DIPLOMARBEIT

Titel

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 29th May 2023

Alexander Schwaiger (Verfasser) Peter Filzmoser (Betreuer)

Abstract

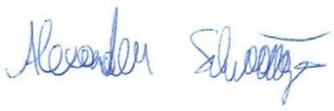
In this thesis, we analyse and compare two approaches for multivariate count data time series with an excessive amount of zeros. The first approach belongs to the class of generalised linear models (GLM) and fits a univariate integer valued generalized autoregressive conditional heteroskedasticity model 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 of 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. At last, we provide a handbook of our analysis in the statistical software R and present the used packages and functions.

Keywords: 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 29th May 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. 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

1. Introduction

1.1. Motivation

Multivariate count data is a reoccurring theme in real world applications. While there exist various methods among the classical statistical models to handle such data, there are less 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.

A company is operating numerous vending machines 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 off which result in monetary losses. The objective is to find a model to predict the amount they need 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 1.1 describes the whole time series available, while 1.2 describes the time series used. It holds $T \leq T_f$. In the following we will use 1.2 to indicated that we may only use a fraction of the whole time series. We will dive more into it 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 1.1 would then change to 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 2. In these sections 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 4.3 we explain the R-functions used and provide a guidebook.

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 [Lib16] and [Hei03] and a more detailed review can be found in [MMM97]. 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 over dispersion, 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 lose tractability. As an extension to the basic Markov chains models, Hidden Markov chains are proposed by [MMM97]. However, since there is no generally accepted way to determine the order of this model, 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 need to be estimated gets big quickly, especially if the order of the model is big.

Other common models for time series data are the ARMA models and there exists a discrete version with 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 [BS09]. While there have been various applications, for example in [CDK87], there seem to be difficulties in their estimation [Hei03].

State space models with conjugated priors are proposed by [HF89].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 are chosen in such a

way that its mean is constant but its variance is increasing. While there are ways proposed by [Zeg88] to handle overdispersion, these models have the weakness of needing further assumptions to handle zeros while also having more complicated model specifications [Hei03].

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 and to factor in additional, external information, one can decide to include a covariate vector. 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. We also introduce an extension of the INGARCH 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 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, which was introduced by [Lam92]. 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 detaisl can be found in 2.5

Another popular approach for count time series are the integer-valued autoregressive (INAR) models presented in section 2.9.1. 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 a 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 [Hei03].

The simple naive random walk 2.4 is the simplest and most basic approach not only for count data but time series in general. This is the model that is currently used for forecasting and is therefore ideal as a benchmark. We will use it to compare the performance of the other models directly to it with the help of a new error measure in 4.1.3.

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 over them for better comparison and clearness on why we choose the integer valued versions. However, we will not consider neither the GARCH nor the AR model in our analysis.

2.2. INGARCH Model

We construct the INGARCH(p,q) model as in [Lib16]. 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 a r-dimensional time varying covariate vector with $\mathbf{X}_{kt} = (X_{t1}^k, \ldots, X_{tr}^k)^T$. Let the conditional mean be $\lambda_{kt} = \mathbb{E}[Y_{kt}|\mathcal{F}_{k,t-1}]$ 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,

$$Y_{kt}|\mathcal{F}_{k,t-1} \sim P(\lambda_{kt}); \forall t \in \mathbb{N}, \tag{2.1}$$

$$\mathbb{E}[Y_{kt}|\mathcal{F}_{k,t-1}] = \lambda_{kt} = \beta_0 + \sum_{i=1}^{p} \beta_i Y_{k,t-i} + \sum_{j=1}^{q} \alpha_j \lambda_{k,t-j}, \tag{2.2}$$

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 add the covariate vector $\mathbf{X}_{kt}$

$$Y_{kt}|\mathcal{F}_{k,t-1} \sim P(\lambda_{kt}); \forall t \in \mathbb{N}, \tag{2.3}$$

$$\mathbb{E}[Y_{kt}|\mathcal{F}_{k,t-1}] = \lambda_{kt} = \beta_0 + \sum_{i=1}^{p} \beta_i Y_{k,t-i} + \sum_{j=1}^{q} \alpha_j \lambda_{k,t-j} + \boldsymbol{\eta}^T \mathbf{X}_{kt}, \tag{2.4}$$

where $\boldsymbol{\eta}$ is the parameter for the covariates such that $\boldsymbol{\eta}^T \mathbf{X}_{kt} \geq 0$. From the distributional assumption $Y_{kt}|\mathcal{F}_{k,t-1} \sim P(\lambda_{kt})$ it follows

$$p_{kt}(y; \boldsymbol{\theta}) = \mathbb{P}(Y_{kt} = y|\mathcal{F}_{k,t-1}) = \frac{\lambda_{kt}^y \exp(-\lambda_{kt})}{y!}, \ \ y \in \mathbb{N}_0. \tag{2.5}$$

Furthermore it can be shown that conditionally on the past history $\mathcal{F}_{k,t-1}$ the model is equidispersed, i.e. it holds $\lambda_{kt} = \mathbb{E}[Y_{kt}|\mathcal{F}_{k,t-1}] = \mathbb{V}[Y_{kt}|\mathcal{F}_{k,t-1}]$. However, unconditionally the model exhibits overdispersion. In that case it holds $\mathbb{E}[Y_{kt}] \leq \mathbb{V}[Y_{kt}]$ [Hei03].

Parameter Estimation and Forecasting

We summarise the estimation of the INGARCH(p,q) Model as described in [Lib16]. The model is estimated for each category $k = 1, \ldots, K$ separately.

The parameter space for the INGARCH(p,q) model with external effects 2.3 is given by

$$\Theta = \left\{ \boldsymbol{\theta} \in \mathbb{R}^{p+q+r+1} : \beta_0 > 0, \beta_1, \ldots, \beta_p, \alpha_1, \ldots, \alpha_q, \eta_1, \ldots, \eta_r \geq 0, \sum_{i=1}^{p} \beta_i + \sum_{j=1}^{q} \alpha_j < 1 \right\}. \tag{2.6}$$

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 [FLO06; FRT09; DFT12]. 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,

$$\ell_k(\boldsymbol{\theta}) = \sum_{t=1}^{T} \log p_{kt}(y_{kt}; \boldsymbol{\theta}) = \sum_{t=1}^{T} \left(y_{kt} \log(\lambda_{kt}(\boldsymbol{\theta})) - \lambda_{kt}(\boldsymbol{\theta}) \right). \tag{2.7}$$

where $p_{kt}(y_{kt}; \boldsymbol{\theta})$ is the probability density function defined in 2.5. The conditional mean is seen as a function $\lambda_{kt} : \Theta \to \mathbb{R}^+$. The conditional score function is given by,

$$S_{kT}(\boldsymbol{\theta}) = \frac{\partial \ell_k(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \sum_{t=1}^{T} \left(\frac{y_{kt}}{\lambda_{kt}(\boldsymbol{\theta})} - 1 \right) \frac{\partial \lambda_{kt}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}}. \tag{2.8}$$

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

$$G_{kT}(\theta) = \sum_{t=1}^{T} Cov\left(\frac{\partial \ell_k(\boldsymbol{\theta}; Y_{kt})}{\partial \boldsymbol{\theta}} \middle| \mathcal{F}_{k,t-1} \right) \tag{2.9}$$

$$= \sum_{t=1}^{T} \left(\frac{1}{\lambda_{kt}(\boldsymbol{\theta})} \right) \left(\frac{\partial \lambda_{kt}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right) \left(\frac{\partial \lambda_{kt}(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} \right)^T. \tag{2.10}$$

Finally, assuming that the quasi maximum likelihood estimator (QMLE) $\hat{\boldsymbol{\theta}}_T$ of $\boldsymbol{\theta}$ exists, it is the solution to

$$\hat{\boldsymbol{\theta}} := \hat{\boldsymbol{\theta}}_T = \arg\max_{\boldsymbol{\theta}\in\Theta}(\ell_k(\boldsymbol{\theta})). \tag{2.11}$$

The optimal one-step ahead forecast with regards to the mean squared error is the conditional expectation $\lambda_{k,t+1} = \mathbb{E}\left[Y_{k,t+1}|\mathcal{F}_{kt}\right]$. The h-step ahead prediction for $h > 1$ is calculated iteratively with the one step ahead predictions of $Y_{k,t+1}, Y_{k,t+2}, \ldots$ [Lib16].

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. Bivariate models have been proposed by [Liu12] and extended for example by [CZ18].

The authors in [Fok+20; Fok21] introduce and review the multivariate INGARCH model on the basis of a data generating process. Let $\boldsymbol{\lambda}_t = \mathbb{E}[\boldsymbol{Y}_t|\mathcal{F}_t]$ 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

$$Y_{kt}|\mathcal{F}_{t-1} \sim P(\lambda_{kt}), \tag{2.12}$$

$$\boldsymbol{\lambda}_t = \boldsymbol{d} + \boldsymbol{A}\boldsymbol{\lambda}_{t-1} + \boldsymbol{B}\boldsymbol{Y}_{t-1}. \tag{2.13}$$

Based on this, a joint distribution is constructed using a copula structure. After an additional transformation, the multivariate INGARCH model is

$$\boldsymbol{Y}_t = \boldsymbol{N}_t(\boldsymbol{\lambda}_t), \tag{2.14}$$

$$\boldsymbol{\lambda}_t = \boldsymbol{d} + \boldsymbol{A}\boldsymbol{\lambda}_{t-1} + \boldsymbol{B}\boldsymbol{Y}_{t-1}, \tag{2.15}$$

where $\{\boldsymbol{N}_t\}$ is a sequence of K-variate independent copula-Poisson processes that counts the number of events in $[0, \lambda_{1t}]\times, \ldots, \times[0, \lambda_{Kt}]$ [Fok+20].

Another approach is taken by [LKK23]. 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}

$$p_k(y|\nu) = \exp(\nu y - A_k(\nu))h_k(y), \quad y \in \mathbb{N}_0, \tag{2.16}$$

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 [LKK23]. The multivariate INGARCH model is then given for each $k = 1, \ldots, K$ by

$$Y_{kt}|\mathcal{F}_{t-1} \sim p_k(y|\nu_{kt}), \tag{2.17}$$

$$\boldsymbol{\lambda}_t := \mathbb{E}[\boldsymbol{Y}_t|\mathcal{F}_{t-1}] = f_\theta(\boldsymbol{\lambda}_{t-1}, \boldsymbol{Y}_{t-1}), \tag{2.18}$$

where $\mathcal{F}_{t-1}$ is the σ-field generated by $\{\boldsymbol{Y}_{t-1}, \boldsymbol{Y}_{t-2}, \ldots, B_k(\nu_{kt})\}$ with $B_k(\nu_{kt}) = \lambda_{kt}$, and f_θ is a non-negative function on $[0, \infty)^K \times \mathbb{N}_0^K$ [LKK23]. So for each component Y_{kt} a univariate INGARCH model is fit but the components are connected by the conditional mean process.

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 [82] in an economic context, model the variance conditional on past values. Let $\{Y_{kt} : t = 1, \ldots T; Y_{kt} \in \mathbb{N}_0\}_f$ be the univariate time series for category k for $k = 1, \ldots, K$ and fridge f and $\mathcal{F}_{k,t}$ be the information available at time t. Then the ARCH(1) model is given by [82]

$$Y_{kt}|\mathcal{F}_{k,t-1} \sim N(0, h_{kt}), \tag{2.19}$$

$$h_{kt} = a_0 + a_1 Y_{k,t-1}^2, \tag{2.20}$$

with $a_0 \geq 0$, $a_1 > 0$.

The variance function can be generally formulated as $h_{kt} = h(Y_{k,t-1}, \ldots, Y_{k,t-p}, \boldsymbol{a})$, where $\boldsymbol{a} \geq 0$ is the parameter vector with $a_p > 0$, and $p \in \mathbb{N}$ is the order of the ARCH process.

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 $a_0 > 0$,

$\boldsymbol{a} = (a_1, \ldots, a_p)^T \geq 0$ and $\boldsymbol{b} = (b_1, \ldots, b_q)^T \geq 0$ with $p, q \in \mathbb{N}$,$p \geq 0, q > 0$ is given by [Bol86]

$$Y_{kt}|\mathcal{F}_{k,t-1} \sim N(0, h_{kt}); \forall t \in \mathbb{N}, \tag{2.21}$$

$$\mathbb{V}[Y_{kt}|\mathcal{F}_{k,t-1}] = h_{kt} = a_0 + \sum_{i=1}^{p} a_i Y_{k,t-i}^2 + \sum_{j=1}^{q} b_j h_{k,t-j}; \forall t \in \mathbb{N}. \tag{2.22}$$

Other distributions than the normal distributions can be taken as well.

2.3.1. Parameter Estimation and Forecasting

Estimation of the parameters can be done with maximum likelihood and an iterative algorithm. The model is rewritten and the log-likelihood function then used and after differentiation with respect to its variance and mean parameters, the Berndt, Hall Hall and Hausman algorithm [Ber+74]is used to obtain the maximum likelihood estimates. Further details and assumptions can be found in [Bol86].

If one is interested in forecasting Y_{kt}, then the minimum mean squared one-step error forecast is $\mathbb{E}[Y_{k,t+h}|\mathcal{F}_{k,t}] = 0$ 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 for should be forecasted, the parameters are estimated and the known values are plugged in. For $h > 1$, h-step are computed recursively with plugging in the forecasts for $h-1, h-2, \ldots$ in the model [Ziv09].

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 [Bol86] the author suggests a Lagrange multiplier test and other popular tests include the Box–Pierce–Lung-type portmanteau tests and residual-based diagnostics [HL17]. The authors in [HL17] 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 and GARCH models then generalised this approach to model a more flexible lag structure [Bol86]. Since then, 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.

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 as well as its possible over dispersion 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 1-step difference between two values is i.i.d distributed with mean 0. Let $\{Y_{kt} : t = 1, \dots T; Y_{kt} \in \mathbb{N}_0\}_f$ be our univariate time series. Then the Naive Random Walk model is given as

$$Y_{k,t+1} = Y_{kt} + \epsilon_{kt}, \;\; k = 1, \dots, K \;, \tag{2.23}$$

where $\epsilon_{kt} \sim WN(\sigma^2)$ is a white noise process with variance $\sigma^2 \in \mathbb{R}_+$. It can be shown easily, that the optimal 1-step ahead forecast with regards to the mean squared error (MSE) is given by

$$\hat{Y}_{k,t+1} = Y_{kt}, \;\; k = 1, \dots, K, \tag{2.24}$$

where $\hat{Y}_{k,t+1}$ is the predicted value at time t. In other words, the predicted value is the last known value.

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 [Zhu12]

$$\mathbb{P}(Y = y) = \omega\delta_{y,0} + (1-\omega)\frac{\lambda^y \exp(-\lambda)}{y!}, \;\; y \in \mathbb{N}_0. \tag{2.25}$$

where $0 < \omega < 1$ 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 [Zhu12].

Now we can define the Zero-Inflated Poisson (ZIP) INGARCH(p,q) as

$$Y_{kt}|\mathcal{F}_{k,t-1} \sim ZIP(\lambda_{kt}, \omega_k); \forall t \in \mathbb{N}, \tag{2.26}$$

$$\mathbb{E}\left[Y_{kt}|\mathcal{F}_{k,t-1}\right] = \lambda_{kt} = \beta_0 + \sum_{i=1}^{p} \beta_i Y_{k,t-i} + \sum_{j=1}^{q} \alpha_j \lambda_{k,t-j}. \tag{2.27}$$

If $\omega = 0$ then we get the normal INGARCH(p,q) model discussed above. It can be shown that the conditional mean and variance are given by

$$\mathbb{E}[Y_{kt}|\mathcal{F}_{k,t-1}] = (1-\omega_k)\lambda_{kt}, \qquad \mathbb{V}[Y_{kt}|\mathcal{F}_{k,t-1}] = (1-\omega)\lambda_{kt}(1+\omega\lambda_{kt}), \tag{2.28}$$

which implies $\mathbb{V}[Y_{kt}|\mathcal{F}_{k,t-1}] > \mathbb{E}[Y_{kt}|\mathcal{F}_{k,t-1}]$ [Zhu12]. This means that model 2.26 can handle overdispersion in our data. More details about zero-inflated models and especially the zero-inflated INGARCH(p,q) model can be found in [Zhu12].

However, due to a lack of available R-packages for zero-inflated Poisson INGARCH models, we use a zero-inflated Poisson autoregressive model. We again assume that our data is conditionally $ZIP(\lambda_{kt}, \omega_{kt})$ distributed. For the parameters λ_{kt} and ω_{kt}, the ZIP autoregressive model is given by [Lam92]

$$\log(\lambda_{kt}) = \boldsymbol{B}_{k,t-1}^T \boldsymbol{\beta}, \tag{2.29}$$

$$\log\left(\frac{\omega_t}{1-\omega_{kt}}\right) = \boldsymbol{Z}_{k,t-1}^T \boldsymbol{\gamma}, \tag{2.30}$$

where $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_p)^T$ and $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_m)^T$ are the parameters to be estimated and the vectors $\boldsymbol{B}_{k,t-1}$ and $\boldsymbol{Z}_{k,t-1}$ are the explanatory covariates. In model 2.30 a logit link function has been used although, it can be replaced with other link functions like the probit or log link.

In our case we regress on the past values of our time series. In that case model 2.30 becomes

$$\log(\lambda_{kt}) = (1, Y_{k,t-1})^T \boldsymbol{\beta}, \tag{2.31}$$

$$\log\left(\frac{\omega_{kt}}{1-\omega_{kt}}\right) = 1 \cdot \gamma. \tag{2.32}$$

Hence we have $\boldsymbol{B}_{t-1}^T = (1, Y_{k,t-1})^T$ and $\boldsymbol{Z}_{t-1} = 1$.

2.5.1. Parameter Estimation and Forecasting

Parameter estimation can be done with the EM algorithm. Further details can be found in [Lam92].

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

2.6. Log-Linear Models

As mentioned in 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. They have the form

$$Y_{kt}|\mathcal{F}_{k,t-1} \sim P(\lambda_{kt}); \forall t \in \mathbb{N}, \tag{2.33}$$

$$\nu_{kt} = \log(\lambda_{kt}) = \beta_0 + \sum_{i=1}^{p} \beta_i \log(Y_{k,t-i} + 1) + \sum_{j=1}^{q} \alpha_j \nu_{k,t-j}. \tag{2.34}$$

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 [Lib16; FT11]. We consider the Log-Linear model because it provides solutions to at least two drawbacks from the INGARCH(p,q) model. First, as a result of 2.6, we have $0 < \sum_{i=1}^{p} \beta_i + \sum_{j=1}^{q} \alpha_j < 1$ and hence it follows for $h \in \mathbb{N}$ that $Cov(Y_{k,t+h}, Y_{kt}) > 0$. Second, when we include covariates, they can only have a positive regression term because otherwise the mean λ_{kt} becomes negative [FT11]. However, in the Log-Linear case we can extend this to

$$Y_{kt}|\mathcal{F}_{k,t-1} \sim P(\lambda_{kt}); \forall t \in \mathbb{N}, \tag{2.35}$$

$$\nu_{kt} = \log(\lambda_{kt}) = \beta_0 + \sum_{i=1}^{p} \beta_i \log(Y_{k,t-i} + 1) + \sum_{j=1}^{q} \alpha_j \nu_{k,t-j} + \boldsymbol{\eta}^T \mathbf{X}_{kt}. \tag{2.36}$$

with $\boldsymbol{\eta} \in \mathbb{R}^r$. Additionally, because of 2.37, it also allows for negative autocorrelation [Lib16].

2.6.1. Parameter Estimation and Forecasting

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

$$\Theta = \left\{ \boldsymbol{\theta} \in \mathbb{R}^{p+q+r+1} : |\beta_1|, \ldots, |\beta_p|, |\alpha_1|, \ldots, |\alpha_q| < 1, |\sum_{i=1}^{p} \beta_i + \sum_{j=1}^{q} \alpha_j| < 1 \right\}. \tag{2.37}$$

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 $\lambda_{k,t+1} = \mathbb{E}[Y_{k,t+1}|\mathcal{F}_{kt}]$. The h-step ahead predictions for $h > 1$ are calculated iteratively again [Lib16].

Log-Linear Models are further discussed in [FT11; WMH11; DDM13].

2.7. Vector Generalised Additive Models

Because we work with multivariate count data, we also look at vector generalised additive models (VGAMs) which extend generalised additive models (GAMs) to higher dimensions. GAMs allow us to reveal and model non-linear relationship in our data, as opposed to linear models or generalised linear models [YW96]. Let y be a univariate response with a distribution in the exponential family and mean μ. Further take a p-dimensional covariate vector and $\boldsymbol{x} = (x_1, \ldots, x_p)^T$. Then the generalised additive model (GAM) is given by

$$g(\mu) = \nu(\boldsymbol{x}) = \beta_0 + f_1(x_1) + \ldots f_p(x_p), \tag{2.38}$$

with f_j being arbitrary smooth functions [YW96]. To extend this model to the

multivariate case, we replace the functions f_j with vector functions. Let $\boldsymbol{f}_k(Y_{kt}) = (f_{(1)k}(Y_{kt}), \ldots, f_{(M)k}(Y_{kt}))^T$ with $M \in \mathbb{N}$ be an arbitrary smooth vector function. Then the vector generalised additive model is given by

$$\mathbb{E}[\boldsymbol{Y}_t] = \boldsymbol{\beta}_0 + \sum_{k=1}^{K} \boldsymbol{f}_k(Y_{k,t-1}), \tag{2.39}$$

where $\mathbb{E}[\boldsymbol{Y}_t] = (\mathbb{E}[Y_{1t}], \ldots, \mathbb{E}[Y_{Kt}])^T$ [YW96]. Further theoretical background about VGAMs are given in [YW96; Yee15; Woo04].

2.8. AR Models

Autoregressive models of order p (AR(p)) are one of the most simple time series models, which makes them very popular. For $\boldsymbol{a} = (a_1, \ldots, a_p)^T \in \mathbb{R}^p$ and a white noise process $\epsilon_{kt} \sim WN(\sigma^2)$, called innovations as well, they are defined as

$$Y_{kt} = a_1 Y_{k,t-1} + \ldots + a_p Y_{k,t-p} + \epsilon_{kt}, \tag{2.40}$$

where $\{Y_{kt} : t = 1, \ldots T; Y_{kt} \in \mathbb{N}_0\}_f$ is again our univariate time series. The multivariate version is defined as

$$\boldsymbol{Y}_t = \boldsymbol{a_1} Y_{t-1} + \ldots + \boldsymbol{a_p} Y_{t-p} + \boldsymbol{\epsilon}_t, \tag{2.41}$$

where $\boldsymbol{a}_j \in \mathbb{R}^{k \times k}$ and $\boldsymbol{\epsilon}_t \sim WN(\Sigma)$.

2.8.1. 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 2.41 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. Further properties and comparison of their estimators can be found in [Sch21].

Like parameter estimation, forecasting is also simple in the AR model. Using the mean squared error as a measure,only considering affine forecasts and using $m \in \mathbb{N}$ past values , hence

$$\hat{\boldsymbol{Y}}_{t+h} = \boldsymbol{c}_0 + \boldsymbol{c}_1 \boldsymbol{Y}_t + \ldots + \boldsymbol{c}_m \boldsymbol{Y}_{t-m+1}, \tag{2.42}$$

we get that the optimal one-step ahead prediction is simply

$$\hat{\boldsymbol{Y}}_{t+1} = \boldsymbol{a}_1 \boldsymbol{Y}_t + \ldots + \boldsymbol{a}_p \boldsymbol{Y}_{t-m+1}, \tag{2.43}$$

for $m > p$ [Sch21]. For $h > 1$, one simply continues recursively using $\hat{\boldsymbol{Y}}_{t+1}$, $\hat{\boldsymbol{Y}}_{t+2}, \ldots$.

2.8.2. Testing for AR Models

To test whether or not a time series follows an AR(p) process the estimates of the white noise process $\hat{\epsilon}_t$ 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 [Sch21].

The Portmanteau Test tests the null hypothesis $H_0 : \mathbb{E}[\boldsymbol{\epsilon}_t, \boldsymbol{\epsilon}_{t-m}^T] = 0$, i.e. if the estimated innovations are uncorrelated. Under the assumption that $(\boldsymbol{Y}_t)_{t=1}^T$ is an AR(p) process and an AR(p) model has been fit, then the used test statistic converges against a chi-squared distribution [Sch21].

The Breusch-Godfrey Test tests if the coefficients $(b_1, \ldots, b_h)$ in the model

$$\boldsymbol{\epsilon}_t = d_1 \boldsymbol{\epsilon}_{t-1} + \ldots d_h \boldsymbol{\epsilon}_{t-h} + \boldsymbol{\eta}_t, \tag{2.44}$$

are zero, i.e. if process $(\boldsymbol{\epsilon}_t)$ follows an AR(h) structure or not. Under the null hypothesis the test statistic follows a chi-squared distribution again [Sch21].

2.9. 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

$$\alpha \circ X = \sum_{i=1}^{X} Z_i. \tag{2.45}$$

The sequence $\{Z_i\}_{i=1}^X$ is called the counting series of X [Sil+05].

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

$$X_t = \alpha_1 \circ X_{t-1} + \alpha_2 \circ X_{t-2} + \ldots + \alpha_p X_{t-p} + \epsilon_t, \tag{2.46}$$

where

1. (ϵ_t) is a sequence of integer-valued i.i.d. random variables, called innovations, with finite first, second and third moment.

2. $\alpha_i \circ X_{t-i}$ for $i = 1, \ldots, p$ and (Z_j) for $j = 1, \ldots, X_{t-i}$ are mutually independent, independent of (ϵ_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. $\sum_{j=1}^{p} \alpha_j < 1$ [Sil+05].

The last condition ensures the existence and stationary of the process.

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

$$Y_{kt} = \alpha_1 \circ Y_{k,t-1} + \alpha_2 \circ Y_{k,t-2} + \ldots + \alpha_p Y_{k,t-p} + \epsilon_{kt}. \tag{2.47}$$

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

2.9.1. Distributional assumptions

While we will mainly assume that the innovations (ϵ_t) follow a Poisson distribution, they can also follow other distributions. One interesting option is, that one can choose a zero-inflated distribution as mentioned in 2.5. This could make the model adequate for our data.

2.9.2. 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 [Sil+05].

The authors in [Sil+05] 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 [Brä93] and [FM04]. Assuming that $(\epsilon_t) \sim_{i.i.d} P(\lambda)$, 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

$$\hat{Y}_{k,n+h} = \mathbb{E}[Y_{k,n+h}|\boldsymbol{Y}_k] = \alpha^h \left[Y_{kn} - \frac{\lambda}{1-\alpha} \right] + \frac{\lambda}{1-\alpha}. \tag{2.48}$$

However, this forecast hardly ever produces integer values. Remedies, like minimising the absolute expected error, have been suggested by [FM04] but the authors in [Sil+05] propose a bayesian approach. It is based on the assumption that both, the future prediction $Y_{k,n+h}$ and the vector of unknown parameters $\boldsymbol{\theta} = (\alpha, \lambda)$ are random [Sil+05]. Since the complexity posterior probability density function makes it difficult to work with it directly, a sampling algorithm can be deployed for estimation. The details are again given in [Sil+05]. The estimator for the conditional expectation is then given by

$$\hat{Y}_{k,n+h} = Y_{kn} \left(\frac{1}{m} \sum_{i=1}^{m} \alpha_i^m \right) + \left(\frac{1}{m} \sum_{i=1}^{m} \frac{1-\alpha_i^h}{1-\alpha_i} \lambda_i \right), \tag{2.49}$$

where m is the sampling size and the pairs (α_i, λ_i) for $i = 1, \ldots, m$ are the sampled parameters.

2.9.3. Testing for INAR(1) Models

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

Parametric re-sampling is the first of the presented methods. 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 [Sil+05].

Residual based methods are based on the Pearson residuals defined by

$$r_{kt} = \frac{Y_{kt} - \mathbb{E}[Y_{kt}|Y_{k,t-1}]}{\mathbb{V}[Y_{kt}|Y_{k,t-1}]^{1/2}}, \tag{2.50}$$

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 [Sil+05].

Another option is based on predictive distributions where an adjusted probability integral transform (PIT) is used. Further details can be found in [Sil+05].

2.9.4. 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 [JY91], which is the one in 2.46, then the autocorrelation function follows that of an AR(p) process [SO05]. However, the authors in [AA90] 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 [AA90].

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 transform the data in such a way, that the values of each category can be seen as the relative share of the total amount at the current time and then predict the relative share of the category for the next point in time. 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 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. One such property is the fact that the VAR model does not depend on the concrete choice of the ilr-transformation [KFH15].

3.2. Preliminaries

The basis of this section is given by [KFH15], [Ego+03] and [FH20].

CoDA, which is short for "'Compositional Data Analysis"' works with compositional data. The key to compositional data is the fact that the absolute value of its parts is less important than the relative relation of the parts to each other. 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 [KFH15]. 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 compositions $\mathbf{x}, \mathbf{z} \in \mathbb{S}^D$ and $a \in \mathbb{R}$ they are defined respectively as [KFH15]

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

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 $z^{-1} := \mathcal{C}(z_1^{-1}, z_2^{-1}, \ldots, z_D^{-1})$, 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

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

This induced geometry is called 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 $\mathbb{S}^D$.

However, in order to use standard statistical tools, it is desirable to move from this geometry to the Euclidean real space [FH20]. There are various ways to map the data from the simplex $\mathbb{S}^D$ to the real space $\mathbb{R}^D$. 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 [KFH15]

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

and map the composition $\mathbf{x}$ to the real space $\mathbb{R}^D$. 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 [FH20].

clr Coefficients

Let $g(\mathbf{x})$ be the geometric mean of $\mathbf{x}$. The centered log-ratio coefficients are then defined as [KFH15]

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

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' [FH20]. It can be shown that the *clr* transformation is an isometry[Ego+03]. Therefore it holds

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

$$d(\mathbf{x}, \mathbf{z})_a = d(clr(\mathbf{x}), clr(\mathbf{z})). \quad (3.10)$$

ilr Coordinates

The isometric log-ratio (ilr) are closely related to the clr Coefficients. Assume the inverse *clr* transformation is isometric. Let $\{v_1, \ldots, v_{D-1}\}$ be an orthonormal base in the hyperplane V. Then $\mathbf{e}_i = clr^{-1}(v_i), i = 1, \ldots, D-1$ is an orthonormal basis of the simplex $\mathbb{S}^D$. For $\mathbf{x} \in \mathbb{S}^D$, the *ilr* transformation can then be defined as [KFH15]

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

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,

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

as well as,

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

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

$$\|x\|_a = \|ilr(x)\| = \|u\|. \tag{3.15}$$

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 $(\mathbf{e}_i)_{i=1}^{D-1}$ [Ego+03]. A special choice of orthogonal coordinates leads to the coordinates

$$ilr(\mathbf{x}) = (u_1, \ldots, u_{D-1})^T, \tag{3.17}$$

$$u_j = \sqrt{\frac{D-j}{D-j+1}} \log \left(\frac{x_j}{\sqrt[D-j]{\prod_{l=j+1}^{D} x_l}} \right), \quad j = 1, \ldots, D-1. \tag{3.18}$$

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 [FH20].

To transform the data back in the simplex, the inverse transformation is given by,

$$x_1 = \exp\left(\sqrt{\frac{D-1}{D}}u_1\right), \tag{3.19}$$

$$x_i = \exp\left(\sum_{j=1}^{i-1}\frac{1}{\sqrt{(D-j+1)(D-j)}}u_j + \sqrt{\frac{D-i}{D-i+1}}u_i\right), \quad i = 2,\ldots,D-1, \tag{3.20}$$

$$x_D = \exp\left(-\sum_{j=1}^{D-1}\frac{1}{\sqrt{(D-j+1)(D-j)}}u_j\right). \tag{3.21}$$

3.4. The VAR Model

Since we have established the basic setting we can now introduce compositional time series (CTS). A CTS $\{\boldsymbol{x}_t : t = 1,\ldots,n\}$ 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 $t = 1,\ldots,n$

$$\sum_{i=1}^{D} x_{it} = \kappa_t, \;\; x_i > 0, i = 1,\ldots D\,; t = 1,\ldots,n. \tag{3.22}$$

Let $\left\{\boldsymbol{Y}_t : t = 1,\ldots,T; \boldsymbol{Y}_t \in \mathbb{N}_0^K\right\}_f$ be our time series for fridge f and assume that $\boldsymbol{Y}_t = (Y_{1t},\ldots,Y_{Kt})^T$ is a K-dimensional compositional vector measured at time $t, t = 1,\ldots,T$. Further, let $\mathbf{u}_t = ilr(\boldsymbol{Y}_t)$ be its ilr transformation determined by the matrix $\mathbf{V}$. Then the VAR model with lag order p is given by [KFH15]

$$\mathbf{u}_t = \mathbf{c_V} + \mathbf{A}_\mathbf{V}^{(1)}\mathbf{u}_{t-1} + \mathbf{A}_\mathbf{V}^{(2)}\mathbf{u}_{t-2} + \ldots + \mathbf{A}_\mathbf{V}^{(p)}\mathbf{u}_{t-p} + \boldsymbol{\epsilon}_t. \tag{3.23}$$

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 $\mathbf{u}_{t-1},\ldots,\mathbf{u}_{t-p}$. 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 [KFH15].

Estimation of the VAR Model

Assuming T observations are used for the model, equation 3.23 can be written in matrix form as

$$\mathbf{U} = \mathbf{ZB} + \mathbf{E},$$
$$\mathbf{U} = (\mathbf{u}_1, \dots, \mathbf{u}_T)^T \in \mathbb{R}^{T\times(K-1)},$$
$$\mathbf{Z} \in \mathbb{R}^{T\times[(K-1)p+1]} \text{ with } \mathbf{Z}_t = \left(1, \mathbf{u}_{t-1}^T, \dots, \mathbf{u}_{t-p}^T\right)^T,$$
$$\mathbf{B} = \left[\mathbf{c}, \mathbf{A}^{(1)}, \dots, \mathbf{A}^{(p)}\right]^T \in \mathbb{R}^{(K-1)p+1\times(K-1)}.$$

The parameter $\mathbf{B}$ can then be estimated separately for each column of $\mathbf{U}$ by the ordinary least squares (OLS) method. In addition, if there are no restrictions posed on the parameter, the estimator is equal to the generalised least squares (GLS). If the VAR(p) process is normally distributed and the rows of the error matrix $\mathbf{E}$ represent a white noise process, thus $\boldsymbol{E} \sim WN(\Sigma)$ where Σ is the covariance matrix, then the estimator is also equal to the maximum likelihood (ML) estimator. Under these assumptions it can be shown that the OLS estimator is consistent and asymptotic normal [KFH15] [Lüt07].

3.5. $\mathcal{T}$-Spaces

As we have seen, lies the focus in compositional data analysis 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 [PEL13]. Here, we will dive deeper into the second method mentioned. An overview over the first method can be found in [PEL13].

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 $(t(\boldsymbol{x}), \mathcal{C}(\boldsymbol{x}))^T$ [PEL13]. 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 [PEL13]. 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 [PEL13]

$$\boldsymbol{x} \oplus_+ \boldsymbol{y} := (x_1 \cdot y_1, \ldots, x_D \cdot y_D)^T, \tag{3.24}$$

$$\alpha \odot_+ \boldsymbol{x} := (x_1^\alpha, \ldots, x_D^\alpha)^T, \tag{3.25}$$

$$\langle \boldsymbol{x}, \boldsymbol{y} \rangle_+ := \langle \log(\boldsymbol{x}), \log(\boldsymbol{y}) \rangle \,. \tag{3.26}$$

Here, $\langle , \rangle$ denotes the usual Euclidean inner product on $\mathbb{R}^D$.

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

$$\tilde{\boldsymbol{x}} \oplus_T \tilde{\boldsymbol{y}} = (t(\boldsymbol{x}) \oplus_+ t(\boldsymbol{y}), \boldsymbol{x} \oplus_a \boldsymbol{y})^T := (t(\boldsymbol{x}) \cdot t(\boldsymbol{y}), \mathcal{C}(\tilde{x}_1 \tilde{y}_1, \ldots, \tilde{x}_D \tilde{y}_D))^T, \tag{3.27}$$

and the external multiplication as

$$\alpha \odot_T \tilde{\boldsymbol{x}} = (\alpha \odot_+ t(\boldsymbol{x}), \alpha \odot_a \boldsymbol{x})^T := (t(\boldsymbol{x})^\alpha, \mathcal{C}(\tilde{x}_1^\alpha, \tilde{x}_D^\alpha)^T, \tag{3.28}$$

where $\oplus_a$ and $\odot_a$ are the perturbation and power transformation defined in 3.2 and $\oplus_+$ and $\odot_+$ the respective operations defined for $\mathbb{R}_+$ 3.26.

The inner product is defined as

$$\langle \tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}} \rangle_T := \langle t(\boldsymbol{x}), t(\boldsymbol{y}) \rangle_+ + \langle \mathcal{C}(\boldsymbol{x}), \mathcal{C}(\boldsymbol{y}) \rangle_a \,, \tag{3.29}$$

where $\langle , \rangle_+$ is the inner product in $\mathbb{R}_+$, and $\langle , \rangle_a$ is the Aitchison inner product defined in 3.5 [PEL13].

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

$$d_T^2(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}) = d_+^2(t(\boldsymbol{x}), t(\boldsymbol{y})) + d_a^2(\mathcal{C}(\boldsymbol{x}), \mathcal{C}(\boldsymbol{y})), \tag{3.30}$$

with $d_+^2(\boldsymbol{x}, \boldsymbol{y}) = d(\log(\boldsymbol{x}), \log(\boldsymbol{y}))$ and d is the Euclidean distance.

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}. As a result, the function t must be related to the sum of the components. To see this, write $\boldsymbol{x} \in \mathbb{R}_+^D$ as $\boldsymbol{x} = \frac{\sum_{i=1}^D x_i}{\kappa} \cdot \mathcal{C}(\boldsymbol{x})$. Hence $\frac{\sum_{i=1}^D x_i}{\kappa} \cdot \mathcal{C}(\boldsymbol{x}) = h^{-1}((t(\boldsymbol{x}), \mathcal{C}(\boldsymbol{x}))^T)$ [PEL13]. The second condition is the preservation of the vector space properties in $\mathbb{R}_+^D$ and $\mathcal{T}$

$$h(\boldsymbol{x} \oplus_+ \boldsymbol{y}) = h(\boldsymbol{x}) \oplus_T h(\boldsymbol{y}), \tag{3.31}$$

$$h(\alpha \odot_T \boldsymbol{x}) = \alpha \odot_T h(\boldsymbol{x}). \tag{3.32}$$

This means for the function t that

$$t(\boldsymbol{x} \oplus_+ \boldsymbol{y}) = t(\boldsymbol{x}) \cdot t(\boldsymbol{y}), \tag{3.33}$$

$$t(\alpha \odot_T \boldsymbol{x}) = (t(\boldsymbol{x}))^{\alpha}. \tag{3.34}$$

In [PEL13] the authors show that for $h_s = ((t_s(\boldsymbol{x}), \mathcal{C}(\boldsymbol{x}))^T)$ 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 $\mathcal{T}$ [PEL13]. The vector space operations can be defined as

$$\boldsymbol{x} \oplus_{+s} \boldsymbol{y} = h_s^{-1}(\tilde{\boldsymbol{x}}) \oplus_T h_s^{-1}(\tilde{\boldsymbol{y}}), \tag{3.35}$$

$$\alpha \odot_{+s} \boldsymbol{x} = \alpha \odot_T h_s{}^{-1}(\tilde{\boldsymbol{x}}), \tag{3.36}$$

$$d_{+s}^2(\boldsymbol{x}, \boldsymbol{y}) = d_T^2(h_s(\boldsymbol{x}, \boldsymbol{y})), \tag{3.37}$$

where $\oplus_{+s}$ and $\odot_{+s}$ are the new operations in $\mathbb{R}_+^D$ and d^2 is the squared distance in $\mathcal{T}$.

With the Euclidean 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 $\boldsymbol{w}_t = (t(\boldsymbol{Y}_t), ilr(\boldsymbol{Y}_t))^T$ take the *irl* 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 $\boldsymbol{\Sigma_\epsilon}$

$$\mathbf{w}_t = \mathbf{c}_{\mathbf{V}} + \mathbf{A}_{\mathbf{V}}^{(1)} \mathbf{w}_{t-1} + \mathbf{A}_{\mathbf{V}}^{(2)} \mathbf{w}_{t-2} + \ldots + \mathbf{A}_{\mathbf{V}}^{(p)} \mathbf{w}_{t-p} + \boldsymbol{\epsilon}_t. \tag{3.38}$$

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 [KFH15]. The estimation of model 3.38 is carried out analogous to 3.4.

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 considerate 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 in order to perform compositional data analysis. In the following we summarise the methods presented in [LFT21; MBP03].

3.6.1. Rounded Zeros

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 compositions as $\boldsymbol{x}_s = \mathcal{C}(\boldsymbol{S}\boldsymbol{x})$. If we have rounded zeros, a simple method proposed in [MBP03] 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 [LFT21]. This means $\boldsymbol{r}$ has the form

$$r_j = \begin{cases} 0.65 \cdot DL, & \text{if } x_j = 0, \\ x_j, & \text{if } x_j > 0, \end{cases} \tag{3.39}$$

Additionally [MBP03] mentions two other methods. First, the Additive Replacement Strategy, which was first introduced by Aitchison in [Ait86], and is given by

$$r_j = \begin{cases} \frac{\delta(m+1)(D-m)}{D^2}, & \text{if } x_j = 0, \\ x_j - \frac{\delta(m+1)m}{D^2}, & \text{if } x_j > 0. \end{cases} \tag{3.40}$$

As we can see in 3.40, 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 $\mathbb{S}^D$ [MBP03]. 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 $\boldsymbol{x}_s$ and $\boldsymbol{y}_s$ 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 [MBP03].

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 $\boldsymbol{r} = \mathcal{C}(\boldsymbol{w})$

$$r_j = \begin{cases} \frac{\kappa}{\kappa + \sum_{i|x_i=0} \delta_i} \delta_j, & \text{if } x_j = 0, \\ \frac{\kappa}{\kappa + \sum_{i|x_i=0} \delta_i} x_j, & \text{if } x_j > 0. \end{cases} \tag{3.41}$$

This method depends again on δ_j and the number of zeros m.

Third, which is the main result of [MBP03], is the multiplicative replacement strategy. The proposed replacement is

$$r_j = \begin{cases} \delta_j, & \text{if } x_j = 0, \\ \left(1 - \frac{\sum_{i|x_i=0} \delta_i}{\kappa}\right) x_j, & \text{if } x_j > 0, \end{cases} \tag{3.42}$$

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

1. It is a more intuitive approach. If δ 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: $\forall \boldsymbol{y} \in \mathbb{S}^D : (\boldsymbol{y} \oplus \boldsymbol{r})_s = (\boldsymbol{y} \oplus \boldsymbol{x})_s$,
   - Power transformation Invariance: $\forall \alpha \in \mathbb{R} : (\alpha \odot \boldsymbol{r})_s = (\alpha \odot \boldsymbol{x})_s$.

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 [MBP03].

Another method proposed in [LFT21] 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 [PM15].

The authors in [PM15] 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 $\mathbb{R}^D$.

EM-based algorithm

The Expectation-Maximisation (EM) algorithm [DLR77] is a widely used method in imputation. In the setting of multivariate compositional data, it uses information in the covariance structure to conditionally estimate the censored values. Given a vector $\boldsymbol{x}$ with observed $\boldsymbol{x}_{obs}$ and unobserved $\boldsymbol{x}_{non}$ components the EM-algorithm consists of two steps

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

Here, DL is the mapped censoring threshold [PM15].

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 [PM15].

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 $\hat{\theta}_t$, simulate from $P(\boldsymbol{x}_{non}|\boldsymbol{x}_{obs}, \boldsymbol{x}_{non} < DL; \hat{\theta}_t)$.

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

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 [PM15].

Bayesian-multiplicative replacement

A common assumption for multivariate count data is 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 $\mathbb{E}[\pi_j|x_j = 0] = t_j \frac{s}{n+s}$ [PM15].

The presented methods as well as additional methods are explained in more detail in [PM15].

3.6.2. Essential Zeros

The case of essential zeros is not as straightforward because zero is the true value of the observation. In [AK03] 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 a MCMC algorithm. After this is done, hypothesis testing and statistical analysis can be performed.

A vector space approach for the simplex is presented in [BTB06] and extended in the R-package *compositions* [vTB23]. 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 [EP05] a subcomposition can be seen as a projection of the clr transformed composition into the null space of the vectors $\{\boldsymbol{w}_i : i \in M\}$. Hence, one only observes a projection of the true composition. Let P_M be the orthogonal projection onto the null space 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 [BTB06]. 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

$$P_M(clr(\boldsymbol{x}))_i = \begin{cases} clr(\boldsymbol{x}_s)_i, & \text{if} i \notin M \\ 0, & \text{if} i \in M. \end{cases} \tag{3.43}$$

The subsequent *ilr* transformation is then based on this modified approach with $ilr_{\mathbf{V}}(\mathbf{x}) = \mathbf{V}^T P_M(clr(\mathbf{x}))$.

In [Lei+13] 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 [Lei+13].

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 an excessive amount of 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.

4.1.1. CoDA Specifications

As mentioned above the CoDA model must not include any zero values. Since in the CoDA context we see our data as relative data, 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 0.5 which is the simple replacement strategy in 3.41. 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 [AK03]. 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 $\boldsymbol{u}_t$ in model 3.23 changes to

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

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 3.38.

4.1.2. INGARCH Specifications

As an alternative to the Poisson distribution in 2.5, a negative binomial distribution can be used as well. This would change 2.5 to

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

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 2.3 we can also choose to include external factors or not. 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.

The optimal one step ahead prediction is given by the conditional expectation 2.2. However, since we only expect integer values and $\lambda_{k,t+1} \in \mathbb{R}_+$, we will round the values of $\lambda_{k,t+1}$ to the next integer and use this value.

4.1.3. 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 difference 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 true value of the time series for category k, $\hat{y}_{ftk}$ the predicted value and $y_{naive_{ftk}}$ the naive predicted value . Then we define our measure as

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

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

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

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

Another possible extension is to take the square root

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

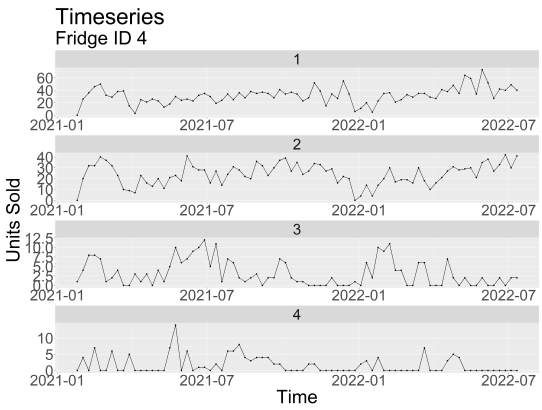
One future extension which can be investigated is the introduction of weights. This could be used for example when the performance of the model in one category should be put more into focus.

4.2. Examples of model application

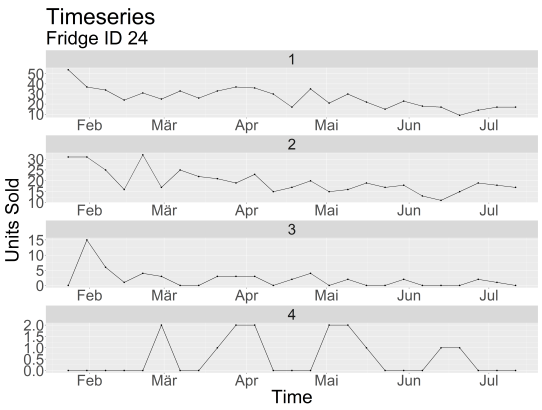
To improve understanding of our data and the models we show some application of the models on some exemplary fridges. We choose fridges 4 and 24. Hence $f \in \{4, 24\}$. Furthermore we start with analysing the aggregated 4 main categories which means $K = 4$.

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 categorie for each fridge, we have four subplots.

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



(a) Fridge 4 with all four main categories

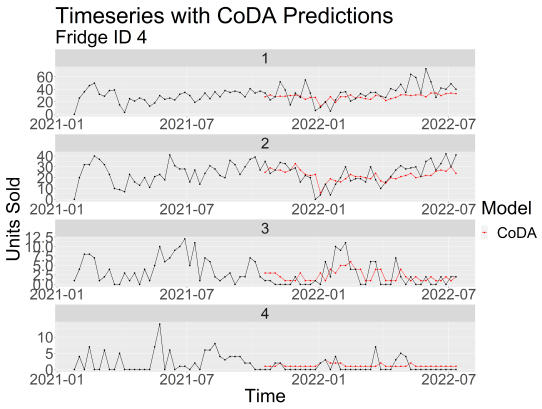


(b) Fridge 24 with all four main categories

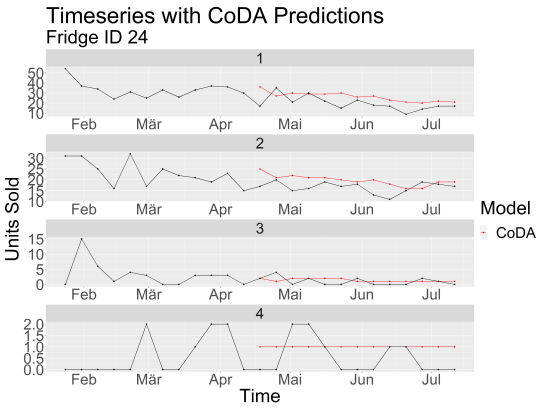
Figure 4.1.: Time series for two fridges

the range of 0 to 2. In both figures 4.1 for category 4, we can see the excessive amount of zero values in our data which makes the previously mentioned transformations 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, add 0.5 to all values use $\mathcal{T}-$Spaces with the logarithmic sum 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 4.2a. Both, categories 1 and 2 with bigger values and categories 3 and 4 with lower values, are in general modelled well. Also in 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.



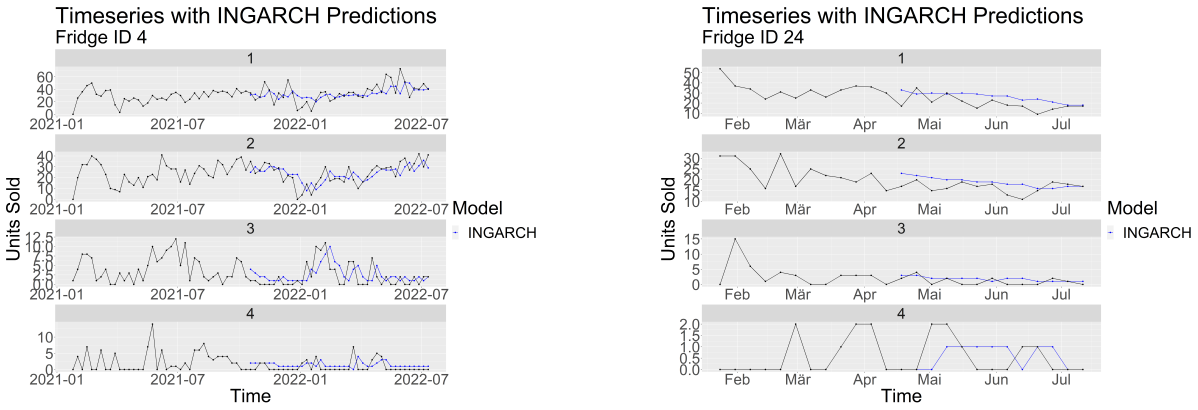
(a) Fridge 4 with the CoDA model



(b) Fridge 24 with the CoDA model

Figure 4.2.: Time series with CoDA model

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 2.3. The general trend is again captured well and in the instance of 4.3a it seems to be more reactive to sudden peaks, as often the value predicted after such a peak is heavily influenced by it.



(a) Fridge 4 with the INGARCH model

(b) Fridge 24 with the INGARCH model

Figure 4.3.: Time series with INGARCH model

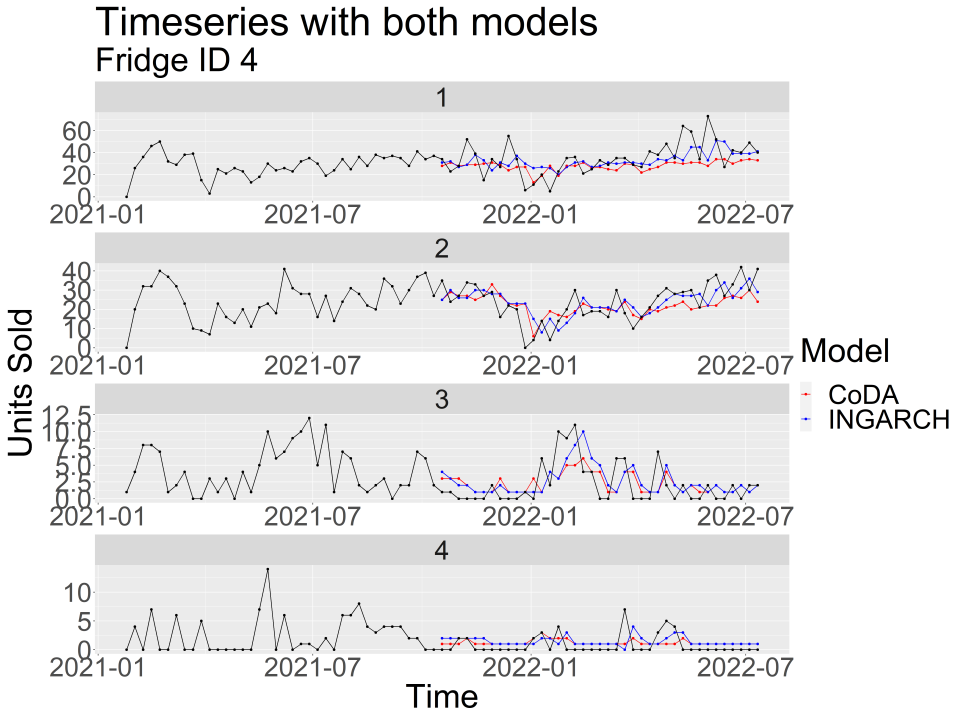
To directly compare both models, we plot the predictions in one figure 4.4. The model specifications are the same as above. We can see that the models produce similar results to each other. In this instances it appears that INGARCH predicts slightly higher values than CoDA.

In order to get some further insight in the accuracy of our predictions, we add 95 % prediction intervals 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 4.5a for category 3 and 4. However, most data points are covered by both bands.

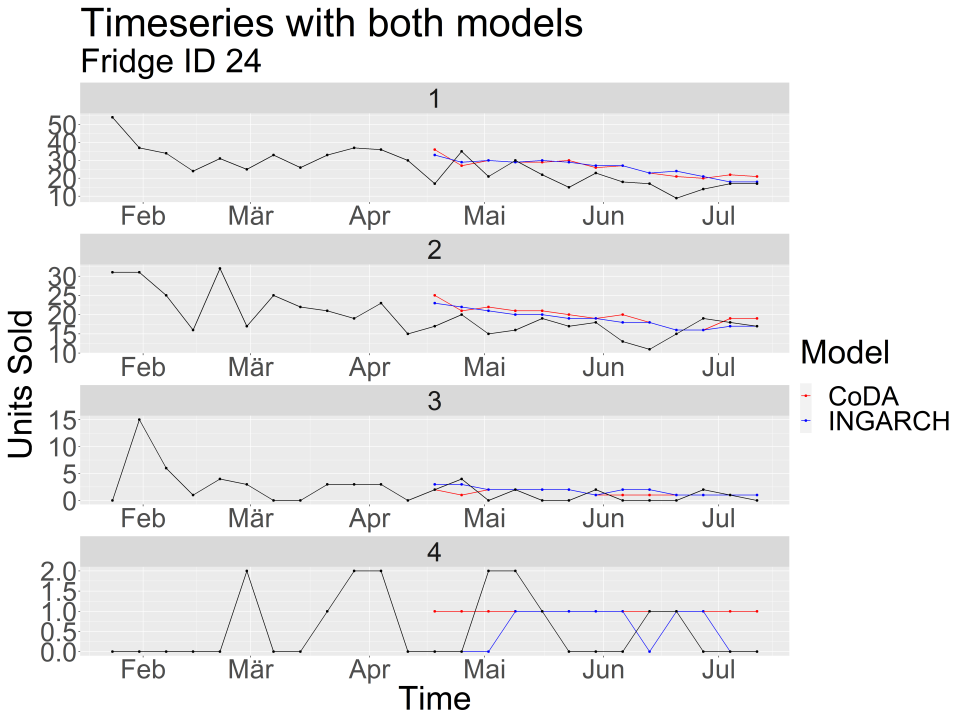
4.3. R-Code

4.3.1. R-Packages

We conducted our analysis in the statistical software R [R C22]. For our data cleansing, data handling and plotting we use the *tidyverse* package [Wic+19]. Further we use the packages *here* [Mül20], *miceadds* [RG23] and *parallel*, which is part of core R, to facilitate

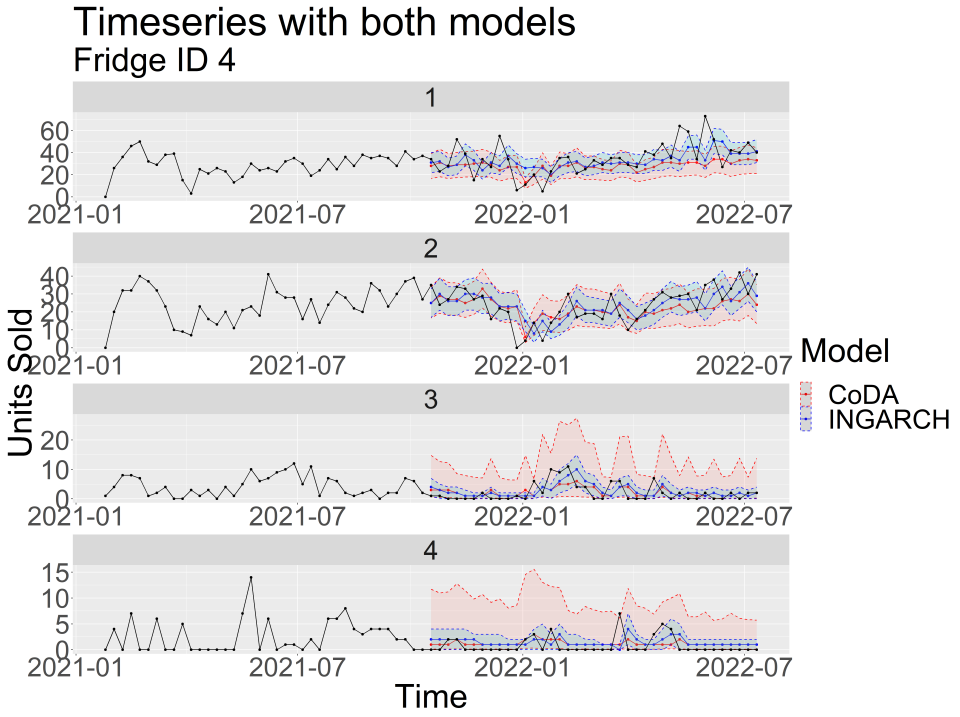


(a) Fridge 4 with the both models

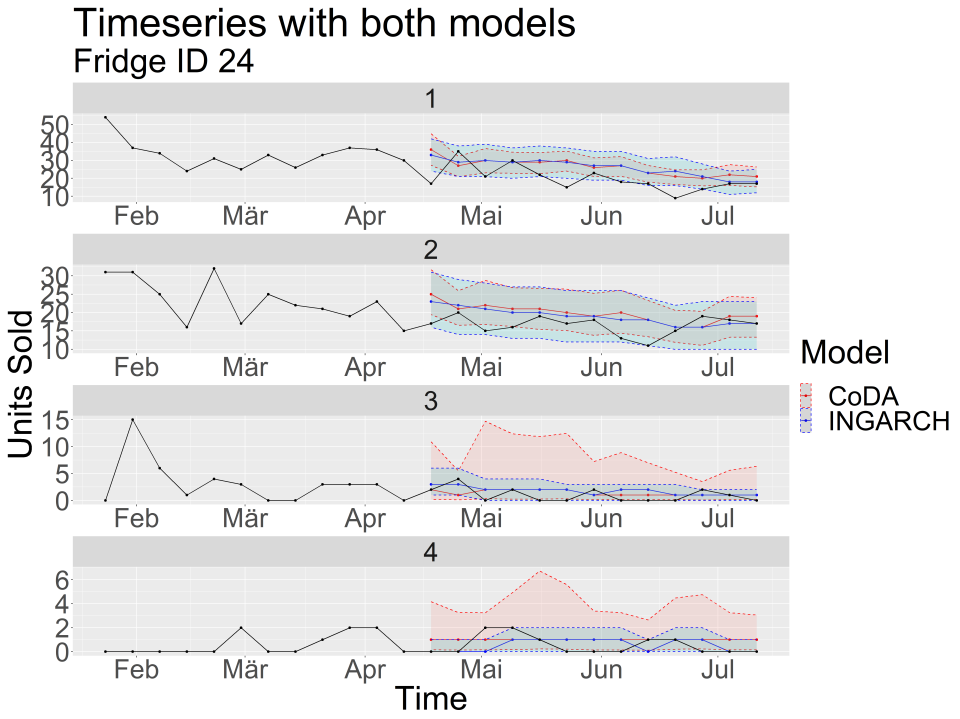


(b) Fridge 24 with the both models

Figure 4.4.: Time series with both models



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



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

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

our analysis.

For building our CoDA model we use the packages *vars* [Pfa08b; Pfa08a] and *robCompositions* [THF11; FHT18]. Especially the functions `pivotCoord`, which performs the ilr transformation described in 3.2,`VAR`, which builds the VAR model described in 3.4, and `D2invPC` which performs the necessary back transformation to get predictions in the desired space. The INGARCH(p,q) analysis is mainly done with the package *tscount* [LFF17; Lib+20]. 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 model were fitted using the function `zeroinfl` from the package *pscl* [ZKJ08]. For the VGAM we used the package *VGAM* [Yee10]. 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 [SPS09]. Second, the classical approach was done using the function `EST_ZINAR` from the package *ZINA1*.

In general, all 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 model and CoDA specific functions are only used for their respective methods.

4.3.2. 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 A.

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 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` brings 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 transform 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: If TRUE, then the one-vs-all method is used.
- PivotGroup: If 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 h.
- TakeSubCategory: If TRUE, then 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 values of the window
- *Name of Category 1*: The number of sold items belonging to *Name of Category 1*.

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

- week_date: The 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 one of the naive model in 2.4 we calculate the errors for this model as well.

Arguments:

- Model_Result: The result calculated by either `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: Which model was used.

Model.ErrorOverall

The function `Model.ErrorOverall` is closely related to `Model.Error`. This function calculates the error measure defined in 4.1.3. One can decide if the error measure should be calculated over all categories or if they should be split up in subsets as in 4.1.3.

Arguments:

- Error_Result: The result of the function `Model.Error`.
- Fnct: Function to summarise the errors. This opens up to use different methods like the mean or median.
- SplitByGroup: If 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: Which model was used.
- group: The subsets of categories as defined in 4.1.3.

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 h.
- TakeSubCategory = If TRUE, then 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 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 4.3.1 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 future 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 distribution 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 be extended at each step or kept at a fixed length.
- External: For INGARCH. Should external factors as in 2.3 be 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 distribution 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: If 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 be 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. Should external factors as in 2.3 be used?
- HistoryLength: The length of the history. Can be an absolute number or a ratio h.
- Multicore: If 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 analogue to CountModel.Preparation
Arguments:

- Data: The data to be transformed.
- ZeroHandling: Method for zero handling. Currently there is no treatment or them being replaced with 1.
- TSpace: If TRUE, then $\mathcal{T}$-Spaces are used.
- Log: If TRUE, then the logarithm of the total sum is used in the $\mathcal{T}$-Space.
- OneVsAll: If TRUE, then the one-vs-all method is used.
- PivotGroup: If one-vs-all is used, this specifies the pivot category.

- HistoryLength: The length of the history. Can be an absolute number or a ratio h.

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

- week_date: The 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 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: If TRUE, then the one-vs-all method is used.
- TSpace: If TRUE, then $\mathcal{T}$-Spaces are used.
- Log: If TRUE, then the logarithm of the total sum is used in the $\mathcal{T}$-Space.
- PivotGroup: If 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: If TRUE, then the logarithm of the total sum is used in the $\mathcal{T}$-Space.
- TSpace: If TRUE, then $\mathcal{T}$-Spaces are used.
- OneVsAll: If TRUE, then the one-vs-all method is used.
- PivotGroup: If one-vs-all is used, this specifies the pivot category.
- HistoryLength: The length of the history. Can be an absolute number or a ratio h.
- 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 be extended at each step or kept at a fixed length.

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.

4.4. Results

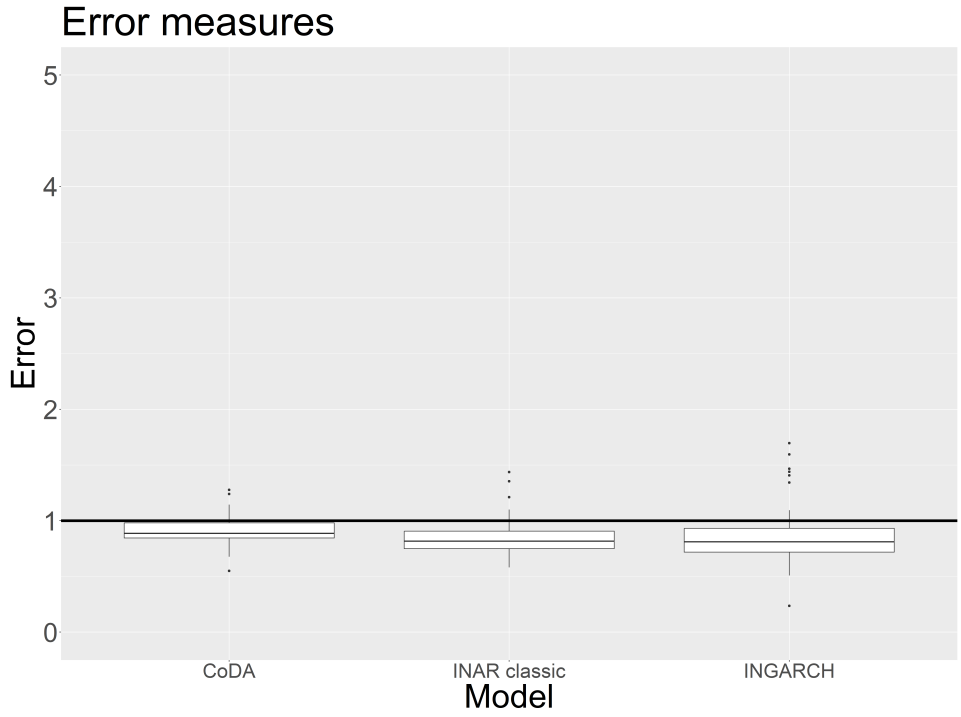
In this section we present and describe the results for our methods with their variations. For this we use the previously introduced error measure, calculate it for all available fridges and summarise the results. We show the results as graphics for easier interpretation. Since we focus on the CoDA and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

4.4.1. Model Comparison

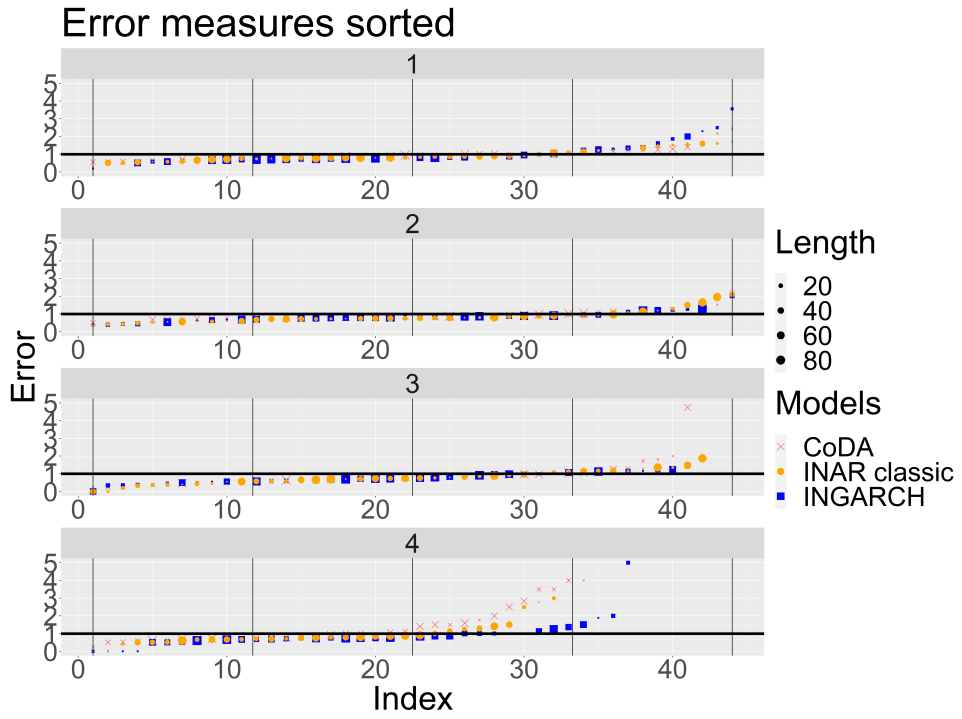
Here, we compare the INGARCH(1,1) model, with the CoDA and INAR(1) model. For all three models we use the same parameter values, namely a window factor of $w = 0.5$, the whole history $h = 1$ and extending windows. For CoDA, the settings are no $\mathcal{T}$-Space, one-vs-all method and the simple replacement strategy. 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 2.9.2. When we speak of standard settings or values in the following, we mean these settings.

In 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 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 reasonable well and similar, differences emerge for category 3 and 4. INGARCH seems to be the clear favourite in category 4, while CoDA and INAR have similar performances. However, for all models there are again time series with errors that are too high to be shown, or that couldn't get calculated at all.



(a) Boxplot for the different models



(b) Quantiles for the different models

Figure 4.6.: Comparison of the different models

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 and CoDA model exhibit similar performance and both are slightly worse than the INAR and INGARCH model. In 4.7b we see again the error measure for each category. While the models perform similar for category 3, the ZIM and INGARCH model outperform the INAR and CoDA model for category 4. Here it is worth mentioning, that category 4 is the main category with the most amount of zeros in our data. Especially CoDA seems to struggle with the big amount of zeros present.

4.4.2. 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

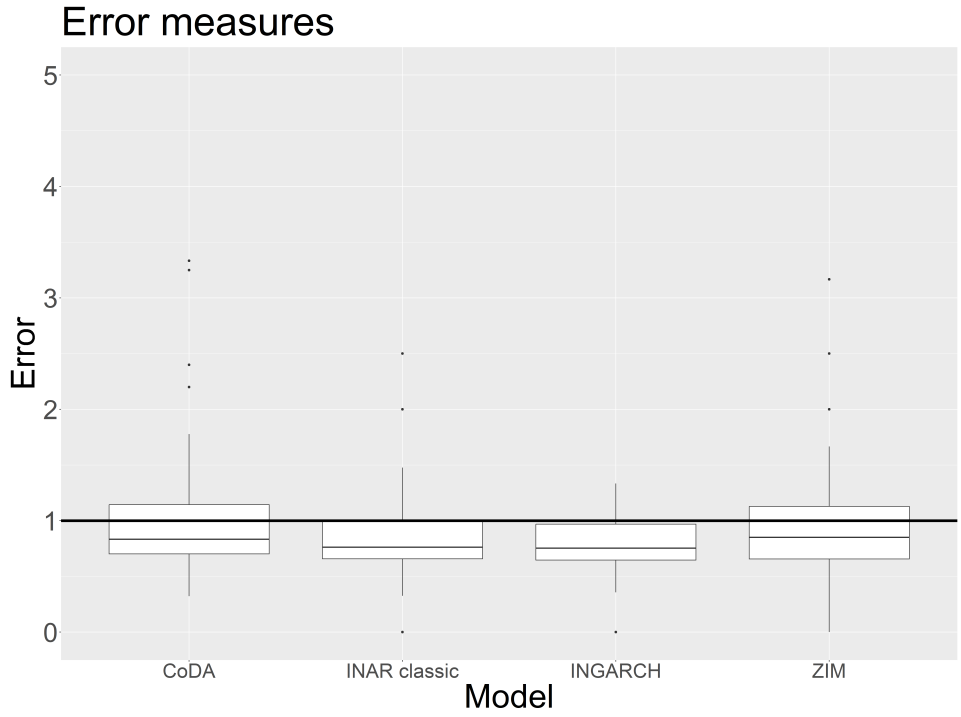
As mentioned various times throughout this thesis, the the history is one of the parameters which can be adjusted. In figure 4.8 we visualise the results as a boxplot, a quantile plot as before and additionally an histogram to get a feeling for the error distribution.

In figure 4.8 we can see that the results for CoDA does 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, that we have too little values for estimating.

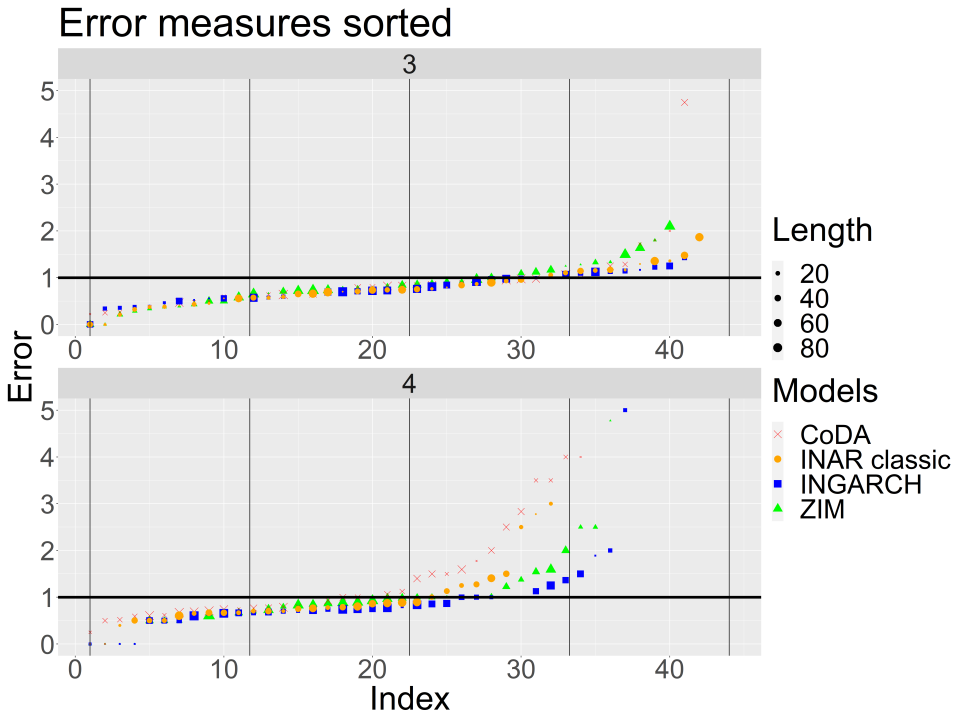
For INGARCH, the results are similar as well. For $h = 1$ we get slightly higher values for the error measure as seen in 4.8a. But again in 4.8c we see that we have less values for the shorter history for the same reason as above.

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

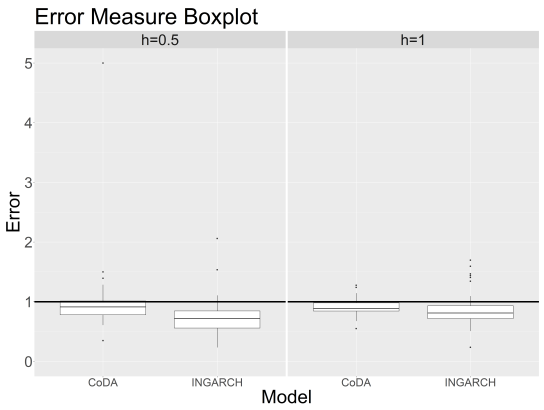


(a) Boxplot for the different models

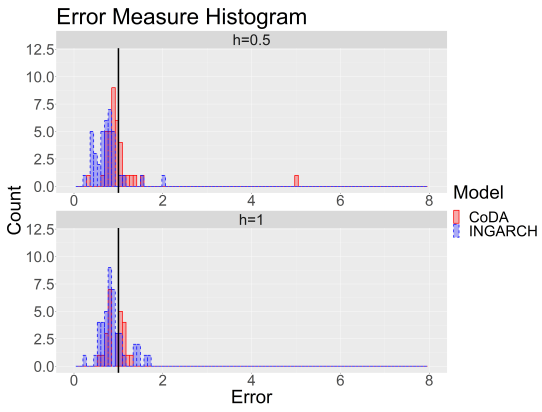


(b) Quantiles for the different models

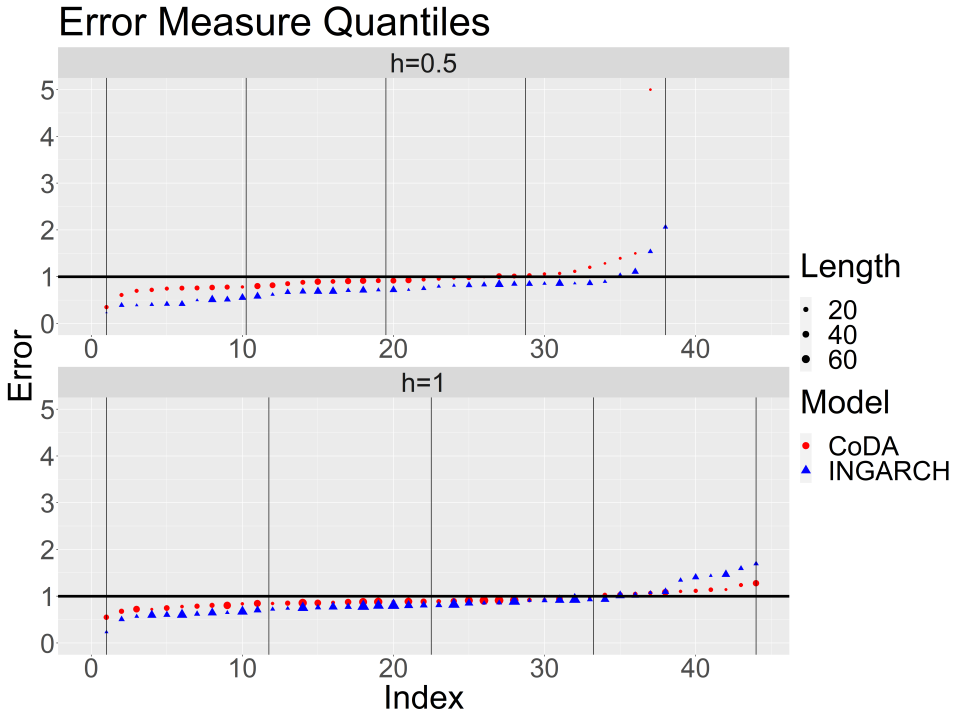
Figure 4.7.: Comparison of the different models



(a) Boxplot of different h



(b) Histogram of different h



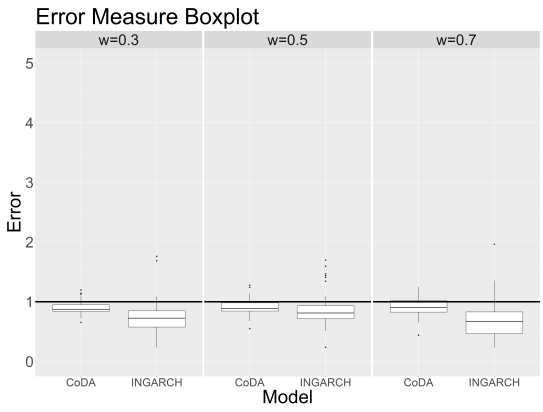
(c) Quantiles Plot of different h

Figure 4.8.: Comparison of different h

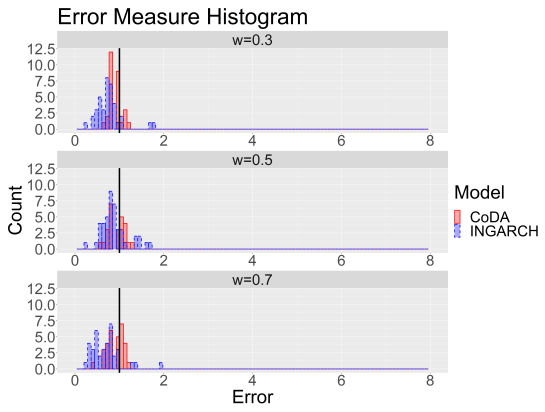
in 4.9. In general, there is not much difference between the different frames. INGARCH seems to perform better for all three values.

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 any more in the plot. This could either be a result of them being to high, or that the model couldn't be fit on those fridges.

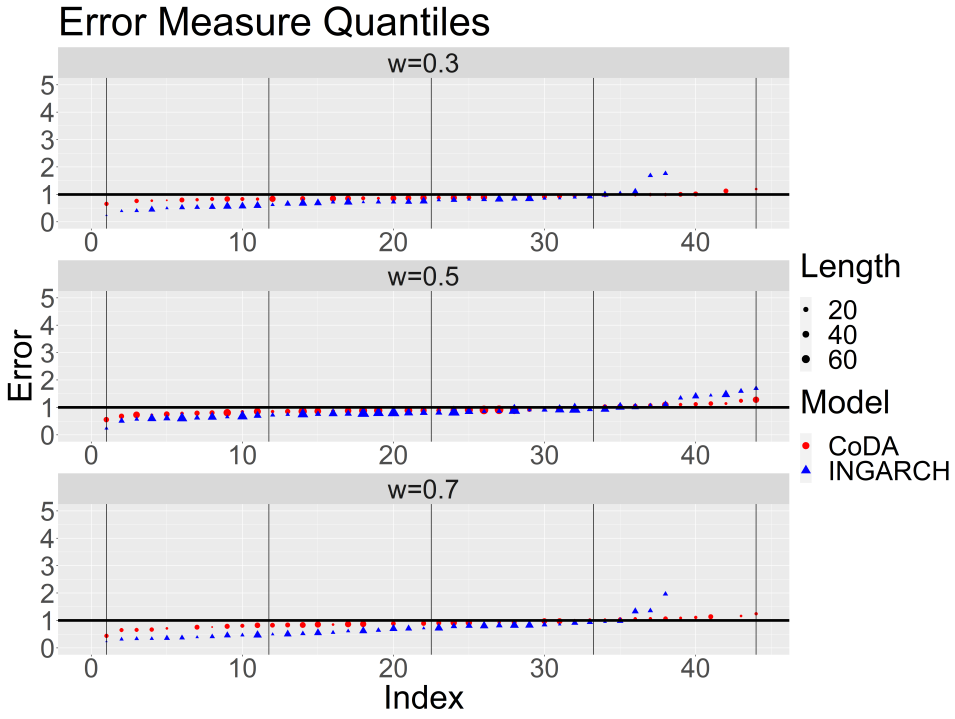
For CoDA there seems not to be much difference. The best results are obtained with



(a) Boxplot of different w



(b) Histogram of different w



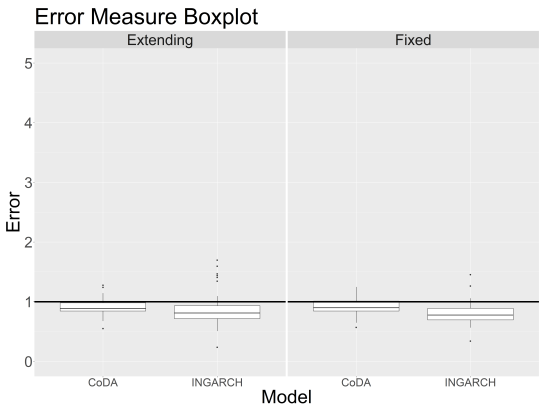
(c) Quantiles of different w

Figure 4.9.: Comparison of different w

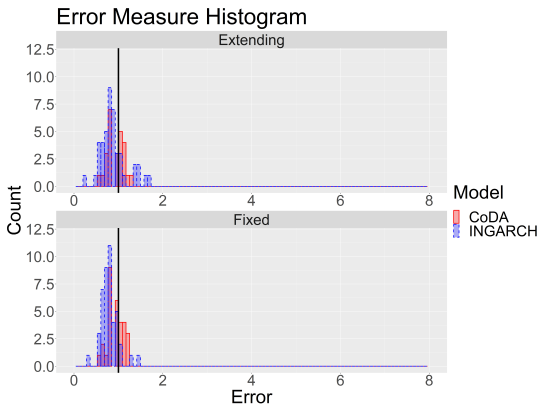
$w = 0.3$, but the differences are only marginal.

Window Shape

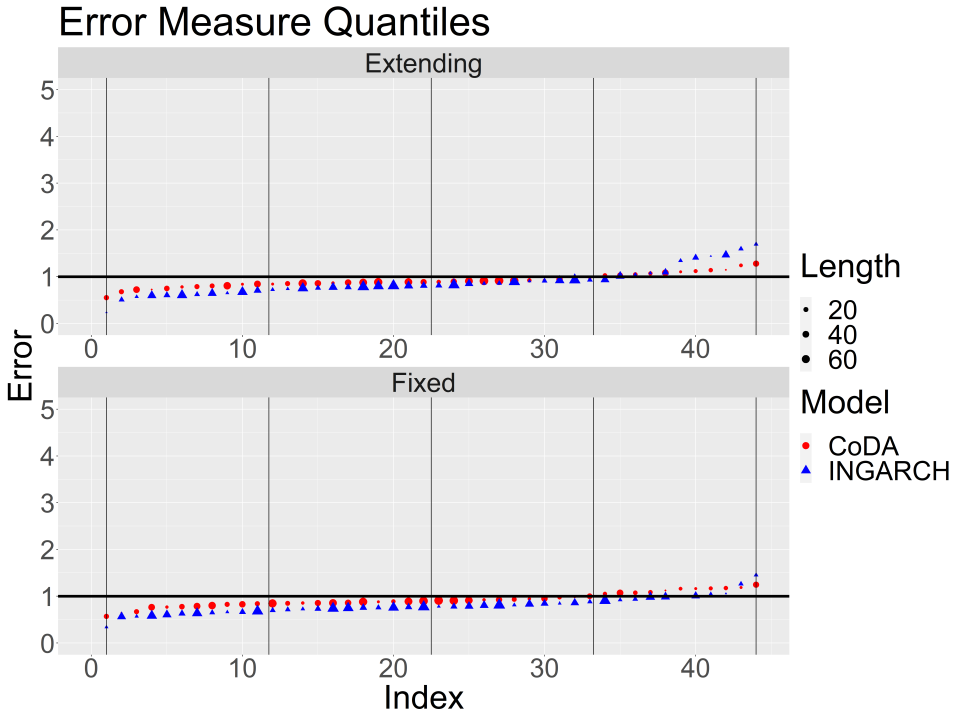
We also vary the shape of the window. As explained in 4.1 we either use a fixed amount of points and add and remove points as time goes on, or we continuously add points to the window. The results are in figures 4.10. We can see that there are no big differences between the methods. For both, CoDA and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



(a) Boxplot for different window shapes



(b) Histogram for different window shapes



(c) Quantiles for different window shapes

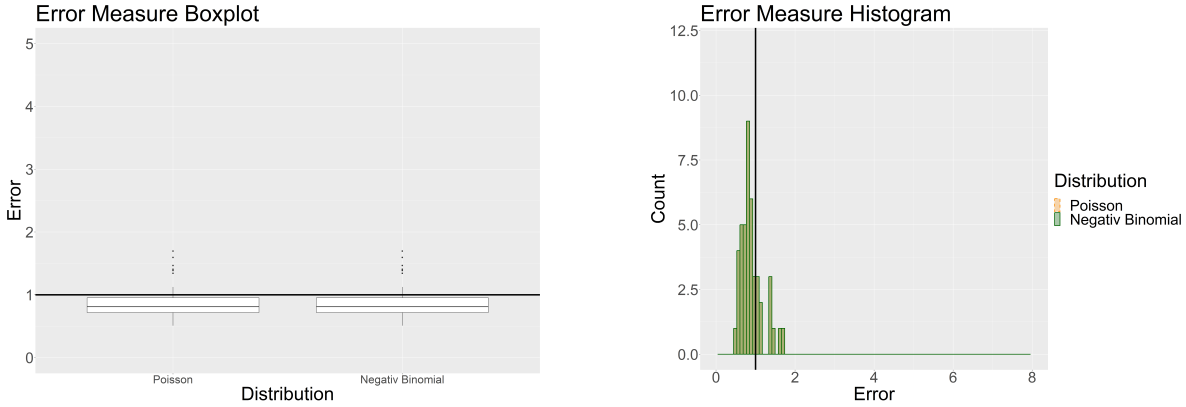
Figure 4.10.: Comparison of different window shapes

4.4.3. 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 4.1.2 we can replace the Poisson distribution with a Negative Binomial Distribution in 2.5. The results are shown in 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 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.

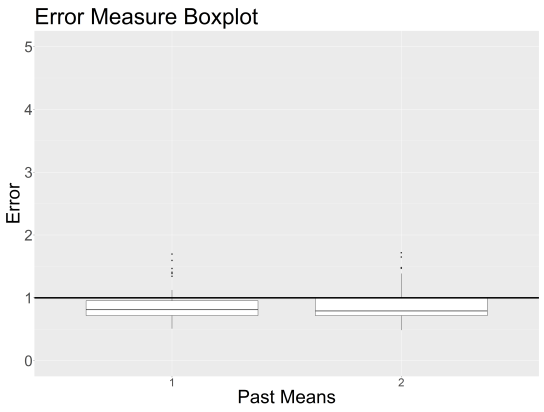
Number of Past Means and Observations

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 INGARCH(2,1) model. However, further models could be tried out and compared. One could even optimise over the optimal order.

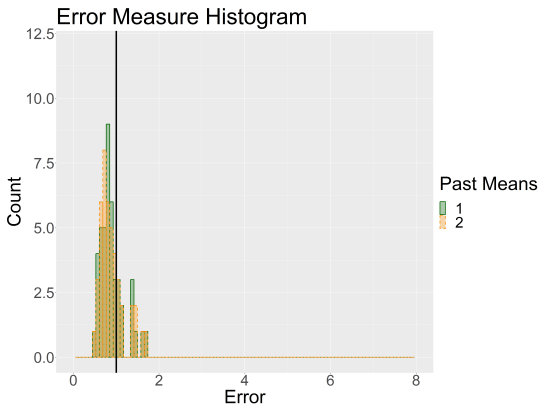
In figure 4.12 we compare the INGARCH(1,1) model (red) with the INGARCH(1,2) model (blue). We can see that the performance is very similar. Hence we prefer the smaller model.

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.

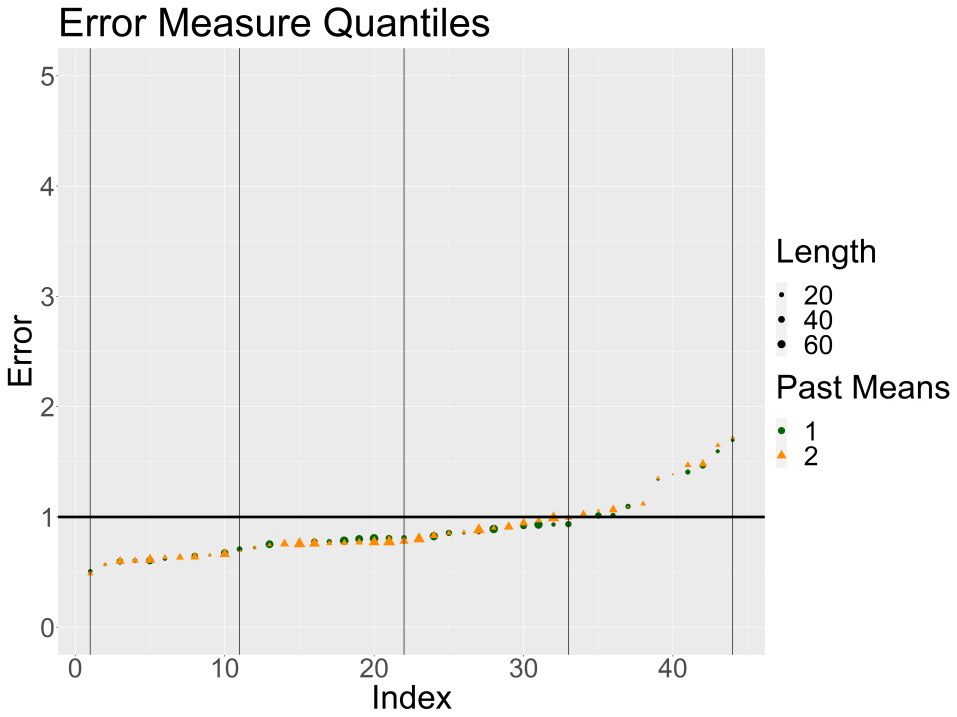
As we saw, there is not much difference between the INGARCH(1,1), INGARCH(2,1) and INGARCH(1,2) model. One could compare the AIC or some other measure for the different models and base their choice on that. However, this is not further explored here and hence the INGARCH(1,1) model is taken because it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



(a) Boxplot for a different number of past means



(b) Histogram for a different number of past means

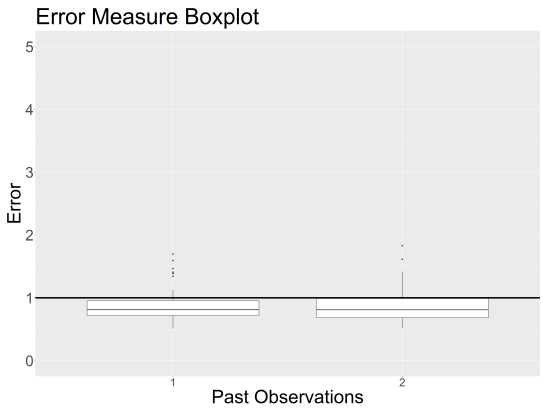


(c) Quantiles for a different number of past means

Figure 4.12.: Comparison of a different number of past means

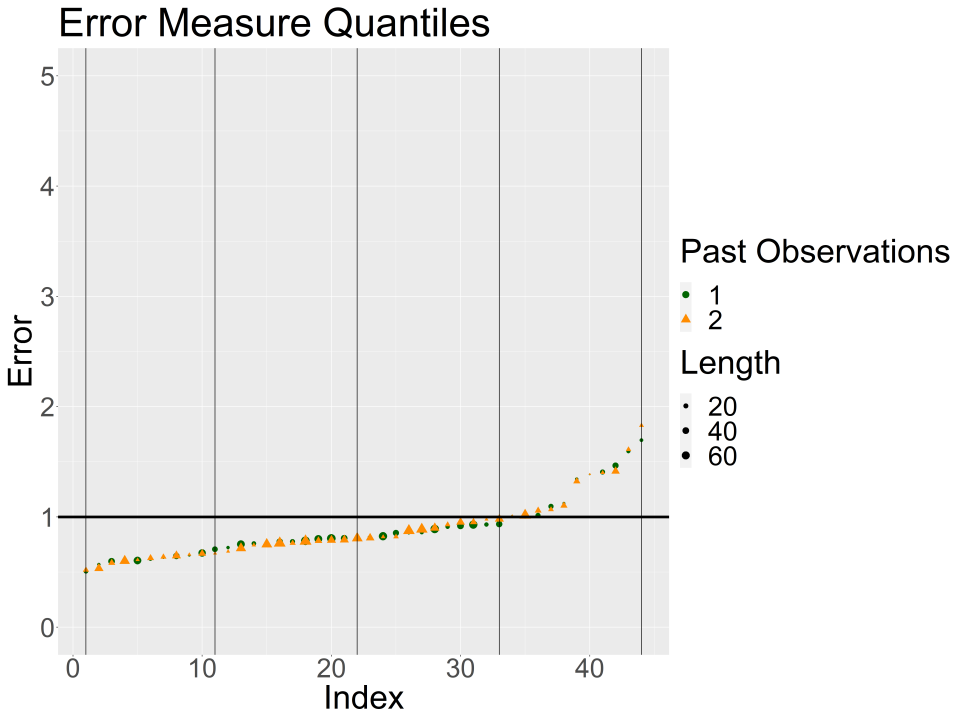
4.4.4. CoDA Specifications Results

Lastly we will compare different CoDA Specifications as mentioned in 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 $w = 0.5$, add the simple replacement strategy, no $\mathcal{T}$-spaces and the one-vs-all method.



(a) Boxplot for a different number of past observations

(b) Histogram for a different number of past observations

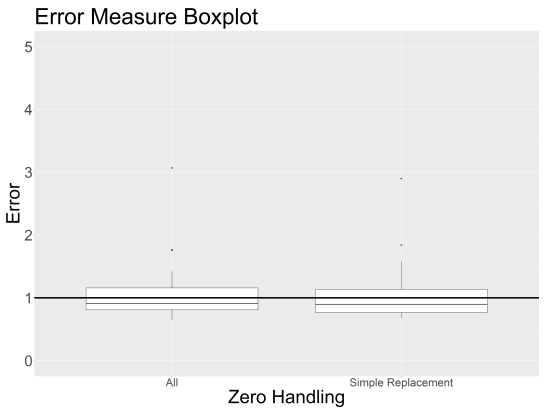


(c) Quantiles for a different number of past observations

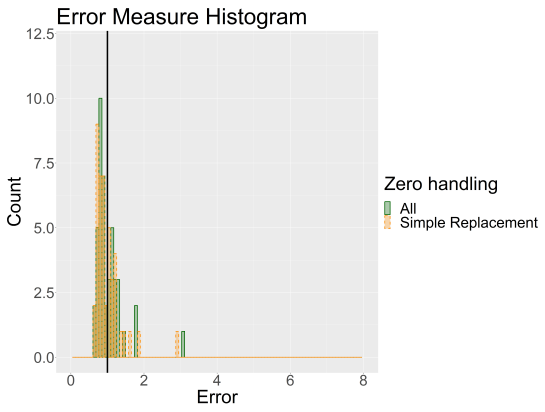
Figure 4.13.: Comparison of a different number of past observations

4.4.5. Zero Handling

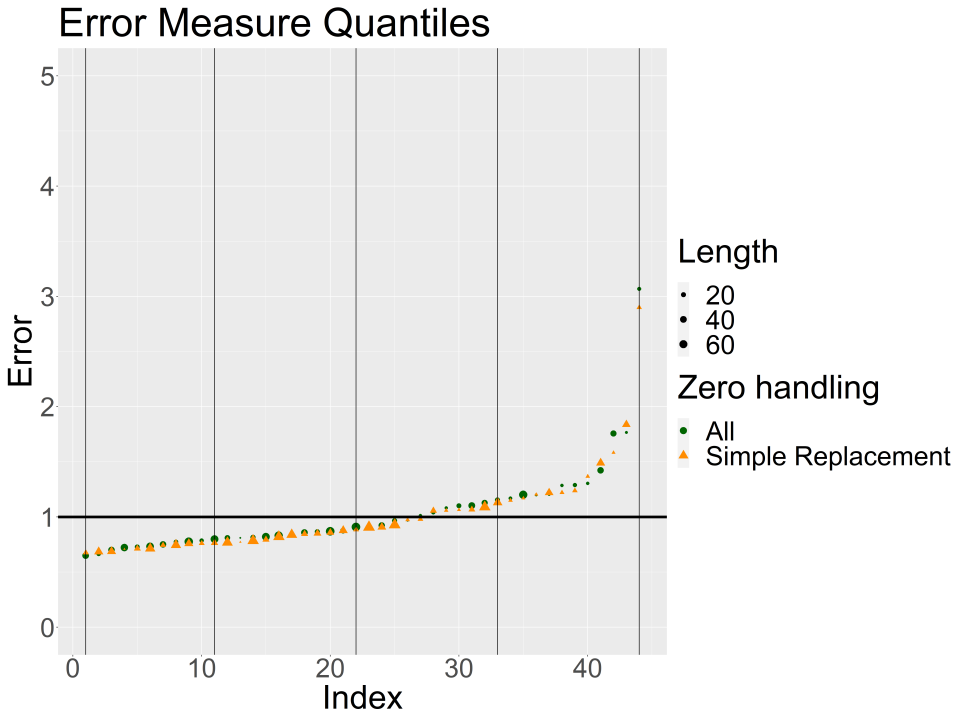
First we compare the different options of handling zeros as explained in 4.1.1. The results are shown in figure 4.14. It seems that the simple replacement strategy with $\delta_j = 0.5, \forall j$ results in marginally better performance.



(a) Boxplot for a different methods of zero handling



(b) Histogram for a different methods of zero handling

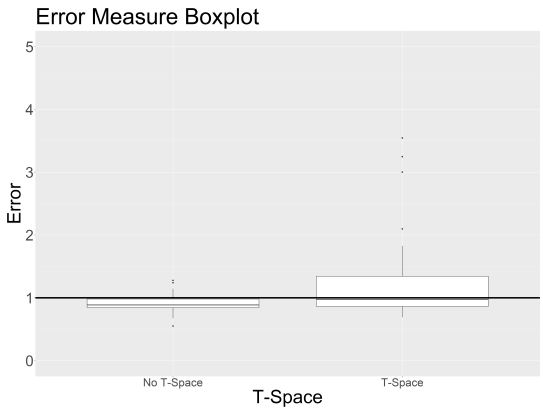


(c) Quantiles for a different methods of zero handling

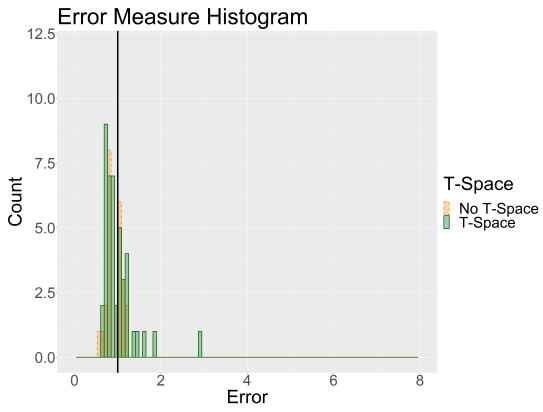
Figure 4.14.: Comparison of different zero handling methods

4.4.6. $\mathcal{T}$-spaces

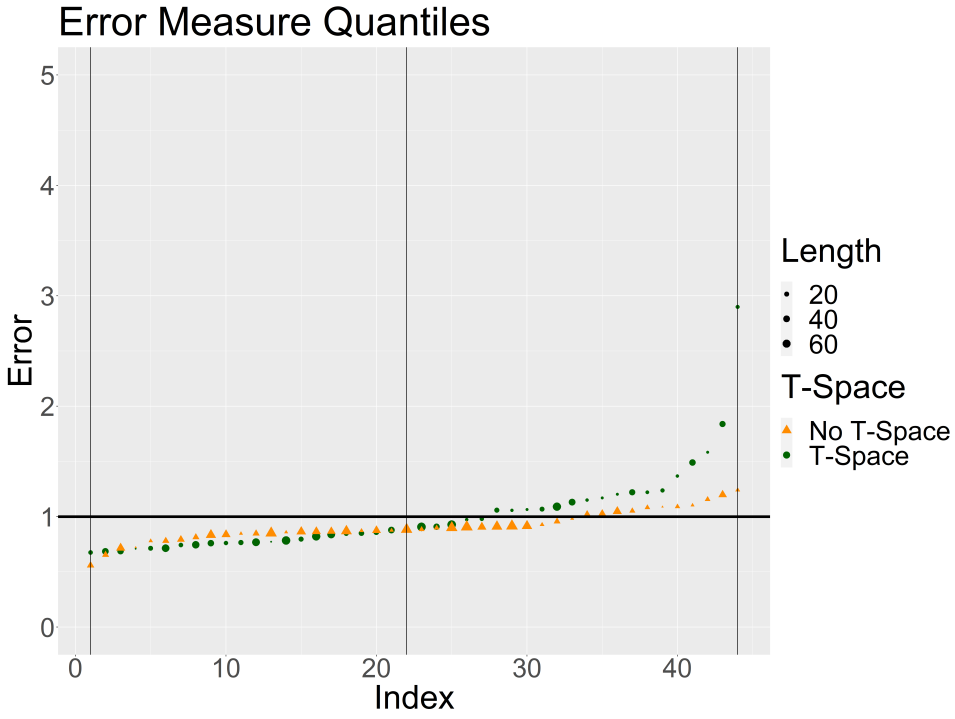
Next we compare CoDA for $\mathcal{T}-$Spaces. The results are shown in 4.15. It seems that using no $\mathcal{T}-$Spaces result in slightly better results. Especially for shorter time series using no $\mathcal{T}-$Spaces returns better results. This can be seen in figure 4.15c



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



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

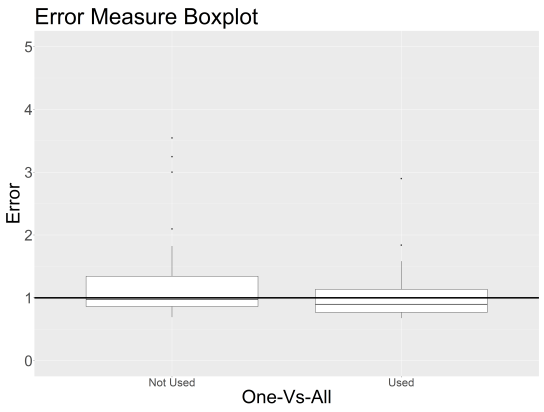


(c) Quantiles for $\mathcal{T}-$Spaces

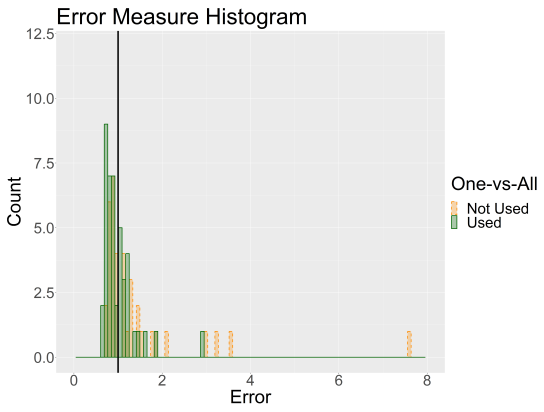
Figure 4.15.: Comparison of CoDA with and without $\mathcal{T}-$Spaces

4.4.7. One-vs-All method

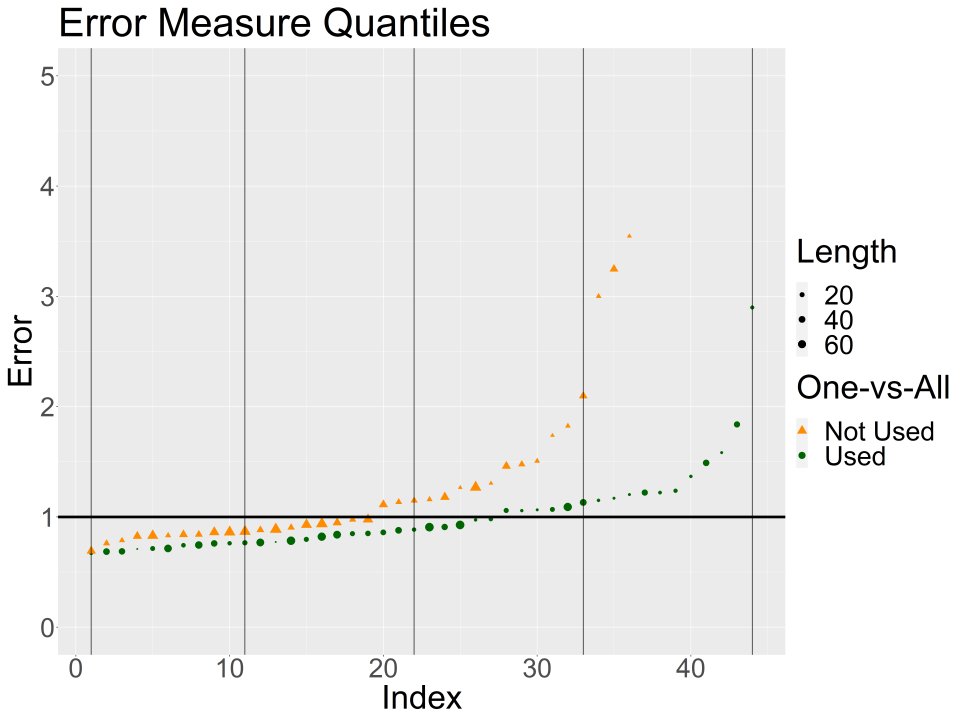
Now we analyse the one-vs-all method. Figure 4.16 shows the results. We can clearly seen, that the one-vs-all method performs better over all time series. This difference is highlighted in figure 4.16c.



(a) Boxplot for One-vs-All



(b) Histogram for One-vs-All



(c) Quantiles for One-vs-All

Figure 4.16.: Comparison of CoDA with and without One-vs-All

Bibliography

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

Aitchison, J. and J.W. Kay. 'Possible solution of some essential zero problems in compositional data analysis'. In: (Jan. 2003).

Alzaid, A. A. and M. Al-Osh. 'An Integer-Valued pth-Order Autoregressive Structure (INAR(p)) Process'. In: *Journal of Applied Probability* 27.2 (1990), pp. 314–324. ISSN: 00219002. URL: http://www.jstor.org/stable/3214650 (visited on 22/05/2023).

'Autoregressive Conditional Heteroscedasticity with Estimates of the Variance of United Kingdom Inflation'. In: *Econometrica* 50.4 (1982), pp. 987–1007. ISSN: 00129682, 14680262. URL: http://www.jstor.org/stable/1912773 (visited on 20/05/2023).

Berndt, Ernst et al. 'Estimation and Inference in Non-Linear Structural Models'. In: *Annals of Economic and Social Measurement* 4 (Jan. 1974).

Biswas, Atanu and Peter X.-K. Song. 'Discrete-valued ARMA processes'. In: *Statistics & Probability Letters* 79.17 (2009), pp. 1884–1889. ISSN: 0167-7152. DOI: https://doi.org/10.1016/j.spl.2009.05.025. URL: https://www.sciencedirect.com/science/article/pii/S0167715209001977.

Bollerslev, Tim. 'Generalized autoregressive conditional heteroskedasticity'. In: *Journal of Econometrics* 31.3 (1986), pp. 307–327. ISSN: 0304-4076. DOI: https://doi.org/10.1016/0304-4076(86)90063-1. URL: https://www.sciencedirect.com/science/article/pii/0304407686900631.

Boogaart, K. Gerald van den, Raimon Tolosana-Delgado and Matevz Bren. 'Concepts for handling of zeros and missing values in compositional data'. In: 2006.

Brännäs, Kurt. 'Estimation and Testing in Integer-valued AR(1) Models'. In: *Umeå Economic Studies* (Jan. 1993).

Chang, Tiao J., J.W. Delleur and M.L. Kavvas. 'Application of Discrete Autoregressive Moving Average models for estimation of daily runoff'. In: *Journal of Hydrology* 91.1 (1987), pp. 119–135. ISSN: 0022-1694. DOI: https://doi.org/10.1016/0022-

1694(87)90132-6. URL: https://www.sciencedirect.com/science/article/pii/0022169487901326.

Cui, Yan and Fukang Zhu. 'A new bivariate integer-valued GARCH model allowing for negative cross-correlation'. In: *TEST: An Official Journal of the Spanish Society of Statistics and Operations Research* 27.2 (June 2018), pp. 428–452. DOI: 10.1007/s11749-017-0552-4. URL: https://ideas.repec.org/a/spr/testjl/v27y2018i2d10.1007_s11749-017-0552-4.html.

Dempster, A. P., N. M. Laird and D. B. Rubin. 'Maximum Likelihood from Incomplete Data Via the EM Algorithm'. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1 (1977), pp. 1–22. DOI: https://doi.org/10.1111/j.2517-6161.1977.tb01600.x. eprint: https://rss.onlinelibrary.wiley.com/doi/pdf/10.1111/j.2517-6161.1977.tb01600.x. URL: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1977.tb01600.x.

Douc, R., P. Doukhan and E. Moulines. 'Ergodicity of observation-driven time series models and consistency of the maximum likelihood estimator'. In: *Stochastic Processes and their Applications* 123.7 (2013). A Special Issue on the Occasion of the 2013 International Year of Statistics, pp. 2620–2647. ISSN: 0304-4149. DOI: https://doi.org/10.1016/j.spa.2013.04.010. URL: https://www.sciencedirect.com/science/article/pii/S0304414913001051.

Doukhan, Paul, Konstantinos Fokianos and Dag Tjøstheim. 'On weak dependence conditions for Poisson autoregressions'. In: *Statistics & Probability Letters* 82.5 (2012), pp. 942–948. ISSN: 0167-7152. DOI: https://doi.org/10.1016/j.spl.2012.01.015. URL: https://www.sciencedirect.com/science/article/pii/S0167715212000259.

Egozcue, Juan Jose and Vera Pawlowsky-Glahn. 'Groups of Parts and Their Balances in Compositional Data Analysis'. In: *Mathematical Geology* 37 (Oct. 2005), pp. 795–828. DOI: 10.1007/s11004-005-7381-9.

Egozcue, Juan Jose et al. 'Isometric Logratio Transformations for Compositional Data Analysis'. In: *Mathematical Geology* 35 (Apr. 2003), pp. 279–300. DOI: 10.1023/A:1023818214614.

Ferland, René, Alain Latour and Driss Oraichi. 'Integer-Valued GARCH Process'. In: *Journal of Time Series Analysis* 27.6 (2006), pp. 923–942. DOI: https://doi.org/10.1111/j.1467-9892.2006.00496.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-9892.2006.00496.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9892.2006.00496.x.

Filzmoser, Peter and Karel Hron. '2.30 - Compositional Data Analysis in Chemometrics'. In: *Comprehensive Chemometrics (Second Edition)*. Ed. by Steven Brown, Romà Tauler and Beata Walczak. Second Edition. Oxford: Elsevier, 2020, pp. 641–662. ISBN: 978-0-444-64166-3. DOI: https://doi.org/10.1016/B978-0-12-409547-2.14591-3. URL: https://www.sciencedirect.com/science/article/pii/B9780124095472145913.

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

Fokianos, Konstantinos. 'Multivariate Count Time Series Modelling'. In: *Econometrics and Statistics* (2021). ISSN: 2452-3062. DOI: https://doi.org/10.1016/j.ecosta.2021.11.006. URL: https://www.sciencedirect.com/science/article/pii/S2452306221001374.

Fokianos, Konstantinos, Anders Rahbek and Dag Tjøstheim. 'Poisson Autoregression'. In: *Journal of the American Statistical Association* 104.488 (2009), pp. 1430–1439. DOI: 10.1198/jasa.2009.tm08270. eprint: https://doi.org/10.1198/jasa.2009.tm08270. URL: https://doi.org/10.1198/jasa.2009.tm08270.

Fokianos, Konstantinos and Dag Tjøstheim. 'Log-linear Poisson autoregression'. In: *Journal of Multivariate Analysis* 102.3 (2011), pp. 563–578. ISSN: 0047-259X. DOI: https://doi.org/10.1016/j.jmva.2010.11.002. URL: https://www.sciencedirect.com/science/article/pii/S0047259X10002320.

Fokianos, Konstantinos et al. 'Multivariate count autoregression'. In: *Bernoulli* 26.1 (2020), pp. 471–499. DOI: 10.3150/19-BEJ1132. URL: https://doi.org/10.3150/19-BEJ1132.

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

Harvey, A. C. and C. Fernandes. 'Time Series Models for Count or Qualitative Observations'. In: *Journal of Business & Economic Statistics* 7.4 (1989), pp. 407–417. ISSN: 07350015. URL: http://www.jstor.org/stable/1391639 (visited on 04/04/2023).

Heinen, Andreas. *Modelling Time Series Count Data: An Autoregressive Conditional Poisson Model*. MPRA Paper 8113. University Library of Munich, Germany, July 2003. URL: https://ideas.repec.org/p/pra/mprapa/8113.html.

Hong, Yongmiao and Yoon-Jin Lee. 'A General Approach to Testing Volatility Models in Time Series'. In: *Journal of Management Science and Engineering* 2.1 (2017), pp. 1–33. ISSN: 2096-2320. DOI: https://doi.org/10.3724/SP.J.1383.201001. URL: https://www.sciencedirect.com/science/article/pii/S2096232019300162.

Jin-Guan, Du and Li Yuan. 'THE INTEGER-VALUED AUTOREGRESSIVE (INAR(p)) MODEL'. In: *Journal of Time Series Analysis* 12.2 (1991), pp. 129–142. DOI: https://doi.org/10.1111/j.1467-9892.1991.tb00073.x. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-9892.1991.tb00073.x. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-9892.1991.tb00073.x.

Kynčlová, Petra, Peter Filzmoser and Karel Hron. 'Modeling Compositional Time Series with Vector Autoregressive Models'. In: *Journal of Forecasting* 34.4 (2015), pp. 303–314. DOI: https://doi.org/10.1002/for.2336. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/for.2336. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/for.2336.

Lambert, Diane. 'Zero-Inflated Poisson Regression, with an Application to Defects in Manufacturing'. In: *Technometrics* 34.1 (1992), pp. 1–14. ISSN: 00401706. URL: http://www.jstor.org/stable/1269547 (visited on 25/05/2023).

Lee, Sangyeol, Dongwon Kim and Byungsoo Kim. 'Modeling and inference for multivariate time series of counts based on the INGARCH scheme'. In: *Computational Statistics & Data Analysis* 177 (2023), p. 107579. ISSN: 0167-9473. DOI: https://doi.org/10.1016/j.csda.2022.107579. URL: https://www.sciencedirect.com/science/article/pii/S0167947322001591.

Leininger, Thomas J. et al. 'Spatial Regression Modeling for Compositional Data With Many Zeros'. In: *Journal of Agricultural, Biological, and Environmental Statistics* 18.3 (2013), pp. 314–334. ISSN: 10857117. URL: http://www.jstor.org/stable/26452944 (visited on 19/05/2023).

Liboschik, Tobias. 'Modeling count time series following generalized linear models'. In: 2016.

Liboschik, Tobias, Konstantinos Fokianos and Roland Fried. 'tscount: An R Package for Analysis of Count Time Series Following Generalized Linear Models'. In: *Journal of Statistical Software* 82.5 (2017), pp. 1–51. DOI: 10.18637/jss.v082.i05.

Liboschik, Tobias et al. *tscount: Analysis of Count Time Series*. R package version 1.4.3. 2020. URL: https://CRAN.R-project.org/package=tscount.

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

Lubbe, Sugnet, Peter Filzmoser and Matthias Templ. 'Comparison of zero replacement strategies for compositional data with large numbers of zeros'. In: *Chemometrics and Intelligent Laboratory Systems* 210 (2021), p. 104248. ISSN: 0169-7439. DOI: https://doi.org/10.1016/j.chemolab.2021.104248. URL: https://www.sciencedirect.com/science/article/pii/S0169743921000162.

Lütkepohl, H. *New Introduction to Multiple Time Series Analysis*. Springer Berlin Heidelberg, 2007. ISBN: 9783540262398. URL: https://books.google.at/books?id=muorJ6FHIiEC.

Macdonald, Lain, Iain L. MacDonald and Iain L. MacDonald. 'Hidden Markov and Other Models for Discrete- valued Time Series'. In: 1997.

Martín-Fernández, Josep Antoni, C. Barceló-Vidal and Vera Pawlowsky-Glahn. 'Dealing with Zeros and Missing Values in Compositional Data Sets Using Nonparametric Imputation'. In: *Mathematical Geosciences* 35.3 (2003), pp. 253–278. DOI: 10.1023/A:1023866030544.

Müller, Kirill. *here: A Simpler Way to Find Your Files*. R package version 1.0.1. 2020. URL: https://CRAN.R-project.org/package=here.

Palarea-Albaladejo, Javier and Josep Antoni Martín-Fernández. 'zCompositions — R package for multivariate imputation of left-censored data under a compositional approach'. In: *Chemometrics and Intelligent Laboratory Systems* 143 (2015), pp. 85–96. ISSN: 0169-7439. DOI: https://doi.org/10.1016/j.chemolab.2015.02.019. URL: https://www.sciencedirect.com/science/article/pii/S0169743915000490.

Pawlowsky-Glahn, Vera, Juan Jose Egozcue and David Lovell. 'The product space T (tools for compositional data with a total)'. In: *Proceedings of the 5th International Workshop on Compositional Data Analysis* (June 2013), pp. 143–152.

Pfaff, B. *Analysis of Integrated and Cointegrated Time Series with R*. Second. ISBN 0-387-27960-1. New York: Springer, 2008. URL: https://www.pfaffikus.de.

Pfaff, Bernhard. 'VAR, SVAR and SVEC Models: Implementation Within R Package vars'. In: *Journal of Statistical Software* 27.4 (2008). URL: https://www.jstatsoft.org/v27/i04/.

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2022. URL: https://www.R-project.org/.

Robitzsch, Alexander and Simon Grund. *miceadds: Some Additional Multiple Imputation Functions, Especially for 'mice'*. R package version 3.16-18. 2023. URL: https://CRAN.R-project.org/package=miceadds.

Scherrer, W. ‘Stationäre Prozesse und Zeitreihenanalyse’. Mar. 2021.

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

Silva, Maria and Vera Oliveira. ‘Difference Equations for the Higher Order Moments and Cumulants of the INAR(p) Model’. In: *Journal of Time Series Analysis* 26 (Jan. 2005), pp. 17–36. DOI: 10.1111/j.1467-9892.2005.00388.x.

Silva, Nélia, Isabel Pereira and Maria Silva. ‘Forecasting in INAR (1) model’. In: *REVSTAT – Statistical Journal Volume* 7 (May 2009), pp. 119–134. DOI: 10.57805/revstat.v7i1.77.

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

van den Boogaart, K. Gerald, Raimon Tolosana-Delgado and Matevz Bren. *compositions: Compositional Data Analysis*. R package version 2.0-6. 2023. URL: https://CRAN.R-project.org/package=compositions.

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

Wood, Simon N. ‘Stable and Efficient Multiple Smoothing Parameter Estimation for Generalized Additive Models’. In: *Journal of the American Statistical Association* 99.467 (2004), pp. 673–686. ISSN: 01621459. URL: http://www.jstor.org/stable/27590439 (visited on 30/04/2023).

Woodard, Dawn B., David S. Matteson and Shane G. Henderson. ‘Stationarity of generalized autoregressive moving average models’. In: *Electronic Journal of Statistics* 5.none (2011), pp. 800–828. DOI: 10.1214/11-EJS627. URL: https://doi.org/10.1214/11-EJS627.

Yee, T. W. and C. J. Wild. ‘Vector Generalized Additive Models’. In: *Journal of the Royal Statistical Society. Series B (Methodological)* 58.3 (1996), pp. 481–493. ISSN: 00359246. URL: http://www.jstor.org/stable/2345888 (visited on 30/04/2023).

Yee, Thomas. *Vector Generalized Linear and Additive Models: With an Implementation in R*. Jan. 2015, pp. 1–589. ISBN: 978-1-4939-2817-0. DOI: 10.1007/978-1-4939-2818-7.

Yee, Thomas W. ‘The VGAM Package for Categorical Data Analysis’. In: *Journal of Statistical Software* 32.10 (2010), pp. 1–34. DOI: 10.18637/jss.v032.i10.

Zeger, Scott L. 'A Regression Model for Time Series of Counts'. In: *Biometrika* 75.4 (1988), pp. 621–629. ISSN: 00063444. URL: http://www.jstor.org/stable/2336303 (visited on 04/04/2023).

Zeileis, Achim, Christian Kleiber and Simon Jackman. 'Regression Models for Count Data in R'. In: *Journal of Statistical Software* 27.8 (2008). URL: http://www.jstatsoft.org/v27/i08/.

Zhu, Fukang. 'Zero-inflated Poisson and negative binomial integer-valued GARCH models'. In: *Journal of Statistical Planning and Inference* 142.4 (2012), pp. 826–839. ISSN: 0378-3758. DOI: https://doi.org/10.1016/j.jspi.2011.10.002. URL: https://www.sciencedirect.com/science/article/pii/S037837581100379X.

Zivot, Eric. 'Practical Issues in the Analysis of Univariate GARCH Models'. In: *Handbook of Financial Time Series*. Ed. by Thomas Mikosch et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 113–155. ISBN: 978-3-540-71297-8. DOI: 10.1007/978-3-540-71297-8_5. URL: https://doi.org/10.1007/978-3-540-71297-8_5.

List of Figures

List of Tables

Appendix A.

R-Functions