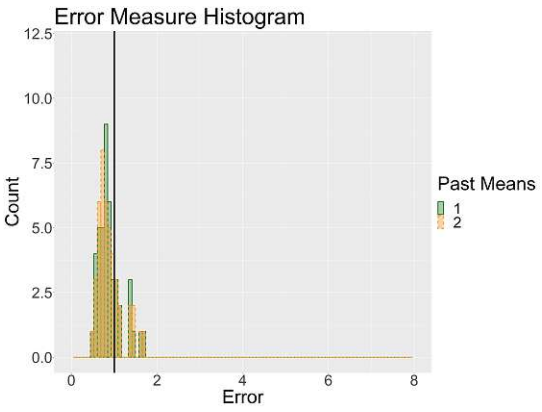
The order in the INGARCH(p,q) model is another parameter which can be chosen. For simplicities sake, we only compare our INGARCH(1,1) with an INGARCH(1,2) and an INGARCH(2,1) model. However, further models could be tried out and compared.

In Figure 4.12 we compare the INGARCH(1,1) model (red) with the INGARCH(1,2) model (blue). We can see that the performance is very similar. Hence we prefer the smaller model.

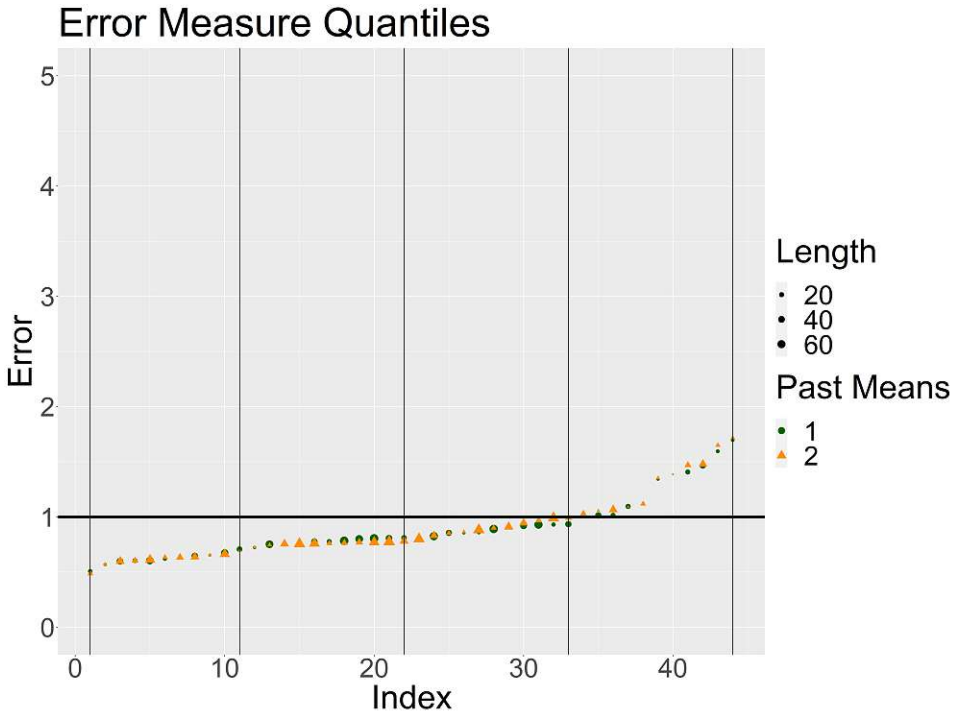
In Figure 4.13 we compare the INGARCH(1,1) (red) model with the INGARCH(2,1) (blue) model. Again the performance is very similar in general.



(a) Boxplot for a varying number of past means



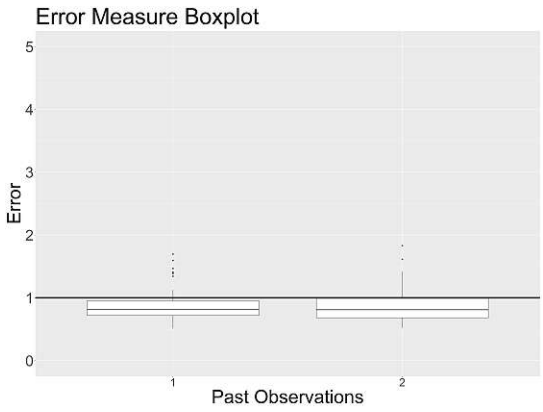
(b) Histogram for a varying number of past means



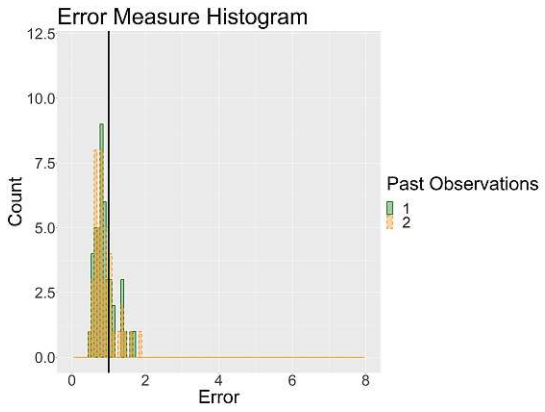
(c) Quantiles for a varying number of past means

Figure 4.12: Comparison of a varying number of past means

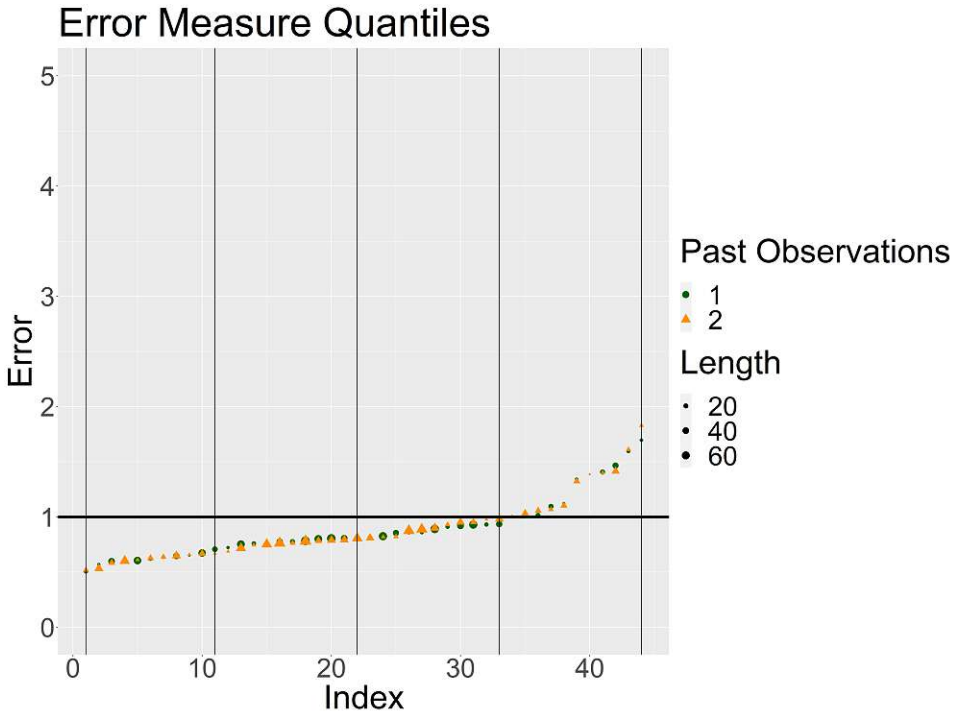
As we see, there is not much difference between the INGARCH(1,1), INGARCH(2,1) and INGARCH(1,2) model. One could compare the AIC or some other measure for the different models and base their choice on that. However, this is not further explored here and hence the INGARCH(1,1) model is taken because it is the smallest.



(a) Boxplot for a varying number of past observations



(b) Histogram for a varying number of past observations



(c) Quantiles for a varying number of past observations

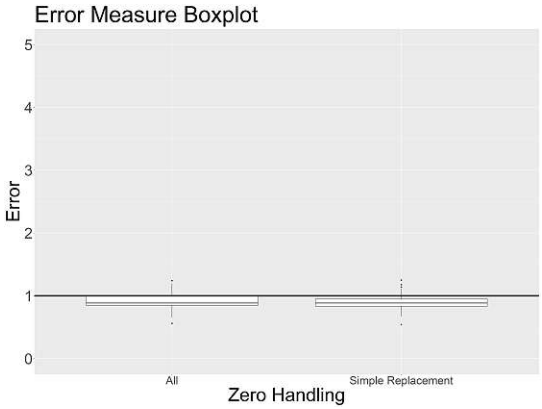
Figure 4.13: Comparison of a varying number of past observations

4.5.4 CoDA Specifications Results

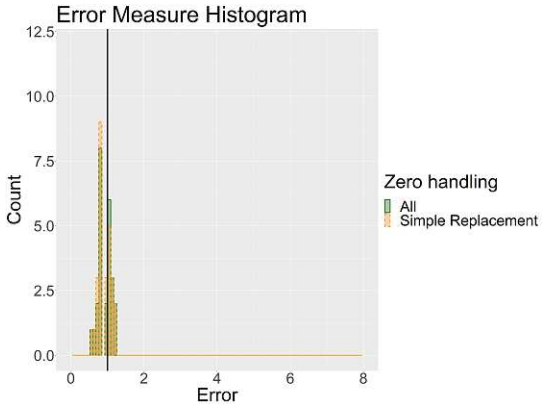
Last, we will compare different CoDA specifications as mentioned in Section 4.1.1. Like above, we choose one standard model for comparison and always only change one setting. For CoDA our standard model uses extending windows, the full history $h = 1$, an initial window length of $w = 0.5$, use the simple replacement strategy with $\delta = 0.1$, no $\mathcal{T}$-spaces, and the one-vs-all method.

4.5.5 Zero Handling

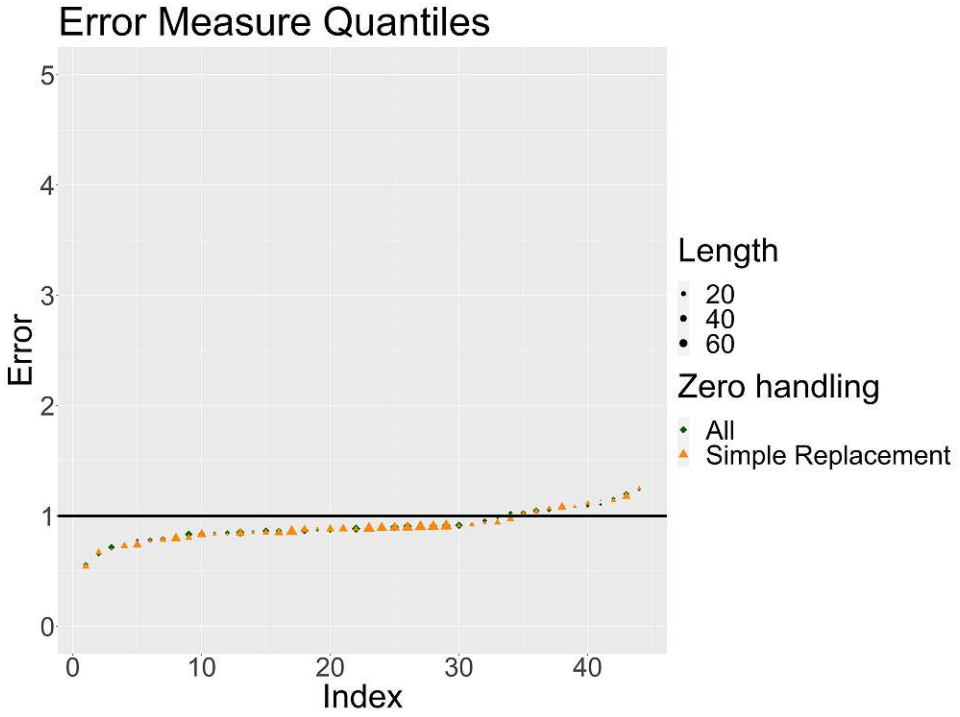
First, we compare the different options of handling zeros as explained in Section 4.1.1. The results are shown in Figure 4.14. It seems that the simple replacement strategy with $\delta_j = 0.1, \forall j$ results in marginally better performance.



(a) Boxplot for different zero handling methods



(b) Histogram for different zero handling methods



(c) Quantiles for different zero handling methods

Figure 4.14: Comparison of different zero handling methods

For the simple replacement strategy, one can also vary the parameter δ. In Figure 4.15 we plotted the results for $\delta = 0.01, 0.1, 0.5$. While the difference does not seem big

at first, when we calculate the error measure only for category 4, the category with most zeros, we can see a drastic rise in performance, see Figure 4.16. While we seem to get better results for smaller values of δ, one should remember, that CoDA cannot predict actual zeros, but instead the predictions get rounded to the nearest integer.

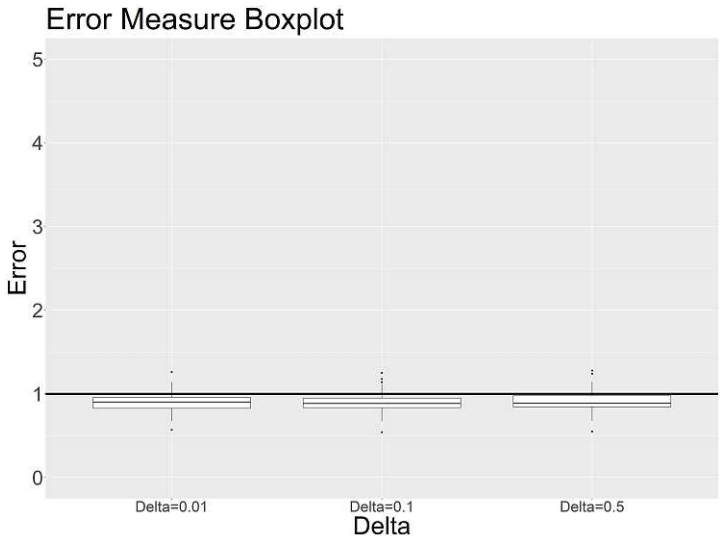


Figure 4.15: Error Measure for all categories

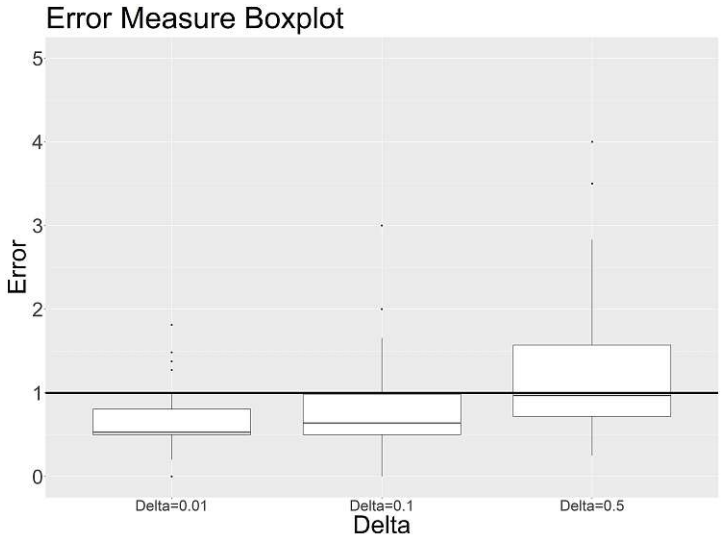


Figure 4.16: Error Measure for category 4

To further investigate the differences, we look in detail at the time series with the highest error measures for $\delta = 0.5$. Fridge 100402 is the fridge with the highest error and shown in Figure 4.17. We can see that for $\delta = 0.5$ in category 4, the predictions

stay at 1, even after a repeated amount of zero values. With the smaller δ-values on the other hand, CoDA starts to predict zero values after one or two time points. While the absolute difference is only one, the error measure is so high because the naive random walk model predicts all values correctly as zero and therefore the nominator in Equation (4.7) is theoretically zero. In practice, we implemented a fail-safe and set the nominator to 1e-6 to avoid division by 0.

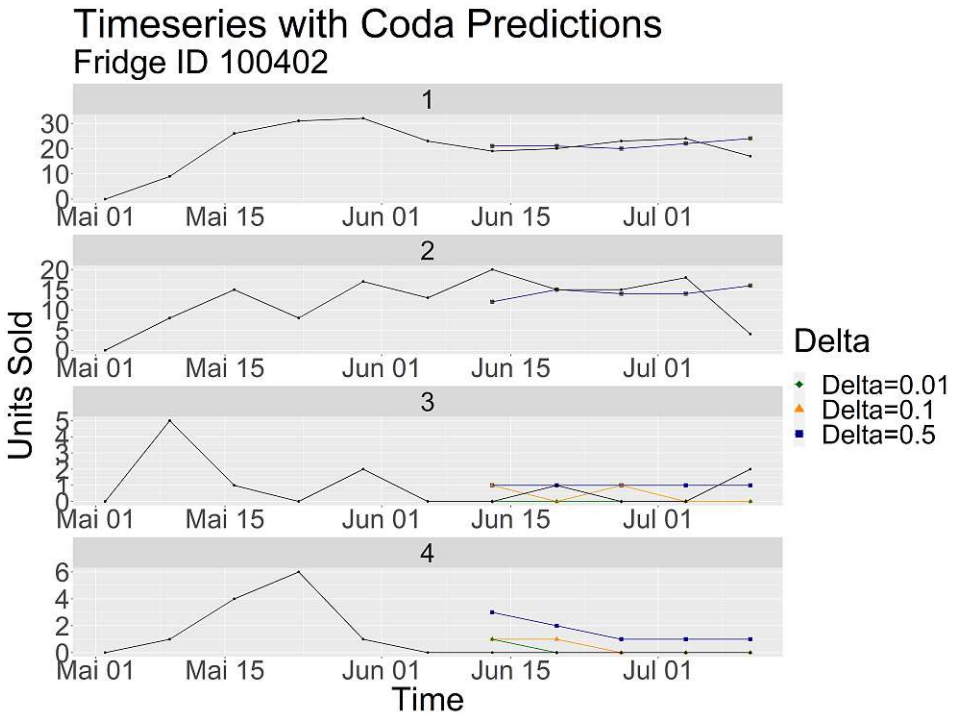


Figure 4.17: Time series of fridge 100402

The second highest error for CoDA is for fridge 100403, shown in 4.18. Again, we only have zero values for category 4 and for $\delta = 0.5$, CoDA never predicts zero. The same reasoning as above can be used to explain the high error value.

The same thing happens for fridge 100191 in Figure 4.19. We have an excessive amount of zero values and if δ is too high, CoDA fails to predict the correct value.

One thing that stands out in these time series is, that for categories 1 and 2, the predicted values are the same for all three values of δ.

4.5.6 $\mathcal{T}$-spaces

Next we compare CoDA for $\mathcal{T}-$Spaces. The results are shown in 4.20. It seems that using no $\mathcal{T}-$Spaces result in slightly better results. Especially for shorter time series

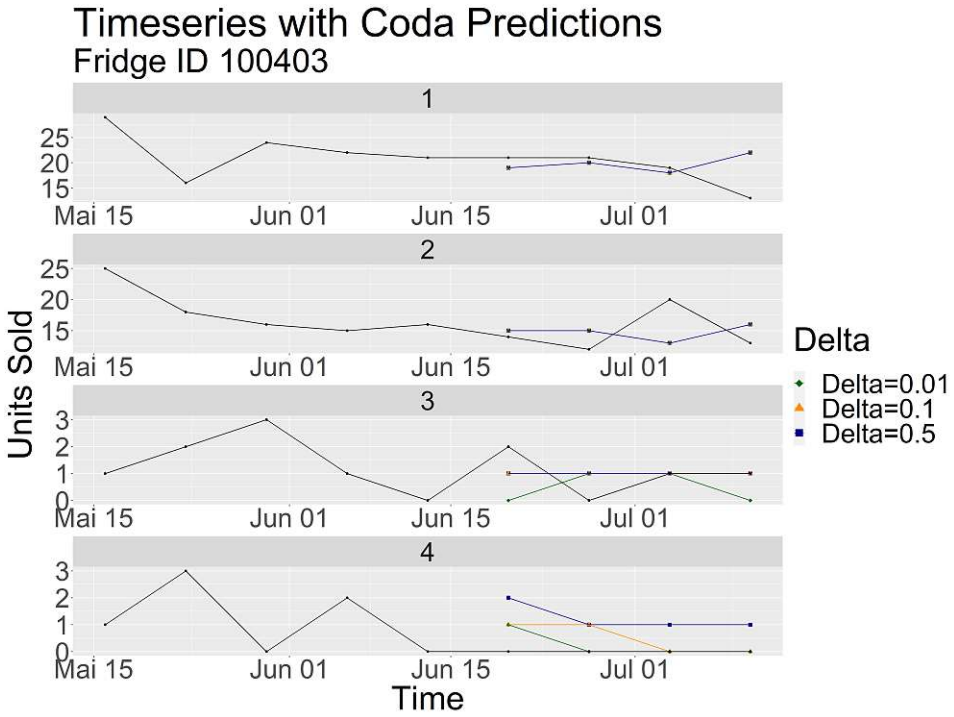


Figure 4.18: Time series of fridge 100403

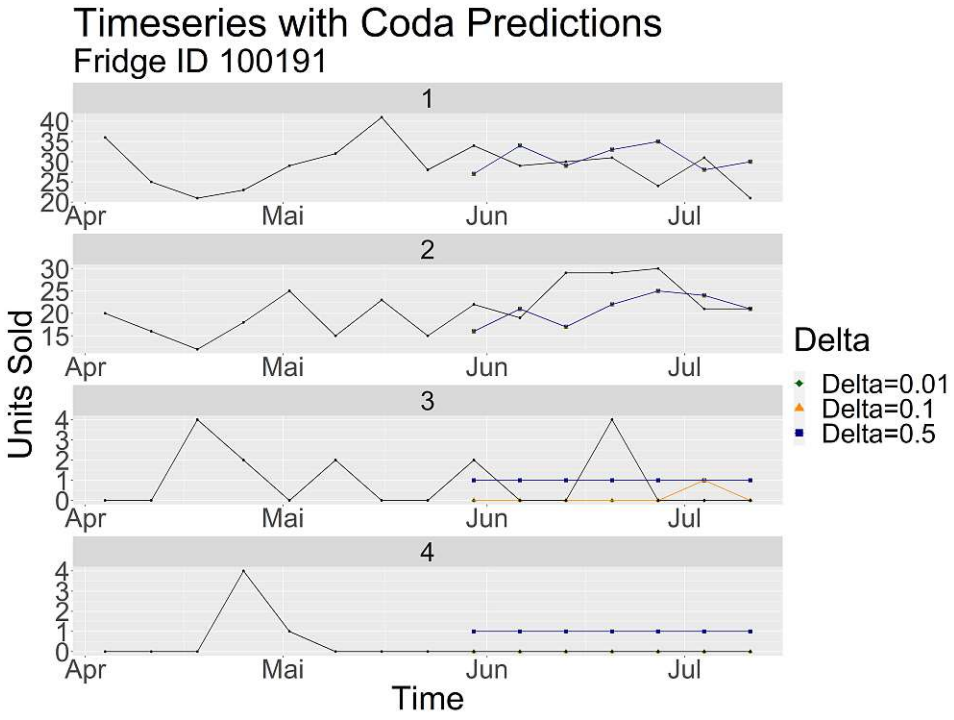
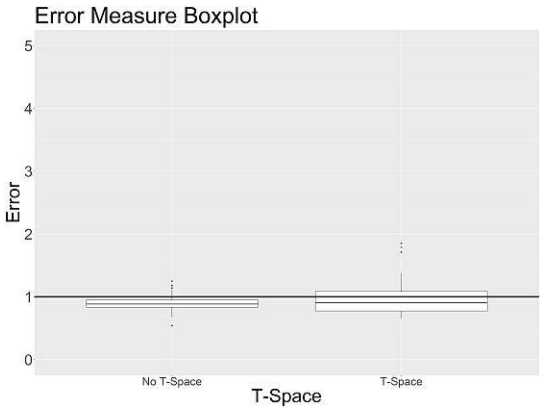
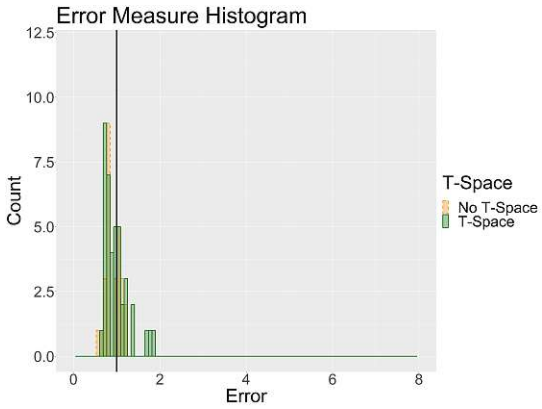


Figure 4.19: Time series of fridge 100191

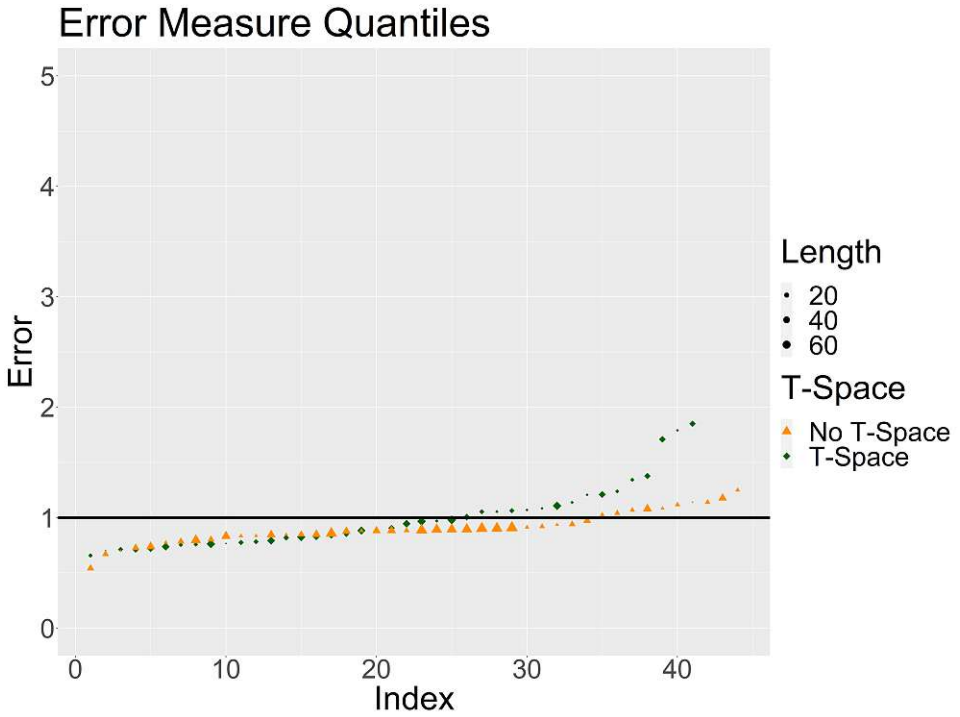
using no $\mathcal{T}-$Spaces returns better results. This can be seen in Figure 4.20c



(a) Boxplot for $\mathcal{T}-$Spaces



(b) Histogram for $\mathcal{T}-$Spaces



(c) Quantiles for $\mathcal{T}-$Spaces

Figure 4.20: Comparison of CoDA with and without $\mathcal{T}-$Spaces

To further investigate the reason of this difference in performance, we picked out the two time series with the highest error. Fridge 100321 has the highest error. Its time series can be seen in Figure 4.21. As we can see, CoDA with $\mathcal{T}$-Spaces performs worse for category 1 and 2. However, the time series is also short by nature with only 14 recorded points in time.

The second highest error measure has fridge 20, shown in 4.22. Again, the problem lies

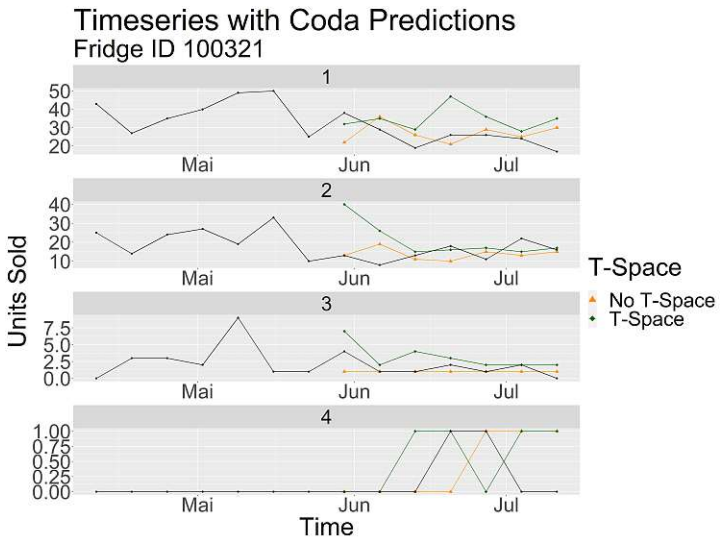


Figure 4.21: Time series of fridge 100321

in category 1 and 2. Especially for category 1, CoDA with $\mathcal{T}$-Space seems to continuously underestimate the true values. For category 3 and 4, both settings have very similar results.

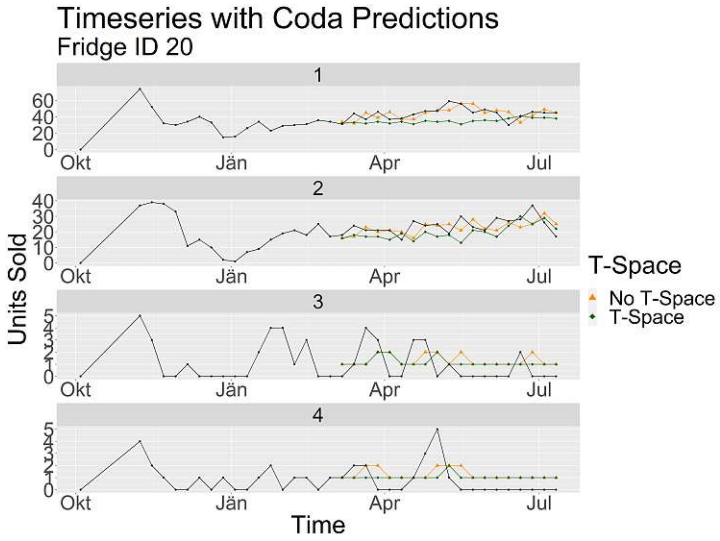
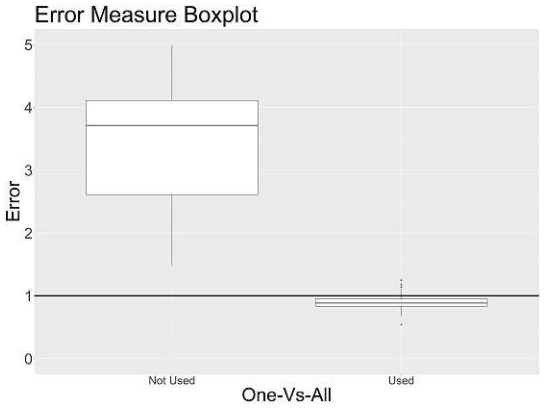


Figure 4.22: Time series of fridge 20

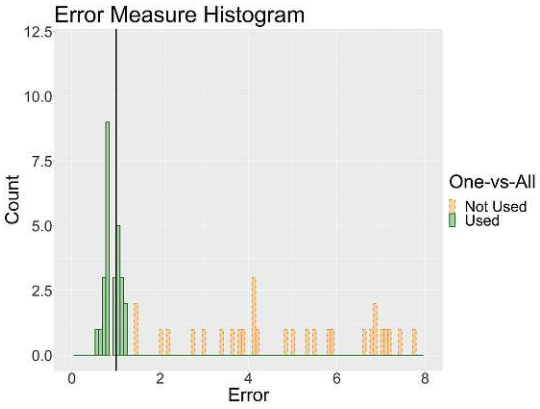
4.5.7 One-vs-All Method

Now we analyse the one-vs-all method. Figure 4.23 shows the results. We can clearly see, that the one-vs-all method performs better over all time series. This difference is

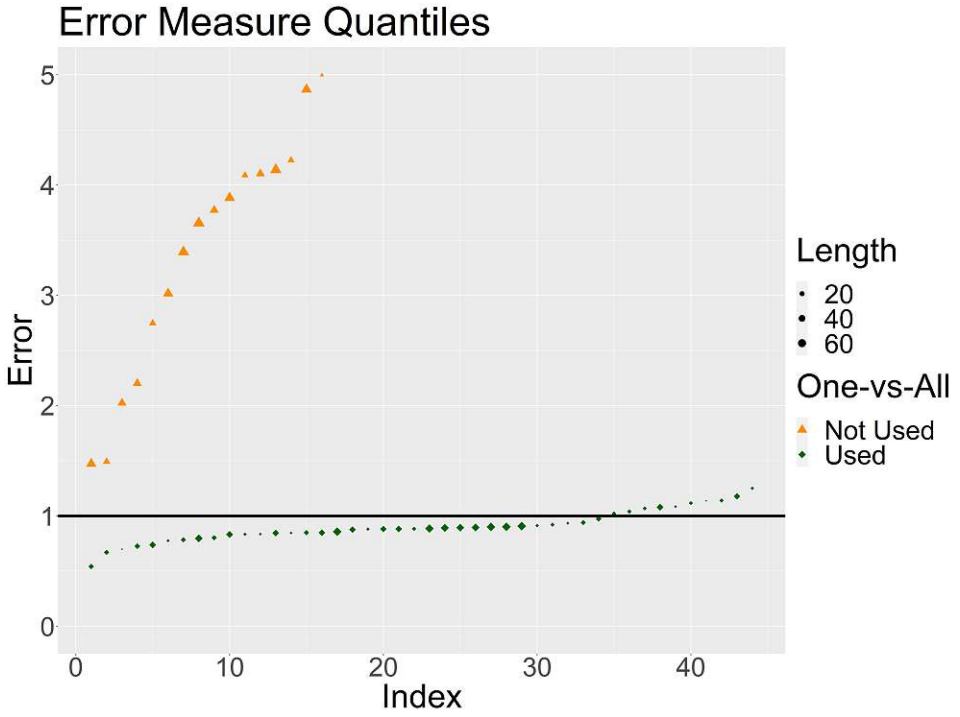
highlighted in Figure 4.23c.



(a) Boxplot for One-vs-All



(b) Histogram for One-vs-All



(c) Quantiles for One-vs-All

Figure 4.23: Comparison of CoDA with and without One-vs-All

5 Conclusion

In this thesis, we compare multiple models for multivariate count data time series with an excessive amount of zeros with the goal of finding the optimal model for predicting future values. A special focus lies in the integer-valued generalized autoregressive conditional heteroskedasticity model of order (p,q) (INGARCH(p,q) model) and the compositional data analysis (CoDA) model. The other models include a zero-inflated model (ZIM) and an integer-valued autoregressive model of order p (INAR(p) model). Since this thesis is being carried out as part of a bigger project at the Technical University of Austria in cooperation with Schrankerl GmbH, we are able to test our models on real world data and compare them with the model currently in use.

The current model used for forecasting is the naive random walk model. While in general, all tested models outperform the current model and deliver a similar output, they come with their respective advantages and disadvantages.

The INGARCH model considers the discrete nature of our data, but ignores both, the multivariate structure of it and the appearance of an unusual amount of zeros. While there exists a multivariate version of the INGARCH model, to our knowledge, there is no software implementation in R. The fitting of a multivariate INGARCH model and assuming the data to be ZIP distributed, can be grounds for further investigations.

The CoDA model fits a multivariate model and sees our data as a compositional time series. While it also neglects the fact that we have integer-valued data, the biggest issue with CoDa is, that the data must not include zeros. While there exist various methods to handle zero values as presented in Section 3.6, the handling of essential zeros, which is what we have, still remains problematic. We opt for the suggested data amalgamation and the simple replacement strategy because of their simplicity but a better way of handling zeros could be worth future research.

The INAR model has similar assumptions as the INGARCH model. It accounts for the discrete nature of our data, but neglects its multivariate structure and the amount of excessive zeros. The excessive zeros seem to be troublesome for the INAR model in general, since it performs the worst out of all models for time series with many zeros.

The zero-inflated model is an intriguing model. It considers both, the excessive amounts of zeros, and the discrete nature of our data and only ignores the multivariate structure of it. However, while we do have an excessive amount of zeros, we do not necessarily have zeros in every category of the time series and especially not if we only use parts of them. But since the zero-inflated model needs to have at least one zero in the sample, it cannot be fit on samples with no zeros present. This restricts us to fit the ZIM only on the categories with the most zeros present.

While we did some small tuning and compared different parameter settings for our models, we did not conduct an extensive analysis. Using some common model selection criteria like the Akaike information criteria (AIC) or Bayesian information criteria (BIC) are areas for further work. Another possible extension is the analysis of time series specific characteristics like seasonality and stationarity. Especially the test for seasonality could be interesting since many offices are emptier during holiday season and hence there are less possible costumers than usual.

Although we tried out many different models and conducted a literature review, it turns out to be difficult to find a model which takes the three major characteristics of our data, integer-valued, multivariate and excessive amount of zeros, into account. While all the models outperform the naive random walk model, it would be interesting to find a model which takes all three major characteristics of our data into account. In addition, the development of a multivariate INGARCH software package or the theoretical extension of ZIM to multivariate data are points for future research.

6 Summary

While multivariate count data time series with an excessive amount of zeros is a frequently encountered real-world problem, there is yet a clear way to handle them. This thesis is part of a bigger project carried out at the Technical University of Vienna in cooperation with the company Schrankerl GmbH. The company operates food vending machines in offices, which are restocked on a weekly basis. Since non-sold food gets disposed of, the company is in search of a model which can predict the amount sold in the upcoming week. For this, we compare various approaches with a focus on an integer-valued generalized autoregressive conditional heteroskedasticity model of order (p,q) (INGARCH(p,q) model) and a compositional data analysis (CoDA) model. Other investigated models include a zero-inflated model (ZIM) and an integer-valued autoregressive model of order p (INAR(p) model). In the first half of the thesis, we provide the mathematical background for these models. First, for the count time series models in Chapter 2 and later, for the CoDA model in Chapter 3. In the second half, we compare these models on a real-world data set provided to us by the company. This is done in Chapter 4. We investigate tuning options for our models in Section 4.1 and for comparison across different time series and the currently employed model, we introduce an error measure in Section 4.2. We conduct our analysis in the statistical software R and provide a handbook for our code as well as an overview of all used packages. In Section 4.5 we present the results of our analysis. Future extensions and possible further research options are mentioned in Chapter 5. We have shown that our models outperform the currently employed model, but come with their respective advantages and disadvantages. Since, to our knowledge, there exists currently no model which takes all three characteristics, the multivariate structure, the integer nature, and the excessive amount of zeros, into account, the development of such a model poses an interesting basis for future research.

Bibliography

Aitchison, J. (1986). *The Statistical Analysis of Compositional Data*. GBR: Chapman & Hall, Ltd. ISBN: 0412280604.

Aitchison, J. and J. W. Kay (2003). 'Possible solution of some essential zero problems in compositional data analysis'. In: *Proceedings of CODAWORK'03, The First Compositional Data Analysis Workshop, October 15-17, University of Girona (Spain)*. Ed. by S. Thió-Henestrosa and J. A. Martín Fernández.

Alzaid, A. A. and M. Al-Osh (1990). 'An integer-valued pth-order autoregressive structure (INAR(p)) process'. *Journal of Applied Probability* 27.2, pp. 314–324.

Bera, A. K. and M. L. Higgins (1993). 'ARCH models: properties, estimation and testing'. *Journal of Economic Surveys* 7.4, pp. 305–366. DOI: https://doi.org/10.1111/j.1467-6419.1993.tb00170.x.

Berndt, E. et al. (Jan. 1974). 'Estimation and inference in non-linear structural models'. *Annals of Economic and Social Measurement* 4, pp. 653–665.

Biswas, A. and P. X.-K. Song (2009). 'Discrete-valued ARMA processes'. *Statistics & Probability Letters* 79.17, pp. 1884–1889. DOI: https://doi.org/10.1016/j.spl.2009.05.025.

Bollerslev, T. (1986). 'Generalized autoregressive conditional heteroskedasticity'. *Journal of Econometrics* 31.3, pp. 307–327. DOI: https://doi.org/10.1016/0304-4076(86)90063-1.

Boogaart, K. G. van den, R. Tolosana-Delgado, and M. Bren (2006). 'Concepts for handling of zeros and missing values in compositional data'. In: *Proceedings of the IAMG'2006 Annual Conference on "Quantitative Geology from multiple sources", September 2006, Liege, Belgium, S07-01, 4pages.*

Brännäs, K. (Jan. 1993). 'Estimation and testing in integer-valued AR(1) models'. *Umeå Economic Studies.*

Chang, T. J., J. Delleur, and M. Kavvas (1987). 'Application of discrete autoregressive moving average models for estimation of daily runoff'. *Journal of Hydrology* 91.1, pp. 119–135. DOI: https://doi.org/10.1016/0022-1694(87)90132-6.

Christou, V. and K. Fokianos (2014). 'Quasi-likelihood inference for negative binomial time series models'. *Journal of Time Series Analysis* 35.1, pp. 55–78. DOI: https://doi.org/10.1111/jtsa.12050.

Cui, Y. and F. Zhu (June 2018). 'A new bivariate integer-valued GARCH model allowing for negative cross-correlation'. *TEST: An Official Journal of the Spanish Society of Statistics and Operations Research* 27.2, pp. 428–452. DOI: 10.1007/s11749-017-0552-4.

Deistler, M. and W. Scherrer (Jan. 2018). *Modelle der Zeitreihenanalyse*. Springer International Publishing. DOI: 10.1007/978-3-319-68664-6.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). 'Maximum likelihood from incomplete data via the EM algorithm'. *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1, pp. 1–22. DOI: https://doi.org/10.1111/j.2517-6161.1977.tb01600.x.

Douc, R., P. Doukhan, and E. Moulines (2013). 'Ergodicity of observation-driven time series models and consistency of the maximum likelihood estimator'. *Stochastic Processes and their Applications* 123.7. A Special Issue on the Occasion of the 2013 International Year of Statistics, pp. 2620–2647. DOI: https://doi.org/10.1016/j.spa.2013.04.010.

Doukhan, P., K. Fokianos, and D. Tjøstheim (2012). 'On weak dependence conditions for poisson autoregressions'. *Statistics & Probability Letters* 82.5, pp. 942–948. DOI: https://doi.org/10.1016/j.spl.2012.01.015.

Egozcue, J. J. and V. Pawlowsky-Glahn (Oct. 2005). 'Groups of parts and their balances in compositional data analysis'. *Mathematical Geology* 37, pp. 795–828. DOI: 10.1007/s11004-005-7381-9.

Egozcue, J. J., V. Pawlowsky-Glahn, et al. (Apr. 2003). 'Isometric logratio transformations for compositional data analysis'. *Mathematical Geology* 35, pp. 279–300. DOI: 10.1023/A:1023818214614.

Engle, R. F. (1982). 'Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation'. *Econometrica* 50.4, pp. 987–1007.

Ferland, R., A. Latour, and D. Oraichi (2006). 'Integer-valued GARCH process'. *Journal of Time Series Analysis* 27.6, pp. 923–942. DOI: https://doi.org/10.1111/j.1467-9892.2006.00496.x.

Filzmoser, P. and K. Hron (2020). '2.30 - Compositional Data Analysis in Chemometrics'. *Comprehensive Chemometrics (Second Edition)*. Ed. by S. Brown, R. Tauler, and B.

Walczak. Second Edition. Oxford: Elsevier, pp. 641–662. ISBN: 978-0-444-64166-3. DOI: https://doi.org/10.1016/B978-0-12-409547-2.14591-3.

Filzmoser, P., K. Hron, and M. Templ (2018). *Applied Compositional Data Analysis. With Worked Examples in R*. Springer Series in Statistics. Springer International Publishing. Springer Nature Switzerland AG, Cham, Switzerland. ISBN: 978-3-319-96420-1.

Fokianos, K. (2021). 'Multivariate count time series modelling'. *Econometrics and Statistics*. DOI: https://doi.org/10.1016/j.ecosta.2021.11.006.

Fokianos, K., A. Rahbek, and D. Tjøstheim (2009). 'Poisson autoregression'. *Journal of the American Statistical Association* 104.488, pp. 1430–1439. DOI: 10.1198/jasa.2009.tm08270.

Fokianos, K., B. Støve, et al. (2020). 'Multivariate count autoregression'. *Bernoulli* 26.1, pp. 471–499. DOI: 10.3150/19-BEJ1132.

Fokianos, K. and D. Tjøstheim (2011). 'Log-linear poisson autoregression'. *Journal of Multivariate Analysis* 102.3, pp. 563–578. DOI: https://doi.org/10.1016/j.jmva.2010.11.002.

Freeland, R. and B. McCabe (Feb. 2004). 'Forecasting discrete valued low count time series'. *International Journal of Forecasting* 20, pp. 427–434. DOI: 10.1016/S0169-2070(03)00014-1.

Garay, A. M. et al. (2022). 'First-Order Integer Valued AR Processes with Zero-Inflated Innovations'. In: *Nonstationary Systems: Theory and Applications*. Ed. by F. Chaari et al. Cham: Springer International Publishing, pp. 19–40. ISBN: 978-3-030-82110-4.

Hamilton, J. D. (Apr. 1994). *Time Series Analysis*. ISBN: 978-0-691-04289-3. DOI: 10.1604/9780691042893.

Harvey, A. C. and C. Fernandes (1989). 'Time series models for count or qualitative observations'. *Journal of Business & Economic Statistics* 7.4, pp. 407–417.

Heinen, A. (July 2003). *Modelling Time Series Count Data: An Autoregressive Conditional Poisson Model*. MPRA Paper 8113. University Library of Munich, Germany. DOI: http://dx.doi.org/10.2139/ssrn.1117187.

Hong, Y. and Y.-J. Lee (2017). 'A general approach to testing volatility models in time series'. *Journal of Management Science and Engineering* 2.1, pp. 1–33. DOI: https://doi.org/10.3724/SP.J.1383.201001.

Jin-Guan, D. and L. Yuan (1991). 'The integer-valued autoregressive (INAR(p)) model'. *Journal of Time Series Analysis* 12.2, pp. 129–142. DOI: https://doi.org/10.1111/j.1467-9892.1991.tb00073.x.

Kreiß, J. and G. Neuhaus (2006). *Einführung in die Zeitreihenanalyse*. Statistik und ihre Anwendungen. Springer Berlin Heidelberg. ISBN: 9783540335719.

Kynčlová, P., P. Filzmoser, and K. Hron (2015). 'Modeling compositional time series with vector autoregressive models'. *Journal of Forecasting* 34.4, pp. 303–314. DOI: https://doi.org/10.1002/for.2336.

Lambert, D. (1992). 'Zero-inflated poisson regression, with an application to defects in manufacturing'. *Technometrics* 34.1, pp. 1–14.

Lee, S., D. Kim, and B. Kim (2023). 'Modeling and inference for multivariate time series of counts based on the INGARCH scheme'. *Computational Statistics & Data Analysis* 177, p. 107579. DOI: https://doi.org/10.1016/j.csda.2022.107579.

Leininger, T. J. et al. (2013). 'Spatial regression modeling for compositional data with many zeros'. *Journal of Agricultural, Biological, and Environmental Statistics* 18.3, pp. 314–334.

Liboschik, T. (July 2016). 'Modeling count time series following generalized linear models'. Dissertation. Dortmund,Germany: TU Dortmund University.

Liboschik, T., K. Fokianos, and R. Fried (2017). 'tscount: An R package for analysis of count time series following generalized linear models'. *Journal of Statistical Software* 82.5, pp. 1–51. DOI: 10.18637/jss.v082.i05.

Liboschik, T., R. Fried, et al. (2020). *tscount: Analysis of Count Time Series*. R package version 1.4.3.

Liu, H. (2012). *Some models for time series of counts*. Columbia University. DOI: 10.7916/D8Z325SW.

Lubbe, S., P. Filzmoser, and M. Templ (2021). 'Comparison of zero replacement strategies for compositional data with large numbers of zeros'. *Chemometrics and Intelligent Laboratory Systems* 210, p. 104248. DOI: https://doi.org/10.1016/j.chemolab.2021.104248.

Lütkepohl, H. (2007). 'Vector Autoregressive Models'. *Palgrave Handbook of Econometrics: Econometric Theory, Volume 1 Econometric Theory*. Palgrave MacMillan, pp. 477–510.

MacDonald, I. and W. Zucchini (1997). *Hidden Markov and Other Models for Discrete-valued Time Series*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability. Taylor & Francis. ISBN: 9780412558504.

Martín-Fernández, J. A., C. Barceló-Vidal, and V. Pawlowsky-Glahn (2003). 'Dealing with zeros and missing values in compositional data sets using nonparametric imputation'. *Mathematical Geosciences* 35.3, pp. 253–278. DOI: 10.1023/A:1023866030544.

Martín-Fernández, J.-A. et al. (2015). 'Bayesian-multiplicative treatment of count zeros in compositional data sets'. *Statistical Modelling* 15.2, pp. 134–158. DOI: 10.1177/1471082X14535524.

Müller, K. (2020). *here: A Simpler Way to Find Your Files*. R package version 1.0.1.

Nau, R. (Apr. 2014). *Notes on the random walk model*.

Palarea-Albaladejo, J. and J. A. Martín-Fernández (2015). 'zCompositions — R package for multivariate imputation of left-censored data under a compositional approach'. *Chemometrics and Intelligent Laboratory Systems* 143, pp. 85–96. DOI: https://doi.org/10.1016/j.chemolab.2015.02.019.

Pawlowsky-Glahn, V., J. J. Egozcue, and D. Lovell (June 2013). 'The product space $\mathcal{T}$ (tools for compositional data with a total)'. *Proceedings of the 5th International Workshop on Compositional Data Analysis*, pp. 143–152.

Pfaff, B. (2008a). *Analysis of Integrated and Cointegrated Time Series with R*. Second. ISBN 0-387-27960-1. New York: Springer.

Pfaff, B. (2008b). 'VAR, SVAR and SVEC models: implementation within R package vars'. *Journal of Statistical Software* 27.4, pp. 1–32. DOI: 10.18637/jss.v027.i04.

R Core Team (2022). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria.

Robitzsch, A. and S. Grund (2023). *miceadds: Some Additional Multiple Imputation Functions, Especially for 'mice'*. R package version 3.16-18.

Schwaiger, A. (21st July 2023). *Master Thesis Software*. URL: https://github.com/AlexSch2/Master-Thesis.

Silva, I. et al. (Dec. 2005). 'Replicated INAR(1) processes'. *Methodology And Computing In Applied Probability* 7, pp. 517–542. DOI: 10.1007/s11009-005-5006-x.

Silva, M. and V. Oliveira (Jan. 2005). 'Difference equations for the higher order moments and cumulants of the INAR(p) model'. *Journal of Time Series Analysis* 26, pp. 17–36. DOI: 10.1111/j.1467-9892.2005.00388.x.

Silva, N., I. Pereira, and M. Silva (May 2009). 'Forecasting in INAR (1) model'. *REVSTAT – Statistical Journal Volume* 7, pp. 119–134. DOI: 10.57805/revstat.v7i1.77.

Templ, M., K. Hron, and P. Filzmoser (2011). *robCompositions: an R-package for robust statistical analysis of compositional data*. John Wiley and Sons, pp. 341–355. ISBN: 978-0-470-71135-4.

van den Boogaart, K. G., R. Tolosana-Delgado, and M. Bren (2023). *compositions: Compositional Data Analysis*. R package version 2.0-6.

Wickham, H. et al. (2019). 'Welcome to the tidyverse'. *Journal of Open Source Software* 4.43, p. 1686. DOI: 10.21105/joss.01686.

Woodard, D. B., D. S. Matteson, and S. G. Henderson (2011). 'Stationarity of generalized autoregressive moving average models'. *Electronic Journal of Statistics* 5.none, pp. 800–828. DOI: 10.1214/11-EJS627.

Yang, M. (2012). 'Statistical models for count time series with excess zeros'. eng. PhD thesis. University of Iowa, viii, 74 pages.

Zeger, S. L. (1988). 'A regression model for time series of counts'. *Biometrika* 75.4, pp. 621–629.

Zeileis, A., C. Kleiber, and S. Jackman (2008). 'Regression models for count data in R'. *Journal of Statistical Software* 27.8, pp. 1–25. DOI: 10.18637/jss.v027.i08.

Zhu, F. (2012). 'Zero-inflated poisson and negative binomial integer-valued GARCH models'. *Journal of Statistical Planning and Inference* 142.4, pp. 826–839. DOI: https://doi.org/10.1016/j.jspi.2011.10.002.

Zivot, E. (2009). 'Practical Issues in the Analysis of Univariate GARCH Models'. In: *Handbook of Financial Time Series*. Ed. by T. Mikosch et al. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 113–155. ISBN: 978-3-540-71297-8. DOI: 10.1007/978-3-540-71297-8_5.

List of Figures

4.1 Time series for two fridges 41
4.2 Time series with the CoDA model 42
4.3 Time series with the INGARCH model 42
4.4 Time series with both models 43
4.5 Time series with both models and their prediction intervals 44
4.6 Comparison of the different models for all main categories 56
4.7 Comparison of the different models for main categories 3 and 4 57
4.8 Comparison of different h 58
4.9 Comparison of different w 59
4.10 Comparison of different window shapes 60
4.11 Comparison of different distributions 61
4.12 Comparison of a varying number of past means 62
4.13 Comparison of a varying number of past observations 63
4.14 Comparison of different zero handling methods 64
4.15 Error Measure for all categories 65
4.16 Error Measure for category 4 65
4.17 Time series of fridge 100402 66
4.18 Time series of fridge 100403 67
4.19 Time series of fridge 100191 67
4.20 Comparison of CoDA with and without $\mathcal{T}-$Spaces 68
4.21 Time series of fridge 100321 69
4.22 Time series of fridge 20 69
4.23 Comparison of CoDA with and without One-vs-All 70

List of Tables

1.1 Example data 2
1.2 Example data aggregated on main category level 3