Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Time Series: A Comparative Study of Forecasting Methods

Bachelor's Thesis

Wayne Zeng

zengw@ethz.ch

D-MATH
ETH Zürich

Supervisors:
Fadoua Balabdaoui

March 2021

Acknowledgements

I would like to thank my supervisor, Fadoua Balabdaoui, for her generous help and support over the last few months. The fast, precise communication we were able to maintain despite the health situation was invaluable for my progress and understanding throughout the writing of this thesis, and is greatly appreciated.

Abstract

We examine two types of forecasting algorithms: seasonal ARIMA and Holt–Winters, as well as their theoretical backgrounds. We forecast the data from the Swiss Weekly Deaths (65+) dataset, training on the data from 2010-2018 and testing on that of 2019. We lastly conclude that the all three methods produce good forecasts by encapturing seasonal phenomena, and that the automatically determined seasonal ARIMA model performs the best with respect to the RMSE. However, none of them are able to forecast the effects of COVID-19 on the number of deaths in the elderly in 2020.

Contents

Chapter 1

Introduction

We would like to discuss several methods used to forecast time series, which are time-dependent processes seen in a variety of applications in the fields of engineering, natural sciences, economics and finance. In particular, we will focus on a special type of process called ARIMA (autoregressive integrated moving-average) processes.

In this chapter, we introduce the basic concepts of time series analysis, such as the notions of stationarity, autocovariance and -correlation functions, as well as trend and seasonality.

We will continue in Chapter 2 to discuss ARMA processes, a family of time series which form the basis for the rest of the paper and are a special case of ARIMA processes. We will also see the prediction operator $P(\cdot|W)$ and the innovations algorithm, an important process which is also used in other forecasting methods. Lastly, we will examine how to fit ARMA models and how to determine their parameters.

Chapter 3 then introduces ARIMA processes and their seasonal counterpart, and introduces a general way of forecasting future values by expanding on the methods explored in Chapter 2.

In Chapter 4, we investigate the Holt–Winters algorithm and the methods it is based on.

Finally, we will apply the theory learned to a real life example in Chapter 5. We will be using the Swiss Weekly Deaths (65+) dataset from 2010 to 2020 and examine how each method performs and forecasts future values. The appendix contains R code for each of the methods.

Much of Chapters 1-4 closely follows Brockwell and Davis "Introduction to Time Series and Forecasting" [1].

1.1 Introductory Terms

A time series is defined as a sequence of events observed over time. We would like to deduce the serial correlation and relationships between the events, and possibly forecast future values based on them. In this paper, we will only consider *discrete* time series, i.e. ones which are defined on discrete time sets.

Definition 1.1. A *time series model* for data $\{X_t\}$ is a specification of the joint distributions (or possibly only the means and covariances) of a sequence of random variables $\{X_t\}$.

We shall consider a few basic concepts of time series, all of which we will see in great abundance in later chapters.

Definition 1.2. A sequence of independent identically distributed random variables such that $\mathbb{E}[X_t] = 0$ for all t is called *i.i.d. (independent and identically distributed) noise.*

Definition 1.3. Let $\{Z_t\}$ be a sequence of random variables with variance σ^2. If $\{Z_t\}$ are uncorrelated with zero mean, i.e.

$$\mathbb{E}[Z_t] = 0, \quad \operatorname{Cov}(Z_r, Z_s) = 0 \quad \forall r \neq s,$$

we call $\{Z_t\}$ *white noise* and use the notation

$$\{Z_t\} \sim \mathrm{WN}(0, \sigma^2).$$

White noise provides the random component in numerous time series models such as the classical decomposition method, ARMA and ARIMA models [2]. Note that all i.i.d. noises are white noise, but not vice versa.

1.2 Stationarity

Many useful analytical tools rely on the concept of stationarity. In essence, stationarity means that the statistical properties of the process generating our observed time series do not change over time [3]. We shall begin with a few definitions.

Definition 1.4. Let $\{X_t\}$ be a time series with $\mathbb{E}[X_t^2] < \infty$. We define the *mean function* of $\{X_t\}$ as

$$\mu_X(t) = \mathbb{E}[X_t], \tag{1.1}$$

and its *covariance function* as

$$\gamma_X(r, s) = \operatorname{Cov}(X_r, X_s), \quad r, s \in \mathbb{Z}. \tag{1.2}$$

Definition 1.5. $\{X_t\}$ is *(weakly) stationary* if $\mu_X(t)$ is independent of t and $\gamma_X(t+h,t)$ is independent of t for each h.

Definition 1.6. $\{X_t\}$ is *strictly stationary* if $(X_1,\dots,X_n)$ and $(X_{1+h},\dots,X_{n+h})$ have the same joint distributions for all $h \in \mathbb{Z}$ and $n > 0$.

Throughout this paper, we will be referring to the term "stationary" as *weakly* stationary.

Properties of Stationary Time Series

1. The random variables X_t are identically distributed.
2. $(X_t, X_{t+h})' \overset{\mathrm{d}}{=} (X_1, X_{1+h})$.
3. $\{X_t\}$ is weakly stationary if $\mathbb{E}[X_t^2] < \infty$ for all t.
4. Weak stationarity does not imply strict stationarity.
5. An i.i.d. sequence is strictly stationary.
6. If $\{X_t\}$ is strictly stationary and $\mathbb{E}[X_t^2] < \infty$ for all t, then $\{X_t\}$ is also weakly stationary.

A proof of these properties can be found on p.43 in [1].

1.2.1 Autocovariance and Autocorrelation

The idea of autocovariance and -correlation is to observe the covariance and correlation respectively of a process with a delayed copy (lag) of itself. We can analyse these to then more easily observe whether the data at hand is stationary or not (cf. Section 2.2).

Definition 1.7. The *autocovariance function* (ACVF) of $\{X_t\}$ at lag h is

$$\gamma_X(h) \equiv \gamma_X(h,0) = \mathrm{Cov}(X_t, X_{t+h}). \tag{1.3}$$

The *autocorrelation function* (ACF) is then defined as

$$\rho_X(h) = \mathrm{Cor}(X_{t+h}, X_t) \equiv \frac{\gamma_X(h)}{\gamma_X(0)}. \tag{1.4}$$

Theorem 1.8. *Some basic properties of the ACVF include:*

1. $\gamma(0) \geq 0$
2. $|\gamma(h)| \leq \gamma(0)$ *for all* $h \in \mathbb{Z}$
3. $\gamma(h) = \gamma(-h)$ *for all* $h \in \mathbb{Z}$

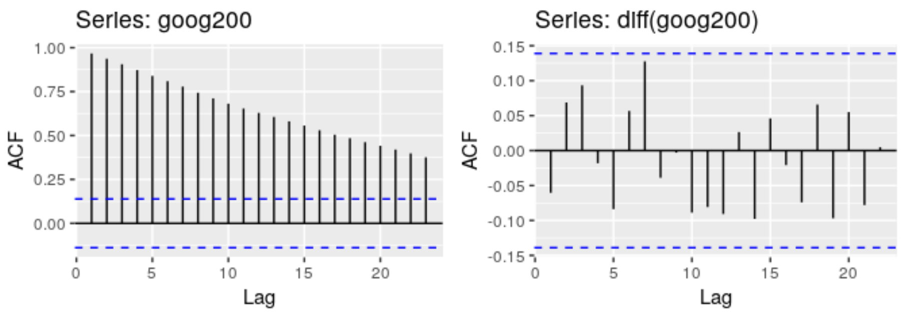

Figure 1.1: The ACF of Google stock prices (left; non-stationary), and its daily price changes (right; stationary). [3]

1.3 Estimation and Elimination of Trend and Seasonal Components

This section is based on Section 1.3.3 of [1].

When forecasting time series, we would like to check whether the following things are present:

(a) trends

(b) seasonal components

(c) sharp changes in behaviour

(d) outliers.

Ideally, by removing the trend and seasonal components, the residuals i.e. remaining errors should be stationary white noise, meaning that all of the signal information has been analysed already and only random fluctuations remain. If the model predictions are more than just white noise, this signifies that further improvements can be made.

Let us consider the *classical decomposition model*

$$X_t = m_t + s_t + Y_t, \tag{1.5}$$

where m_t is the *trend component* (a more slowly changing function), s_t the *seasonal component* with period d, and Y_t being a weakly stationary *noise component*, i.e.

$$\mathbb{E}[Y_t] = 0, \quad s_{t+d} = s_t, \text{ and } \sum_{j=1}^{d} s_j = 0.$$

There are several methods of estimating the trend and seasonality, such as smoothing methods like using a finite moving average filter and exponential smoothing (which we will see in Chapter 5 in more detail); as well as the differencing methods. For the latter, we shall introduce the difference operators:

Definition 1.9. The *lag-1 difference operator* ∇ is defined as:

$$\nabla X_t = X_t - X_{t-1} = (1-B)X_t. \tag{1.6}$$

Where the *backward shift operator* B is defined as:

$$BX_t = X_{t-1}. \tag{1.7}$$

and the *lag-d differencing operator* ∇_d is defined as

$$\nabla_d X_t = (1-B^d)X_t. \tag{1.8}$$

Remark 1.10. Note that ∇_d is not the same as ∇^d, which is defined as $\nabla^d = (1-B)^d$.

Differencing a time series d times may result in a stationary time series, allowing us to apply the methods from later chapters which rely on their useful properties.

1.4 Estimating the Noise Sequence

This section is based on Sections 1.4.1 and 1.6 of [1].

Once we have transformed the time series data and isolated the stationary components Y_t, the next step is to model the residuals, or "estimated noise sequence". If these residuals are independent to each other, it remains for us to estimate the mean and variance, and no further action is required. However, if there is a significant level of dependence, we need to consider more complex stationary time series models for the noise. In fact, this is helpful, since dependence between past and present noise terms is beneficial for forecasting future ones.

1.4.1 Sample Autocorrelation Method

We will present one method of testing the residuals, namely the sample autocorrelation function method. Further methods can be found in [1], Section 1.6.

Definition 1.11. For a time series $\{X_t\}$, the *sample autocovariance function* is defined as

$$\hat{\gamma}(h) = \frac{1}{n}\sum_{t=1}^{n-|h|}(X_{t+|h|}-\bar{X})(X_t-\bar{X}),$$

and its *sample autocorrelation function* is similarly defined as

$$\hat{\rho}(h) = \frac{\hat{\gamma}(h)}{\hat{\gamma}(0)} \quad \text{for } -n < h < n.$$

For $\{X_t\}$ i.i.d. noise, it can be proven that for $n \to \infty$,

$$\sqrt{n}\hat{\boldsymbol{\rho}} \xrightarrow{d} N(\mathbf{0}, Id_n) \text{ with } \hat{\boldsymbol{\rho}} := (\hat{\rho}(1), \hat{\rho}(2), \dots, \hat{\rho}(h))^T.$$

Let us now define the quantity

$$N_h := \sum_{i=1}^{h} \mathbb{1}_{\{|\hat{\rho}(i)| \geq \frac{Z_{1-\alpha/2}}{\sqrt{n}}\}},$$

where $Z_{1-\alpha/2}$ denotes the $(1-\alpha/2)$th quantile of $N(0,1)$.

It holds $N_h \xrightarrow{d} \text{Bin}(h, \alpha)$ as $n \to \infty$, since by "almost independence" of $\hat{\rho}(1), \dots, \hat{\rho}(h)$:

$$P(|\hat{\rho}(1)| \geq \frac{Z_{1-\alpha/2}}{\sqrt{n}}) \xrightarrow[n\to\infty]{} P(|Z| \geq Z_{1-\alpha/2}) = \alpha.$$

Hence, $\mathbb{E}[N_h] \approx h\alpha$ and $\text{Var}(N_h) \approx h\alpha(1-\alpha)$ for large n.

This observation on N_h allows us to perform a hypothesis test on the assumption that the noise is i.i.d.; if it is not the case, then we can expect the number of sample autocorrelations to be larger than $\mathbb{E}[N_h]$.

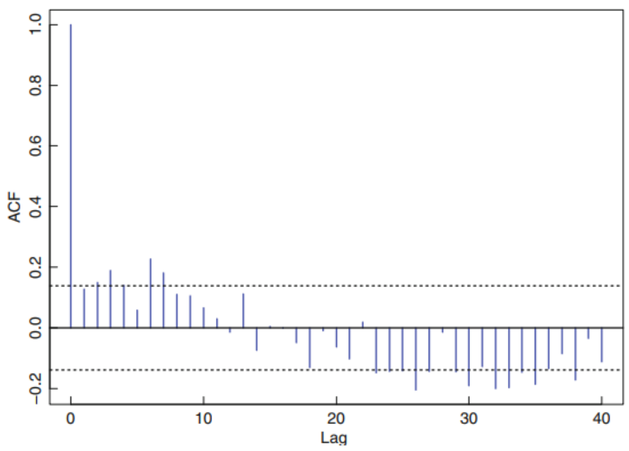

Figure 1.2: The sample autocorrelation function for $X_t = \cos t/10 + N_t$ for $t = 1, \dots, 200$ and N_t a sequence of independent normal variables.([1], fig. 1-28).

Chapter 2

ARMA Processes

We will discuss the properties of ARMA processes, conditions showing when the process has a stationary solution, the notion of causality and lastly how to forecast them. As well as being based on Chapters 2, 3 and 5 of [1], some parts of this chapter has been based on [4].

Autoregressive moving-average (ARMA) processes are an important family of models used mainly to model stationary time series. We will see in later sections that they are simple to forecast.

We will first begin with a more generalised class of processes (which ARMA itself is a type of) called *linear processes*. These provide us with a framework for analysing stationary processes.

Definition 2.1. A time series $\{X_t\}$ is a *linear process* if it can be written as

$$X_t = \sum_{j=-\infty}^{\infty} \psi_j Z_{t-j} \quad \forall t, \tag{2.1}$$

where $\{Z_t\} \sim \mathrm{WN}(0, \sigma^2)$, and $\{\psi_j\}$ is a real sequence of constants with $\sum_{j=-\infty}^{\infty} |\psi_j| < \infty$.

Definition 2.2. A time series $\{X_t\}$ is said to be an ARMA(p, q) stationary process if $\{X_t\}$ is stationary, and if for every t,

$$\phi(B)X_t = \theta(B)Z_t, \tag{2.2}$$

where $Z_t \sim \mathrm{WN}(0, \sigma^2)$, the polynomials $\phi(z) = (1 - \phi_1 z - \cdots - \phi_p z^p)$ and $\theta(z) = (1 + \theta_1 z + \cdots + \theta_q z^q)$ have no common factors, and B denotes the backward shift operator.

An ARMA process possesses a *stationary solution* if we can write X_t purely in terms of Z_t. This solution can almost surely be unique, as we will see in the theorems below.

Definition 2.3. A time series $\{X_t\}$ is said to be an *autoregressive process of order p* (AR(p)) if $\theta(z) \equiv 1$, and a *moving-average process of order q* (MA(q)) if $\phi(z) \equiv 1$. An ARMA process is a mixed model, containing p AR terms and q MA terms.

2.1 Causality and Invertibility

This section is based on Section 3.1 of [1].

ARMA(p, q) processes can be causal or invertible. These two properties allow us to write a solution of the ARMA(p, q) process in terms of only past or future values, making it easier to find a solution as well as being able to say more about uniqueness of the solution. Furthermore, causality enables us to conveniently find the ACVF of the ARMA process and shorten the application of the innovations algorithm (see Section 2.4).

Definition 2.4. A time series $\{X_t\}$ is a *causal solution* if the linear process can be written as an expression of only current and past values of the white noise term $\{Z_t\}$, i.e.

$$X_t = \sum_{j=0}^{\infty} \psi_j Z_{t-j} \quad \forall t \in \mathbb{Z}. \tag{2.3}$$

$\{X_t\}$ is a *non-causal solution* if it can be written only using current and future values of $\{Z_t\}$:

$$X_t = \sum_{j=0}^{\infty} \psi_j Z_{t+j} \quad \forall t \in \mathbb{Z}.$$

A parallel notion to causality is invertibility:

Definition 2.5. A white noise $\{Z_t\}$ is an *invertible solution* if it can be written as an expression of only current and past values of the time series events $\{X_t\}$:

$$Z_t = \sum_{j=0}^{\infty} \pi_j X_{t-j} \quad \forall t \in \mathbb{Z}.$$

Analogously, a *non-invertible solution* of $\{Z_t\}$ is defined as

$$Z_t = \sum_{j=0}^{\infty} \pi_j X_{t+j} \quad \forall t \in \mathbb{Z},$$

i.e. it can be written purely as present and future values of X_t.

The coefficients ψ_j and π_j can be found by Taylor-expanding $1/\phi(z)$ and $1/\theta(z)$ accordingly, in particular, for causal $\{X_t\}$, we can write the coefficients ψ_j as (see [1], section 2.3)

$$\psi_j = \theta_j + \sum_{i=1}^{\min(j,p)} \phi_i \psi_{j-i}, \quad j = 0, 1, \ldots, \tag{2.4}$$

and we define $\theta_0 := 1$ and $\theta_j := 0$ for $j > q$.

It is important to determine whether a stationary solution of the ARMA equations exists and whether $\{X_t\}$ is causal, which is illustrated in the following theorem.

Theorem 2.6. *If ϕ and θ as defined in (2.2) do not admit any common factors, then a stationary solution of the equations $\phi(B)X_t = \theta(B)Z_t$ only exists if the AR polynomial ϕ does not admit a root on the unit circle.*

Furthermore, $\{X_t\}$ is causal if and only if ϕ does not admit any roots in the unit disc.

This theorem can be proven using properties of the power series, and a proof can be found in the HS20 course "Time Series Analysis" lecture notes (Lecture 11) by F. Balabdaoui.

2.2 The ACVF and PACF of ARMA Processes

This section is based on Section 3.2 of [1].

2.2.1 ACVF

Proposition 2.7. *Let $\{Y_t\}$ be a stationary zero-mean time series with covariance function γ_Y. If $\sum_{j=-\infty}^{\infty} |\psi_j| < \infty$, then the transformed time series*

$$X_t = \sum_{j=-\infty}^{\infty} \psi_j Y_{t-j} = \psi(B)Y_t$$

is itself a stationary zero-mean time series with ACVF

$$\gamma_X(h) = \sum_{j=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} \psi_j \psi_k \gamma_Y(h+k-j).$$

Specifically, for a linear process $\{X_t\}$, the above expression is simplified to

$$\gamma_X(h) = \sum_{j=-\infty}^{\infty} \psi_j \psi_{j+h} \sigma^2, \tag{2.5}$$

where σ^2 is the variance of the white noise component $\{Z_t\}$.

Proof. Cf. p.45 in [1]. □

For causal ARMA(p,q) processes

$$X_t = \sum_{j=0}^{\infty} \psi_j Z_{t-j},$$

where $\{\psi_j\}$ is calculated in [1], section 2.3, $\sum_{j=0}^{\infty} \psi_j z^j = \theta(z)/\phi(z)$, $|z| \le 1$, there are several ways to calculate their ACVFs $\gamma(h)$. For example, we can use (2.5) to say that $\gamma_X(h) = \sum_{j=0}^{\infty} \psi_j \psi_{j+|h|} \sigma^2$.

The significance of considering the ACVF and ACF respectively lies with the moving-average process MA(q). From (2.5), we can deduce, for an MA(q) process

$$X_t = Z_t + \theta_1 Z_{t-1} + \cdots + \theta_q Z_{t-q}, \quad \{Z_t\} \sim \mathrm{WN}(0, \sigma^2)$$

with ACVF

$$\gamma(h) = \begin{cases} \sigma^2 \sum_{j=0}^{q-|h|} \theta_j \theta_{j+|h|}, & \text{if } |h| \le q, \\ 0, & \text{if } |h| > q, \end{cases}$$

where $\theta_0 := 1$, that its ACVF vanishes at $|h| > q$. We can therefore inspect the ACF of a moving average process to see where it is close to 0. Here, "close to 0" means that the sample ACF values with $h > p$ are approximately distributed as $N(0, 1/n)$ random variables, i.e. they lie within a 95% confidence interval. This property is very useful in order selection of ARMA and even ARIMA models.

2.2.2 PACF

The *partial autocorrelation function* (PACF) $\alpha(\cdot)$ of an ARMA process $\{X_t\}$ is defined by the equations

$$\alpha(0) = 1, \quad \alpha(h) = \phi_{hh}, \quad h \ge 1,$$

where ϕ_{hh} represents the last component of

$$\boldsymbol{\phi}_h = \Gamma_h^{-1} \boldsymbol{\gamma}_h, \quad \gamma_h = [\gamma(i-j)]_{i,j=1}^{h}, \quad \boldsymbol{\gamma}_h = [\gamma(1), \gamma(2), \ldots, \gamma(h)]^T.$$

The *sample PACF* $\hat{\alpha}(h)$ is defined by the relations

$$\hat{\alpha}(0) = 1, \quad \hat{\alpha}(h) = \hat{\phi}_{hh}, \quad \hat{\boldsymbol{\gamma}}_h = \hat{\Gamma}_h^{-1} \hat{\boldsymbol{\gamma}}_h.$$

In a similar fashion as to why one considers the ACF and ACVF of an MA(q) process, it can be shown that, for a purely autoregressive process AR(p), if the sample PACF $\hat{\alpha}(h)$ is much larger/smaller than 0 for $0 \le h \le p$ and close to 0 for $h > p$, then an AR(p) model may be a suitable representative of the data (cf. Example 3.2.6 in [1]).

2.3 Forecasting Stationary Time Series

This section is based on Section 2.5 of [1].

Forecasting, or extrapolation, involves taking models based on historical data and using them to predict future observations. The future data is completely unavailable and can only be estimated from what has already happened. The effectiveness of a forecasting model is determined by several parameters, such as why a specific prediction was made, underlying relationships, and confidence intervals on how accurate the predictor is compared to actual future values.

2.3.1 Derivation of the Linear Prediction Operator

In order to forecast, we would like to firstly consider the linear prediction operator P_n, a special case of the general prediction operator $Y \mapsto P(Y \,|\, \boldsymbol{W})$.

Definition 2.8. The linear prediction operator P_n is defined as

$$P_n X_{n+h} = a_0 + a_1 X_n + \cdots + a_n X_1, \tag{2.6}$$

where $a_0, \dots, a_n$ are such that $P_n X_{n+h}$ achieves the smallest prediction error among all such possible linear combinations.

Theorem 2.9. *The equation (2.6) can be simplified to*

$$P_n X_{n+h} = \mu + \sum_{i=1}^{n} a_i (X_{n+1-i} - \mu), \tag{2.7}$$

where μ is the mean of the time series.

Proof. This proof follows the derivation from Section 2.5 of [1].

In order to determine the coefficients $a_0, a_1, \dots, a_n$, we would like to find the values that minimise the functional S:

$$S(a_0, \dots, a_n) = \mathbb{E}[(X_{n+h} - P_n X_{n+h})^2] = \mathbb{E}[(X_{n+h} - a_0 - a_1 X_n - \cdots - a_n X_1)^2].$$

To do so, we would like to calculate the partial derivatives of S, i.e.

$$\frac{\partial S(a_0, \dots, a_n)}{\partial a_j} = 0 \tag{2.8}$$

$$\mathbb{E}[X_{n+h} - a_0 - \sum_{i=1}^{n} a_i X_{n+1-i}] = 0 \quad j = 0 \tag{2.9}$$

$$\mathbb{E}[(X_{n+h} - a_0 - \sum_{i=1}^{n} a_i X_{n+1-i}) X_{n+1-j}] = 0 \quad j \geq 1 \tag{2.10}$$

Moreover, we can write the set of equations (2.10) in vector form:

$$a_0 = \mu \left(1 - \sum_{i=1}^{n} a_i \right),$$

and

$$\Gamma_n \boldsymbol{a}_n = \boldsymbol{\gamma}_n(h),$$

where we define

$$\begin{aligned}\boldsymbol{a}_n &= (a_1, \dots, a_n)^T, \\ \Gamma_n &= [\gamma(i-j)]_{i,j=1}^n, \text{ and} \\ \boldsymbol{\gamma}_n(h) &= (\gamma(h), \gamma(h+1), \dots, \gamma(h+n-1))^T.\end{aligned}$$

Hence

$$P_n X_{n+h} = \mu + \sum_{i=1}^{n} a_i (X_{n+1-i} - \mu)$$

as required. □

Properties of $P_n X_{n+h}$:

1. $\mathbb{E}[(X_{n+h} - P_n X_{n+h})^2] = \gamma(0) - \boldsymbol{a}_n^T \boldsymbol{\gamma}_n(h)$.
2. $\mathbb{E}[X_{n+h} - P_n X_{n+h}] = 0$
3. $\mathbb{E}[(X_{n+h} - P_n X_{n+h}) X_j] = 0$

These properties are equivalent to (2.9) and (2.10) and can be referred to as the *orthogonality conditions.*

2.3.2 The Prediction Operator and Second-Order Random Variables

In the more general case, let Y and $W_n, \dots, W_1$ be arbitrary random variables defined on the same probability space with finite second moments. Let $\mu = \mathbb{E}[Y]$, $\mu_i = \mathbb{E}[W_i]$ and covariances $\mathrm{Cov}(Y,Y)$, $\mathrm{Cov}(Y, W_i)$ and $\mathrm{Cov}(W_j, W_i)$ be all well defined (implied by the finite second moment property). Furthermore, let us denote

$$\begin{aligned}\boldsymbol{W} &= (W_n, \dots, W_1)^T, \\ \boldsymbol{\mu}_W &= (\mu_n, \dots, \mu_1)^T, \\ \boldsymbol{\gamma} &= \mathrm{Cov}(Y, \boldsymbol{W}) \\ \boldsymbol{\Gamma} &= [\mathrm{Cov}(W_{n+1-i}, W_{n+1-j})]_{j=1}^n.\end{aligned}$$

Using an analogous derivation as in the previous section, we obtain that the best linear predictor of Y in terms of $\{1, \dots, w\}$ is given by:

$$P(Y \,|\, \boldsymbol{W}) = \mu_Y + \boldsymbol{a}^T (W - \mu_W),$$

where $\boldsymbol{a} = (a_1, \dots, a_n)^T$ is defined as a solution of the equation

$$\Gamma \boldsymbol{a} = \boldsymbol{\gamma}$$

with mean squared error $\mathbb{E}[(Y - P(Y \,|\, \boldsymbol{W}))^2] = \mathrm{Var}(Y) - \boldsymbol{a}^T \boldsymbol{\gamma}$.

Definition 2.10. For any given $\boldsymbol{W} = (W_n, \dots, W_1)^T$ and Y with finite second moments, the *prediction operator* $Y \mapsto P(Y|\,\boldsymbol{W})$ is defined as

$$P(Y|\,\boldsymbol{W}) = \mu_Y + \boldsymbol{a}^T(W - \mu_W). \tag{2.11}$$

Letting $\boldsymbol{W} = (X_n, X_{n-1}, \dots, X_1)^T$, we obtain the linear predictor (2.7).

Properties of the Prediction Operator $P(\cdot|W)$:
For U, V with finite second moments, $\Gamma = \operatorname{Cov}(\boldsymbol{W}, \boldsymbol{W})$, and $\beta, \alpha_1, \dots, \alpha_n$ constants, the following facts hold:

1. $P(U|\,\boldsymbol{W}) = \mathbb{E}[U] + \boldsymbol{a}^T(\boldsymbol{W} - \mathbb{E}[\boldsymbol{W}])$, where $\Gamma\boldsymbol{a} = \operatorname{Cov}(U, \boldsymbol{W})$.
2. $\mathbb{E}[(U - P(U|\,\boldsymbol{W}))\,\boldsymbol{W}] = \boldsymbol{0}$, and $\mathbb{E}[(U - P(U|\,\boldsymbol{W}))] = 0$.
3. $\mathbb{E}[(U - P(U|\,\boldsymbol{W}))^2] = \operatorname{Var}(U) - \boldsymbol{a}^T \operatorname{Cov}(U, \boldsymbol{W})$.
4. $P(\alpha_1 U + \alpha_2 V + \beta|\,\boldsymbol{W}) = \alpha_1 P(U|\,\boldsymbol{W}) + \alpha_2 P(V|\,\boldsymbol{W}) + \beta$.
5. $P(\sum_{i=1}^n \alpha_i W_i + \beta|\,\boldsymbol{W}) = \sum_{i=1}^n \alpha_i W_i + \beta$.
6. If $\operatorname{Cov}(U, \boldsymbol{W}) = \boldsymbol{0}$, then $P(U|\,\boldsymbol{W}) = \mathbb{E}[U]$.
7. If $\boldsymbol{V}$ is a random vector such that the components of $\mathbb{E}(\boldsymbol{V}\boldsymbol{V}')$ are all finite, then $P(U|\,\boldsymbol{W}) = P(P(U|\,\boldsymbol{W}, \boldsymbol{V})|\,\boldsymbol{W})$.

Both the linear and general prediction operators are used extensively in forecasting, such as the innovations algorithm and general forecasting of ARIMA processes.

2.4 The Innovations Algorithm

This section is based on Section 2.5.4 of [1].

The innovations algorithm is the de-facto main forecasting algorithm used to predict any time series, regardless of stationarity, and is used in the many popular time series packages, such as `stats`.

Suppose that $\{X_t\}$ is a zero-mean series with finite second moments for any t, and

$$\mathbb{E}[X_i X_j] = \kappa(i, j).$$

Furthermore, let

$$\hat{X}_n = \begin{cases} 0, & \text{if } n = 1, \\ P_{n-1}X_n, & \text{if } n \geq 2, \end{cases} \tag{2.12}$$

and

$$\nu_n = \mathbb{E}[(X_{n+1} - P_n X_{n+1})^2]$$

Lastly, we shall define an *innovation* to be a one-step predictor $U_n := X_n - \hat{X}_n$.

2.4.1 The General Innovations Algorithm

It can be shown that (2.12) can be rewritten as

$$\hat{X}_n = \begin{cases} 0, & \text{if } n = 0, \\ \sum_{j=1}^{n} \theta_{nj} U_{n+1-j}, & \text{if } n \geq 1, \end{cases} \tag{2.13}$$

where the coefficients $\theta_{n1}, \ldots, \theta_{nn}$ can be recursively computed from the equations

$$\nu_0 = \kappa(1,1), \tag{2.14}$$

$$\theta_{n,n-k} = \nu_k^{-1} \left(\kappa(n+1,k+1) - \sum_{j=0}^{k-1} \theta_{k,k-j}\, \theta_{n,n-j}\, \nu_j \right), \quad 0 \leq k < n, \tag{2.15}$$

and

$$\nu_n = \kappa(n+1,k+1) - \sum_{j=0}^{n-1} \theta_{n,n-j}^2\, \nu_j. \tag{2.16}$$

Proof. This proof closely follows Proposition 5.2.2 in [5].

Let $\mathcal{H}_n$ be the closed linear subspace $\operatorname{span}\{X_1, \ldots, X_n\}$, and let $\langle X, Y \rangle$ be the scalar product defined by $\mathbb{E}[X, Y]$.

The set $U := \{U_1, U_2, \ldots, U_n\}$ is orthogonal w.r.t. $\langle \cdot, \cdot \rangle$, since $U_i \in \mathcal{H}_{j-1}$. Furthermore, $U_j \perp \mathcal{H}_{j-1}$ by definition of $\hat{X}_j$.

Taking the scalar product of both sides of (2.12) with U_{k+1}, $0 \leq k < n$, we have

$$\langle \hat{X}_{n+1}, U_{k+1} \rangle = \theta_{n,n-k}\, \nu_k. \tag{2.17}$$

Since $U_{n+1} \perp U_{k+1}$ by orthogonality of U, we may apply Gram-Schmidt to calculate the coefficients $\theta_{n,n-k}\, \nu_k$ for $k = 0, \ldots, n-1$ as

$$\theta_{n,n-k} = \nu_k^{-1} \langle X_{n+1}, U_{k+1} \rangle. \tag{2.18}$$

Making use of the representation (2.12), and replacing n with k, we obtain

$$\theta_{n,n-k} = \nu_k^{-1} \left(\kappa(n+1,k+1) - \sum_{j=0}^{k-1} \theta_{k,k-j}\, \theta_{n,n-j}\, \nu_j \right), \tag{2.19}$$

as required. □

2.4.2 Application of the Innovations Algorithm to ARMA Processes

This subsection is based on Section 3.3 of [1].

For causal ARMA processes, we can simplify the Innovations algorithm to

$$\hat{X}_{n+1} = \begin{cases} \sum_{j=1}^{n} \theta_{nj} \left(X_{n+1-j} - \hat{X}_{n+1-j} \right), & 1 \le n < m, \\ \phi_1 X_n + \cdots + \phi_p X_{n+1-p} + \sum_{j=1}^{q} \theta_{nj} \left(X_{n+1-j} - \hat{X}_{n+1-j} \right), & n \ge m, \end{cases}$$

where θ_{nj} can be calculated using the autocovariance $\kappa(i,j)$ (which themselves can be found with (2.5)):

$$\kappa(i,j) = \begin{cases} \sigma^{-2}\gamma_X(i-j), & 1 \le i,j \le m, \\ \sigma^{-2}[\gamma_X(i-j) - \sum_{r=1}^{p} \phi_r \gamma_X(r - |i-j|)], & \min(i,j) \le m < \max(i,j) \le 2m, \\ \sum_{r=0}^{q} \theta_r \theta_{r+|i-j|}, & \min(i,j) > m, \\ 0, & \text{otherwise.} \end{cases}$$

The mean squared error can be expressed as

$$\mathbb{E}[(X_{n+1} - \hat{X}_{n+1})^2] = \sigma^2\, \mathbb{E}[(W_{n+1} - \hat{W}_{n+1})^2] =: \sigma^2 r_n. \tag{2.20}$$

Proof. This derivation follows the one in Section 3.3 of [1].

We first consider the transformed time series

$$\begin{cases} W_t = \sigma^{-1} X_t, & t = 1, \ldots, m, \\ W_t = \sigma^{-1}\phi(B) X_t, & t > m, \end{cases} \tag{2.21}$$

where

$$m = \max(p, q). \tag{2.22}$$

Without loss of generality, define $\theta_0 := 1$ and $\theta_j := 0$ for $j \le q$, $p \ge 1$ and $q \ge 1$.

Applying the Innovations Algorithm to $\{W_t\}$ and inferring the coefficients θ_{nj} the above expression for κ gives us

$$\begin{cases} \hat{W}_{n+1} = \sum_{j=1}^{n} \theta_{nj}(W_{n+1-j} - \hat{W}_{n+1-j}), & 1 \le n < m, \\ \hat{W}_{n+1} = \sum_{j=1}^{q} \theta_{nj}(W_{n+1-j} - \hat{W}_{n+1-j}), & n \ge m. \end{cases}$$

Equations (2.21) show us a linear relationship between X_t and its corresponding transformed observation W_t. Thus by linearity, the linear one-step predictor P_n is identical for both W_t and X_t. We obtain

$$\begin{cases} \hat{W}_t = \sigma^{-1}\hat{X}_t, & t = 1, \ldots, m, \\ \hat{W}_t = \sigma^{-1}[\hat{X}_t - \phi_{1t-1} - \cdots - \phi_p X_{t-p}], & t > m, \end{cases}$$

and using (2.21) once more, we obtain the required result. □

Corollary 2.11. *The h-step predictors satisfy*

$$P_n X_{n+h} = \sum_{i=1}^{p} \phi_i P_n X_{n+h-i} + \sum_{j=h}^{q} \theta_{n+h-1,j}(X_{n+h-j} - \hat{X}_{n+h-j}). \tag{2.23}$$

Proof. Cf. Section 3.3.1 in [1]. □

2.5 Fitting mixed ARMA models using Innovations

This section is based on Section 5.1, p.136–137 and Section 5.5.2 of [1].

Prior to forecasting ARMA processes based on our data, we must firstly choose a suitable ARMA process which best fits our data. To this end, we have to determine several parameters, including the choice of p and q (order selection), estimating the mean, the coefficients of $\phi(z)$ and $\theta(z)$, as well as the white noise variance σ^2. Such methods are implemented in `forecast`'s `auto.arima()` function, and this section will only cover such techniques for mixed ARMA models.

2.5.1 Estimation of ϕ and θ

Consider a causal model X_t, which can be expressed using (2.3), its coefficients $\psi_1, \ldots, \psi_{p+q}$ satisfying (2.4). To estimate these coefficients, we can refer to the innovation estimates $\hat{\theta}_{m1}, \ldots, \hat{\theta}_{m,p+q}$, and substituting these into (2.4), we obtain

$$\hat{\theta}_{mj} = \theta_j + \sum_{i=1}^{\min(j,p)} \phi_i \hat{\theta}_{m,j-i}, \quad j = 1, \ldots, p+q. \tag{2.24}$$

We can solve the equations (2.24) by rewriting them in matrix form to get our initial parameter estimates of $\hat{\boldsymbol{\phi}}$ and $\hat{\boldsymbol{\theta}}$:

$$\begin{pmatrix} \hat{\theta}_{m,q+1} \\ \hat{\theta}_{m,q+2} \\ \vdots \\ \hat{\theta}_{m,q+p} \end{pmatrix} = \begin{pmatrix} \hat{\theta}_{mq} & \hat{\theta}_{m,q-1} & \cdots & \hat{\theta}_{m,q+1-p} \\ \hat{\theta}_{m,q+1} & \hat{\theta}_{m,q} & \cdots & \hat{\theta}_{m,q+2-p} \\ \vdots & \vdots & & \vdots \\ \hat{\theta}_{m,q+p-1} & \hat{\theta}_{m,q+p-2} & \cdots & \hat{\theta}_{m,q} \end{pmatrix} \begin{pmatrix} \phi_1 \\ \phi_2 \\ \vdots \\ \phi_p \end{pmatrix}. \tag{2.25}$$

We can now estimate the first q θ_js using (2.24) and (2.25):

$$\hat{\theta}_j = \hat{\theta}_{mj} - \sum_{i=1}^{\min(j,p)} \hat{\phi}_i \, \hat{\theta}_{m,j-i}, \quad j = 1, \ldots, q.$$

The remaining parameter is the white noise variance σ^2, which is estimated by

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{t=1}^{n}\left(X_t - \hat{X}_t\right)^2 / r_{t-1},$$

where $\hat{X}_t$ is the one-step predictor calculated using $\hat{\boldsymbol{\phi}}$ and $\hat{\boldsymbol{\theta}}$ from above; r_n is defined as in (2.20).

2.5.2 Order Selection

A naive attempt to choose the orders p and q is to set both parameters to be as large as possible. However, this may result in overfitting, meaning that the model may have a low bias (in our case, a low white noise variance) but a higher (test) variance.

We shall turn our attention to some introductory terms:

Definition 2.12. The *Gaussian Likelihood for an ARMA Process* is defined as

$$L(\boldsymbol{\phi}, \boldsymbol{\theta}, \sigma^2) = \frac{1}{\sqrt{(2\pi\sigma^2)^n r_0 \cdots r_{n-1}}} \exp\left\{ -\frac{1}{2\sigma^2}\sum_{j=1}^{n}\frac{(X_j - \hat{X}_j)^2}{r_{j-1}} \right\}$$

(The r_ns as well as a motivation of this definition can be found by applying the definition of $\hat{X}_j$ found in the innovations algorithm (2.24))

Following the standard procedure, we differentiate $L(\boldsymbol{\phi}, \boldsymbol{\theta}, \sigma^2)$ partially w.r.t. σ^2 and obtain the *maximum likelihood estimators* $\hat{\boldsymbol{\phi}}, \hat{\boldsymbol{\theta}}$ and $\hat{\sigma}^2$:

$$S\left(\hat{\boldsymbol{\phi}}, \hat{\boldsymbol{\theta}}\right) = \sum_{j=1}^{n}\left(X_j - \hat{X}_j\right)^2 / r_{j-1},$$
$$\hat{\sigma}^2 = \frac{1}{n} S\left(\hat{\boldsymbol{\phi}}, \hat{\boldsymbol{\theta}}\right),$$

and $\hat{\boldsymbol{\phi}}, \hat{\boldsymbol{\theta}}$ are the values of $\boldsymbol{\phi}, \boldsymbol{\theta}$ which minimise

$$\ell(\boldsymbol{\phi}, \boldsymbol{\theta}) = \ln(n^{-1} S\left(\boldsymbol{\phi}, \boldsymbol{\theta}\right)) + n^{-1}\sum_{j=1}^{n} \ln r_{j-1},$$

Using the above terminology, we are able to consider the AICC (Akaike Information [Corrected] Criterion), which is an unbiased estimate of the Kullback-Leibler index :

Definition 2.13. The AICC is defined as

$$\text{AICC} := -2\ln L(\boldsymbol{\phi}_p, \boldsymbol{\theta}_p, S(\boldsymbol{\phi}_p, \boldsymbol{\theta}_p)/n) + 2(p+q+1)n/n(n-p-q-2). \tag{2.26}$$

Minimising this statistic provides us with estimates for p and q.

Chapter 3

ARIMA Processes

Most time series are not necessarily stationary, hence they cannot be modelled by an ARMA process. However, many can be transformed to an *autoregressive integrated moving-average*, or ARIMA model, which are a generalisation of ARMA processes. The key difference between ARIMA and ARMA is that ARIMA models can also model non-stationary processes; in fact, an ARIMA process reduces to an ARMA process when differenced finitely many times, enabling us to utilise the forecasting methods introduced in the previous chapter.

3.1 General ARIMA Processes

This section is based on Sections 6.1 and 6.4 of [1].

Definition 3.1 (ARIMA(p, d, q) Process)**.** A time series $\{X_t\}$ is an *ARIMA(p, d, q) process* if $Y_t := (1 - B)^d X_t$ is a causal ARMA(p, q) process.

In particular, $\{X_t\}$ satisfies the difference equation of the form

$$\phi^*(B)X_t \equiv \phi(B)(1 - B)^d X_t = \theta(B)Z_t, \quad \{Z_t\} \sim \mathrm{WN}(0, \sigma^2), \tag{3.1}$$

where $\phi(z) \neq 0$ for $|z| \leq 1$, and the polynomial $\phi^*(z)$ has a zero of order d at $z = 1$.

ARIMA processes have the special property that one can add an arbitrary polynomial trend of degree $(d-1)$ to $\{X_t\}$ without altering the difference equation (3.1), making them especially suited to model processes with trends.

3.1.1 Forecasting ARIMA processes

Let us consider an ARIMA process with $d \geq 1$, since for $d = 0$, we can refer to (2.2). The first and second moments $\mathbb{E}[X_t]$ and $\mathbb{E}[X_{t+h}X_t]$ cannot be determined by (3.1), therefore, we require further assumptions to calculate the predictor function. Following an analogous derivation to the linear predictor for ARMA models, we obtain the following expressions for the best ARIMA linear predictor:

Theorem 3.2. *Let $\phi^*(z) = (1-z)^d\phi(z) = 1 - \phi_1^* z - \cdots - \phi_{p+d}^* z^{p+d}$ be the ARIMA differencing polynomial. Then for $1 \le h \le q$, the best linear predictor for ARIMA processes can be written as*

$$P_n X_{n+h} = \sum_{j=1}^{p+d} \phi_j^* P_n Y_{n+h-j} + \sum_{j=h}^{q} \theta_{n+h-1,j} \, (X_{n+h-j} - \hat{X}_{n+h-j}). \tag{3.2}$$

Its mean squared error can be expressed as

$$\sigma_n^2(h) = \mathbb{E}[(X_{n+h} - P_n X_{n+h})^2] = \sum_{j=0}^{h-1} \left(\sum_{r=0}^{j} \chi_r \theta_{n+h-r-1,j-r} \right)^2 \nu_{n+h-j-1},$$

where

$$\begin{aligned} \theta_{n0} &= 1 \\ \chi(z) &= \sum_{r=0}^{\infty} \chi_r z^r = (1 - \phi_1^*(z) - \cdots - \phi_{p+d}^*(z^{p+d}))^{-1} \\ \nu_{n+h-j-1} &= \mathbb{E}[(X_{n+h-j} - \hat{X}_{n+h-j})^2] = \mathbb{E}[(Y_{n+h-j} - \hat{Y}_{n+h-j})^2]. \end{aligned}$$

By multiplying out terms and comparing coefficients, χ_j can be found from the recursions

$$\chi_j = \sum_{k=1}^{\min(p,j)} \phi_k^* \chi_{j-k}, \quad j = 1, 2, \ldots$$

Furthermore, for $h > q$, the general solution, assuming that the AR polynomial Φ admits distinct roots $(\xi_1, \ldots, \xi_p)$, is given by

$$P_n X_{n+h} = a_0 + a_1 h + \cdots + a_{d-1} h^{d-1} + b_1 \xi_1^{-h} + \cdots + b_p \xi_p^{-h} \quad h > q - p - d. \tag{3.3}$$

For the proof, we must consider the following lemmas:

Lemma 3.3. *Let $U, W_n, \ldots, W_1$ and $\tilde{W}_n, \ldots, \tilde{W}_1$ be random variables defined on the same probability space such that all of their second moments are finite for all values of j. If*

$$\begin{aligned} (W_n, \ldots, W_1)^T &= \boldsymbol{A}(\tilde{W}_n, \ldots, \tilde{W}_1)^T \\ (\tilde{W}_n, \ldots, \tilde{W}_1)^T &= \boldsymbol{B}(W_n, \ldots, W_1)^T, \quad \boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{n \times n}, \end{aligned}$$

then

$$P(U | W_n, \ldots, W_1) = P(U | \tilde{W}_n, \ldots, \tilde{W}_1) \quad \text{almost everywhere.}$$

Lemma 3.4. *Let $U, W_n, \dots, W_1$ be as above. Suppose there exists an $i_0 \in \{1, \dots, n-1\}$ such that* $\operatorname{Cov}(U, W_{n-j+1}) = 0$ *for all $j \in i_0 + 1, \dots, n$. Then*

$$P(U|W_n, \dots, W_1) = P(U|W_n, \dots, W_{n-i_0+1}) \quad \textit{almost everywhere.}$$

Proof. (of Theorem 3.2): Assume $\{X_t\}$ satisfies

$$(1-B)^d X_t = Y_t, \quad t = 1, 2, \dots .$$

Then by definition, $\{Y_t\}$ is a causal ARMA(p, q) process and $(X_{1-d}, \dots, X_0)$ is uncorrelated with Y_t for positive t. Using the binomial formula, we can rearrange for X_t to obtain

$$X_t = Y_t - \sum_{j=1}^{d} \binom{d}{j} (-1)^j X_{t-j}, \quad t > 0. \tag{3.4}$$

For convenience, we now shift the time scale so that we consider the observations $X_{1-d}, \dots, X_n$, in order to ultimately predict $X_n + h$ based on the past realisations.

By definition of Y_t, we observe that Y_j is a linear combination of the past realisations $X_{1-d}, \dots, X_n$. By Lemma 3.3, we may apply the best linear predictor operator P_n to both sides of (3.4) to get

$$P_n X_{n+h} = P_n Y_{n+h} - \sum_{j=1}^{d} \binom{d}{j} (-1)^j X_{n+h-j}. \tag{3.5}$$

Additionally, we may identify $P_n X_{n+1-j} = X_{n+1-j}$, since X_{n+1-j} is itself a linear combination of $X_n, \dots, X_{1-d}$. Now let us denote the one-step predictors $P_n X_{n+1} := \hat{X}_{n+1}$ and $P_n Y_{n+1} := \hat{Y}_{n+1}$. Furthermore, we can use (3.4) and (3.5) with $h = 1$ to obtain

$$X_{n+1} - \hat{X}_{n+1} = Y_{n+1} - \hat{Y}_{n+1}, \quad n \geq 1.$$

Because $\{Y_t\}$ is an ARMA process, we can use (2.23) and the above identity, resulting in

$$P_n X_{n+h} = \sum_{i=1}^{p} \phi_i P_n Y_{n+h-i} + \sum_{j=h}^{q} \theta_{n+h-1,j} (X_{n+h-j} - \hat{X}_{n+h-j}).$$

Setting $\phi^*(z)$ as in the theorem, we obtain (3.2) as required.

For $h > q$, $\sum_{j=h}^{q} \theta_{n+h-1,j} (X_{n+h-j} - \hat{X}_{n+h-j}) = 0$, and by defining the *forecast function* $g(h) := P_n X_{n+h}$, (3.2) simplifies down to

$$g(h) = \sum_{j=1}^{p+d} \phi_j^* \, g(h-j) \quad \forall h > q,$$

and the general solution is given by (3.3) for $h > q-p-d$. The range of h can be explained by observing the smallest argument of $g(h)$ is $h - p - d$. □

3.2 Seasonal ARIMA Processes

This section is based on Section 6.5 of [1].

If the data shows signs of seasonality, we have seen from the introduction that it is useful to eliminate the seasonal component of period s. A special type of ARIMA process designed to model such processes is the SARIMA process:

Definition 3.5. If d and D are non-negative integers, then $\{X_t\}$ is called a *seasonal ARIMA$(p,d,q) \times (P,D,Q)_s$ process with period s* if the *differenced* series $Y_t = (1-B)^d(1-B^s)^D X_t$ is a causal ARMA process defined by

$$\phi(B)\Phi(B^s)Y_t = \theta(B)\Theta(B^s)Z_t, \quad \{Z_t\} \sim \mathrm{WN}(0,\sigma^2), \tag{3.6}$$

where $\phi(z) = 1-\phi_1 z - \cdots - \phi_p z^p$, $\Phi(z) = 1-\Phi_1 z - \cdots - \Phi_P z^P$, $\theta(z) = 1+\theta_1 z + \cdots + \theta_q z^q$ and $\Theta(z) = 1+\Theta_1 z + \cdots + \Theta_Q z^Q$.

Remark 3.6. Analogous to ARMA processes, $\{Y_t\}$ is causal if and only if $\phi(z)$ and $\Phi(z)$ do not possess any roots inside the unit circle.

An advantage of using SARIMA models over the classical decomposition model $X_t = m_t + s_t + Y_t$ (cf. Equation (1.5)) is that, unlike the classical model which requires the seasonality component s_t to be fixed, SARIMA models allow more flexibility and randomness per cycle, enabling us to simulate a larger class of data. In the practical sense, the parameter D is rarely more than 1; P and Q are less than 3.

3.3 Forecasting SARIMA Processes

Forecasting SARIMA processes is similar to its non-seasonal counterpart, except we now obtain the following recursions:

Theorem 3.7. *Let $\{X_t\}$ be a time series fulfilling the equation*

$$(1-B)^d(1-B^s)^D X_t = Y_t$$

with d, D and s as given in (3.6) and $\{Y_t\}$ an ARMA process. Then the best h-step linear prediction is given by the recursion

$$P_n X_{n+h} = P_n Y_{n+h} + \sum_{j=1}^{d+Ds} a_j P_n X_{n+h-j}$$

with mean squared error

$$\mathbb{E}[(X_{n+h} - P_n X_{n+h})^2] = \sum_{j=0}^{h-1} \left(\sum_{r=0}^{j} \chi_r \theta_{n+h-r-1,j-r} \right)^2 \nu_{n+h-j-1},$$

where θ_{nj} and ν_n are calculated using the Innovations Algorithm on $\{Y_t\}$ and

$$\chi(z) = \sum_{r=0}^{\infty} \chi_r z^r = \frac{1}{\phi(z)\Phi(z^s)(1-z)^d(1-z^s)^D}, \quad |z| < 1.$$

Having covered all the terminology needed to understand SARIMA models and their forcasting methods, we shall turn to another approach of modelling using exponential smoothing.

Chapter 4

Holt–Winters

The *Holt–Winters Exponential Smoothing* (HW) or *Triple Exponential Smoothing* method, named after Charles Holt and Peter Winters in 1960, is used for forecasting a time series that shows both a trend and seasonal component. It is based on the three basic notions of smoothing with a finite moving average filter, exponential smoothing, and Holt exponential smoothing. This chapter is based on Sections 1.5 and 10.2 of [1], as well as [6].

Let us first consider the nonseasonal model of the time series $\{X_t\}$ with trend:

$$X_t = m_t + Y_t, \quad t = 1, \dots, n, \quad \mathbb{E}[Y_t] = 0. \tag{4.1}$$

Let q be a non-negative integer and consider the *two-sided moving average*

$$W_t = \frac{1}{2q+1}\sum_{j=-q}^{q} X_{t-j} = \frac{1}{2q+1}\sum_{j=-q}^{q} m_{t-j} + \frac{1}{2q+1}\sum_{j=-q}^{q} Y_{t-j} \approx m_t,$$

under the assumptions that $q+1 \leq t \leq n-q$, the trend m_t is approximately linear over the interval $[t-q, t+q]$ and the average of the error terms are approximately zero over this interval. Then the simple moving average gives us the estimates

$$\hat{m_t} = \frac{1}{2q+1}\sum_{j=-q}^{q} X_{t-j}, \quad q+1 \leq t \leq n-q.$$

For the cases $t \leq 0$ or $t > n$, packages such as `stats` define $X_t := X_1$ and $X_t := X_n$ respectively.

Estimation using averaging smooths out the variation of our time series by “smoothing” out the variation between points. This method is unreliable due to two reasons: firstly, we consider a two-sided average, meaning that, depending on the value of t, we may encounter events in the time series which are in the future; in addition, every historical value is given the same equal emphasis for forecasting future values of the time series which may lead to inaccuracies, leading us to use exponential smoothing.

4.1 Exponential smoothing

Exponential smoothing relies on the assumption that more recent values are more useful to our forecast than the ones further in the past, i.e. the weights of each element of the time series decrease exponentially with respect to time.

Consider again the non-seasonal model denoted by (4.1). Then for any fixed $\alpha \in [0,1]$, the one-sided moving averages $\hat{m}_t$, $t = 1, \ldots, n$ are defined by the recursions

$$\hat{m}_t = \alpha X_t + (1-\alpha)\,\hat{m}_{t-1}, \quad t = 2, \ldots, n, \tag{4.2}$$

$$\hat{m}_1 = X_1, \tag{4.3}$$

for some constant $\alpha \in [0,1]$. Substituting (4.3) into (4.2), we obtain

$$\hat{m}_t = \sum_{j=0}^{t-2} \alpha(1-\alpha)^j X_{t-j} + (1-\alpha)^{t-1} X_1,$$

showing the exponentially decreasing weights for X_t, especially for α close to 0, as desired.

As with the naive moving average method, the basic exponential smoothing method is unable to model seasonal models.

4.2 Holt's Algorithm

Let $Y_1, Y_2, \ldots, Y_n$ be observations from the non-seasonal model (4.1). If the series is stationary, then the slope m_t is constant, and we can express the exponential smoothing h-step forecast as

$$P_n Y_{n+h} = \hat{m}_n. \tag{4.4}$$

However, for data which is not necessarily stationary, i.e. its trend is not constant, we can generalise the above expression and obtain

$$P_n Y_{n+h} = \hat{a}_n + \hat{b}_n h, \quad h = 1, 2, \ldots, \tag{4.5}$$

where $\hat{a}_n$ and $\hat{b}_n$ are denoted as the estimates of the "level" and "slope" of the trend $\hat{m}_n$, respectively.

Let us denote the one-step forecast $P_n Y_{n+1}$ as Y_{n+1}. From (4.5) and setting $h = 1$, we obtain

$$\hat{Y}_{n+1} = \hat{a}_n + \hat{b}_n. \tag{4.6}$$

Holt's algorithm estimates $\hat{a}_n$ and $\hat{b}_n$ in a similar recursive fashion to the exponential smoothing method as

$$\hat{a}_{n+1} = \alpha Y_{n+1} + (1-\alpha)(\hat{a}_n + \hat{b}_n), \tag{4.7}$$
$$\hat{b}_{n+1} = \beta(\hat{a}_{n+1} - \hat{a}_n) + (1-\beta)\hat{b}_n, \tag{4.8}$$

with initial conditions

$$\hat{a}_2 = Y_2, \quad \hat{b}_2 = Y_2 - Y_1.$$

Equations (4.7) and (4.8) make use of the exponential smoothing parameters α and β similar to (4.2). These are best chosen between 0 and 1 to minimise the MSE $\sum_{i=3}^{n}(Y_i - P_{i-1}Y_i)^2$.

Now we would like to show that Holt's algorithm is analogous to ARIMA. Using (4.2) and (4.4), the one-step forecasts obtained by exponential smoothing with parameter α can be re-written as:

$$P_nY_{n+1} = Y_n - (1-\alpha)(Y_n - P_{n-1}Y_n), \quad n \geq 2 \tag{4.9}$$

Notice that these are the same relations satisfied by the large-sample minimum MSE forecasts of the invertible ARIMA$(0, d, q)$ process

$$Y_t = Y_{t-1} + Z_t - (1-\alpha)Z_{t-1}, \quad \{Z_t\} \sim \mathrm{WN}(0, \sigma^2), \tag{4.10}$$

with optimised α found by minimising the average squared error of the one-step average squared error.

In the general case, it can be shown that Holt's Algorithm can be viewed as fitting a family of ARIMA(p, d, q) processes

$$(1-B)^2Y_t = Z_t - (2-\alpha-\alpha\beta)Z_{t-1} + (1-\alpha)Z_{t-2}, \quad \{Z_t\} \sim \mathrm{WN}(0, \sigma^2). \tag{4.11}$$

The coefficients α and β are selected as described above, and the estimate of σ^2 is the average squared error of the one-step forecasts of $Y_3, \ldots, Y_n$ obtained using the methods described in Chapter 2.5.

4.3 Holt–Winters' Algorithm

The methods described in the previous section were based on the non-seasonal model assumption (4.2). However, data such as that used in Chapter 5 is expected to possess seasonality, hence we need to expand on Holt's Algorithm. In essence, we now consider three parameters α, β and γ, where γ now models the seasonal component at the current point in time. Generalising the h-step forecast again, we obtain

$$P_nY_{n+h} = \hat{a}_n + \hat{b}_nh + \hat{c}_{n+h}, \quad h = 1, 2, \ldots, \tag{4.12}$$

where $\hat{c}_{n+h}$ represents the seasonal component. Letting k be the smallest integer such that $n+h-kd \leq n$, then we set the $\hat{c}_i$s to have a seasonal structure:

$$\hat{c}_{n+h} = \hat{c}_{n+h-kd}, \quad h = 1, 2, \ldots,$$

and similar to (4.7) and (4.8), we obtain the three following recursions for $\hat{a}_i$, $\hat{b}_i$ and $\hat{c}_i$ (leading to the procedure's alternative name of triple exponential smoothing):

$$\begin{aligned}
\hat{a}_{n+1} &= \alpha\,(Y_{n+1} - \hat{c}_{n+1-d}) + (1-\alpha)(\hat{a}_n + \hat{b}_n),\\
\hat{b}_{n+1} &= \beta\,(\hat{a}_{n+1} - \hat{a}_n) + (1-\beta)\,\hat{b}_n,\\
\hat{c}_{n+1} &= \gamma\,(Y_{n+1} - \hat{a}_{n+1}) + (1-\gamma)\,\hat{c}_{n+1-d},
\end{aligned}$$

with initial conditions

$$\begin{aligned}
\hat{a}_{d+1} &= Y_{d+1},\\
\hat{b}_{d+1} &= (Y_{d+1} - Y_1)/d,\\
\hat{c}_i &= Y_i - (Y_1 + \hat{b}_{d+1}(i-1)), \quad i = 1, \ldots, d+1.
\end{aligned}$$

As with Holt' algorithm, we can specify the smoothing parameters α, β and γ to have values between 0 and 1, optimised to minimise the MSE $\sum_{i=d+2}^{n}(Y_i - P_{i-1}Y_i)^2$. Furthermore, we can again find an analogy between Holt–Winters and ARIMA. It can be proven that the recursions (4.12) are equivalent to the ARIMA forecasts

$$\begin{aligned}
(1-B)(1-B^d)Y_t = Z_t &+ \cdots + Z_{t-d+1}\\
&+ \gamma(1-\alpha)(Z_{t-d} - Z_{t-d-1}) - (2-\alpha-\alpha\beta)(Z_{t-1} + \cdots + Z_{t-d})\\
&+ (1-\alpha)(Z_{t-2} + \cdots + Z_{t-d-1}).
\end{aligned}$$

Naturally, the question arises as to what the difference between seasonal ARIMA and Holt–Winters is and which one performs better. We will discuss the advantages and disadvantages of both methods in the following chapter, as well as compare their effectiveness in forecasting on the Swiss deaths dataset.

Chapter 5

Experiments

Having investigated the theory behind the different approaches to time series forecasting, we shall now perform some numerical experiments. All methods were implemented in `R` and applied to the Swiss Weekly deaths (65+) datasets [7] [8].

The `R` packages `stats` and `forecast` contain the methods `ARIMA()`, `auto.arima()` and `HoltWinters()` will be used to perform the required analysis. Furthermore, the `forecast` package contains useful methods such as `ggtsdisplay()`, `checkresiduals()` and `autoplot()`, aiding us with the visualisation of our analysis. The code for each method can be found in Chapter A.

Firstly, we will train both models on the deaths dataset from 2010 to 2018, and will attempt to forecast 52 weeks ahead, verifying our forecasting using the testing set data for 2019.

Observing the figure below, we already notice a clear pattern of seasonality in our data towards the end and beginning of each year. This corresponds to the hypothesis that many elderly are more prone to illnesses in winter, such as the flu.

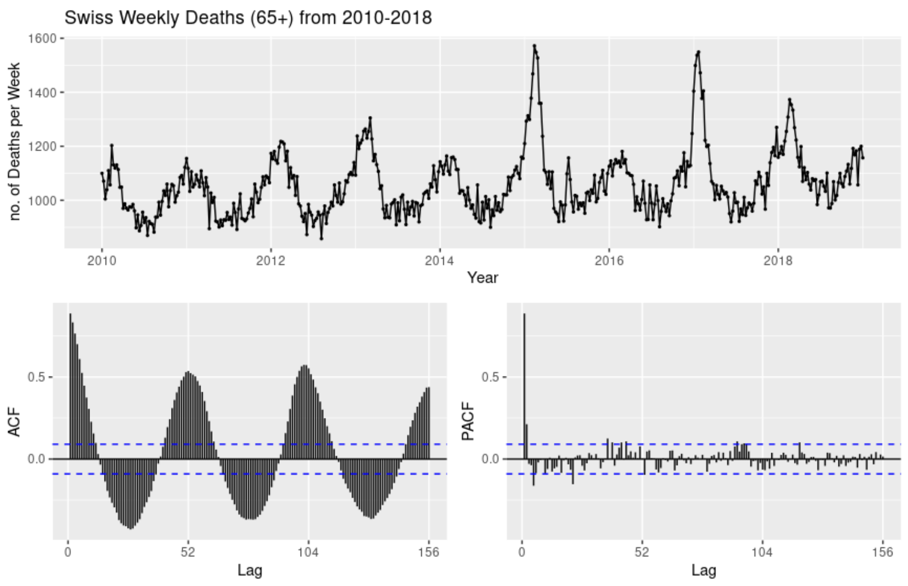

Figure 5.1: Swiss Deaths 65+ Time Series, ACF and PACF from 2010-2018 [7]

5.1 SARIMA

We follow a similar procedure to Example 6.5.4 in [1], Section 2.6 in [6] and Section 8.9 of [9].

Reiterating the theory from the previous chapters, the methodology of manually forecasting SARIMA processes is as follows:

1. Transform the dataset until we obtain a stationary time series. This is done using the methods described in Section 1.3.
2. Plot the modified, now stationary time series' PACF and ACF and note any spikes and their lags. This provides us with our estimates of the ARMA coefficients p and q. (cf. Section 2.2)
3. Fit an ARMA model using the innovations algorithm (cf. Chapter 2.5) and calculate the residuals and note the AICC and RMSE (root MSE). Test the noise using the methods from Section 1.4.
4. Inspect the ACF again and adjust the parameters p and q and adjust accordingly.

By analysing figure 5.1, we see that the ACF of our dataset is sinusoidal in form and outside of the confidence region, and there is a large spike in the PACF near 0 which then tails off to 0, indicating that the series can be made stationary after differencing. Furthermore, we would like to have a zero-mean time series, so we shall difference the data once to obtain the following graphic:

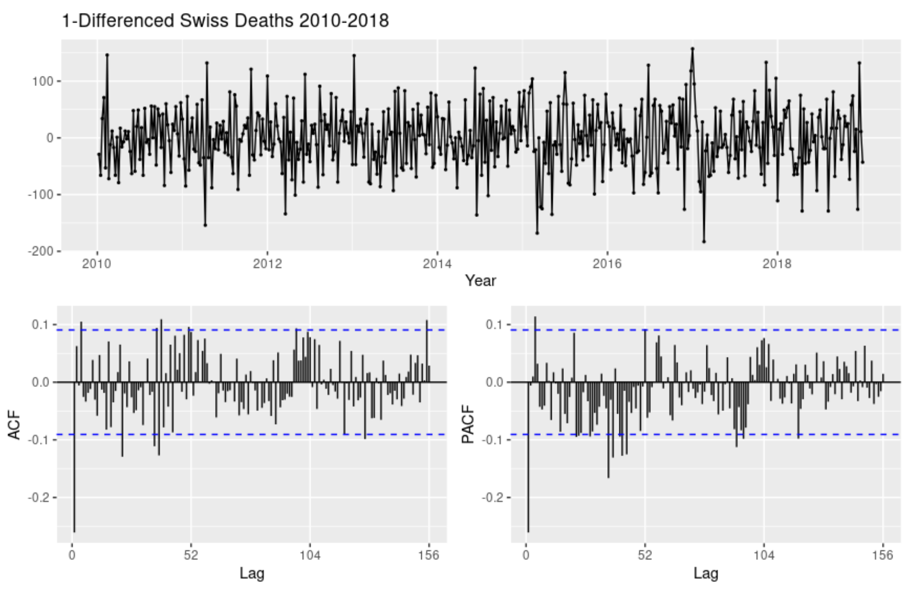

Figure 5.2: 1-Differenced Swiss Deaths 2010-2018

The differenced time series appears to be more stationary and well behaved around a common mean 0, its ACF and PACF also much better behaved, with both functions tailing off to 0 for larger lags, and the ACF only showing a significant spike at lag $= 50 \approx 1$ year. This implies that a good estimate for an ARIMA model is $\text{SARIMA}(0,1,1) \times (0,1,1)$.

After making this initial guess, we attempt to fit the $\text{SARIMA}(0,1,1) \times (0,1,1)$ model to our data, and obtain the following results:

```
Series: deaths_10_18
ARIMA(0,1,1)(0,1,1)[52]
Coefficients:
          ma1     sma1
      -0.3984   -1.000
s.e.   0.0468    0.375
sigma^2 estimated as 2436:  log likelihood=-2268.47
AIC=4542.94   AICc=4543   BIC=4555.03
```

i.e. our time series is estimated to have the form $Y_t = (1 - 0.3984B)(1 - B^{52})Z_t$, $Z_t \sim \text{WN}(0, 2436)$.

Plotting the residuals for the inital guess, we see that the noise is a white noise around 0 with standard deviation $\sqrt{2436} \approx 49$. Furthermore, the ACF is within the appropriate confidence levels save a few small spikes. However, the large spike in the histogram at level 0 suggests that the distribution may not be normal. To check this, we perform a Shapiro-Wilk test at a significance level of 5% to check for normality [10]. This produces a p-value of $6.623 \times 10^{-5} \ll 0.05$, hence we can only deduce that the residuals follow a white noise distribution.

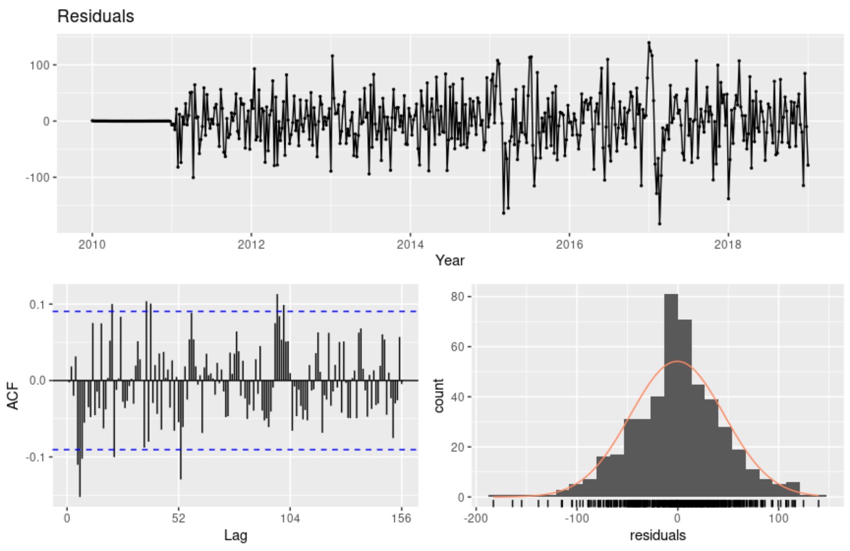

Figure 5.3: Initial Seasonal ARIMA Model Residuals, ACF, and Initial Seasonal ARIMA Residual Histogram, showing a white noise distribution with mean 0 and stdev. ≈ 49

Using the `forecast` function, we obtain the forecast below.

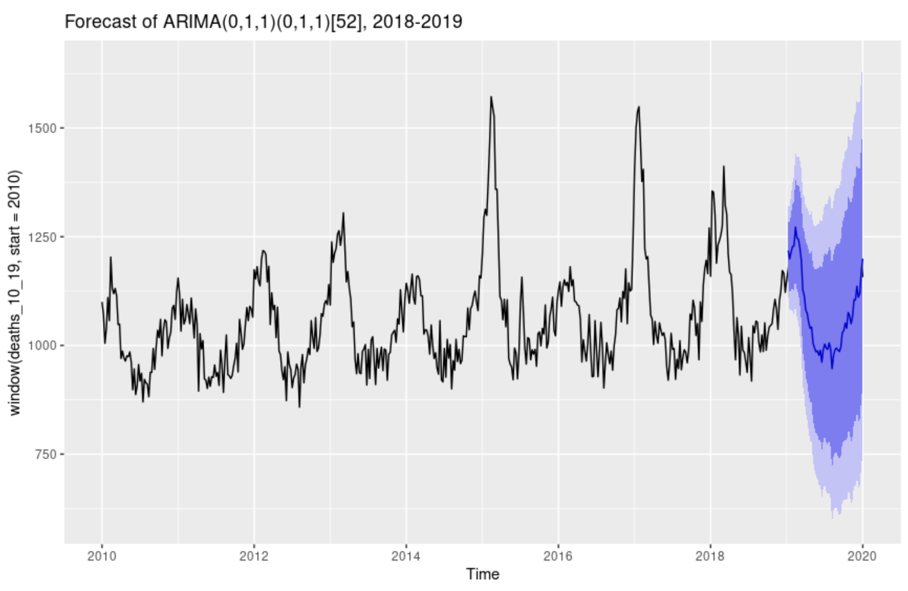

Figure 5.4: Initial Seasonal ARIMA Forecasts. The dark blue confidence region represents the 80% percentile, light blue represents the 95% percentile.

Comparing with the actual testing data, we obtain the following errors:

```
                    ME     RMSE      MAE        MPE     MAPE      MASE         ACF1 Theil's U
Training set -0.5195797 46.37004 33.97801 -0.1602148 3.174346 0.4683137 -0.002437964        NA
Test set     46.4951614 63.64347 54.28420  4.0733216 4.763479 0.7481908  0.256039786  1.127634
```

We are only interested in the RMSE (root mean squared error), and we will compare each method's error in the comparison section.

The `forecast` package also contains the `auto.arima()` function, providing us with automatic estimates of the orders of our seasonal model as well as estimates for each of the autoregressive and moving average coefficients. Running `auto.arima()` gives us the following output:

```
ARIMA(1,0,0)(1,1,0)[52] with drift
Coefficients:
         ar1      sar1   drift
      0.6625   -0.5517  0.2782
s.e.  0.0380    0.0418  0.1099
sigma^2 estimated as 3464:  log likelihood=-2299.22
AIC=4606.43   AICc=4606.53   BIC=4622.56
```

i.e.

$$(1-0.6625B)(1+0.5517B^{52})(Y_t-a-bt)=Z_t, \quad Z_t\sim \mathrm{WN}(0,3464), \quad a,b\in\mathbb{R}$$

where drift represents a fitted linear regression with ARIMA errors, i.e. we replace Y_t with $(Y_t - a - bt)$. (The theory behind ARIMA drifts is outside the scope of this thesis, however additional information can be found in [11] and [12]). Surprisingly, the automatic ARIMA model produces larger values of the AIC and AICc compared to the initial guess, suggesting that `auto.arima` did not search all possible models. Adding the parameters `stepwise=FALSE` and `approximation=FALSE` forces the algorithm to check more possibilities, however, the code did not stop executing upon multiple tries.

As with the initial guess, we plot the model's residuals and note that the ACF and PACF both show lags at weeks 52 and 104, i.e. 1 and 2 years, showing that the automatically chosen model can be improved on. Performing a Shapiro–Wilk test at a 5% significance level produces a p-value of $0.004941 \ll 0.05$, so we again deduce that the residuals maintain a zero-mean white noise distribution with standard deviation $\sigma = \sqrt{3464} \approx 59$.

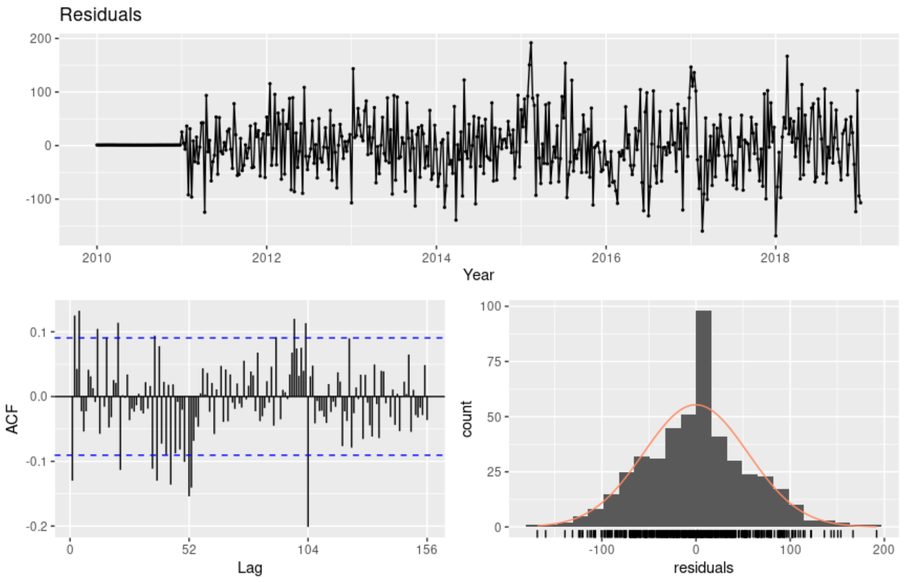

Figure 5.5: Automatic Seasonal ARIMA Residuals, ACF, and automatic Seasonal ARIMA residual Histogram, showing a white noise distribution with mean 0 and stdev. ≈ 59

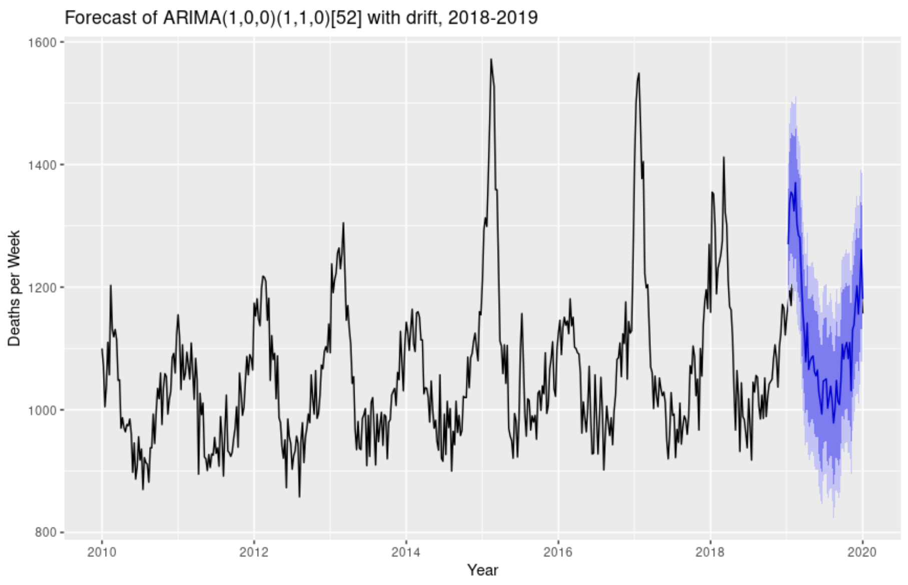

Figure 5.6: Automatic Seasonal ARIMA Forecast. The dark blue confidence region represents the 80% percentile, light blue represents the 95% percentile.

The forecast above produces the following errors:

```
                     ME     RMSE      MAE        MPE     MAPE      MASE       ACF1 Theil's U
Training set -0.2426614 55.29622 41.37073 -0.1960501 3.851327 0.5702064 -0.1299272        NA
Test set     -3.4913204 55.12722 41.08346 -0.3270072 3.560461 0.5662470  0.6487384 0.9172278
```

Notice that the RMSE is on average smaller in the initial guess, however the RMSE in the automatic model stays more consistent between the training set and test set, showing less overfitting to our data.

5.2 Holt–Winters

As with seasonal ARIMA, we shall follow a similar route to [1] as well as the article [6].

We first begin by fitting a Holt–Winters model to the deaths dataset from 2010-2018, noting that we leave the rest of the inputs of `HoltWinters()` blank to indicate that we want to model a possible trend and seasonality. Running the model provides us with the following Holt–Winters estimates for α, β and γ:

```
Call:
HoltWinters(x = deaths_10_18)
Smoothing parameters:
 alpha: 0.6040622
 beta : 0
 gamma: 0.7080221
```

with $\beta = 0$, since our data shows no visible trend.

Plotting the Holt–Winters estimates for the training data, we observe a visible smoothing of the data, as its alternative name *Triple Exponential Smoothing* suggests.

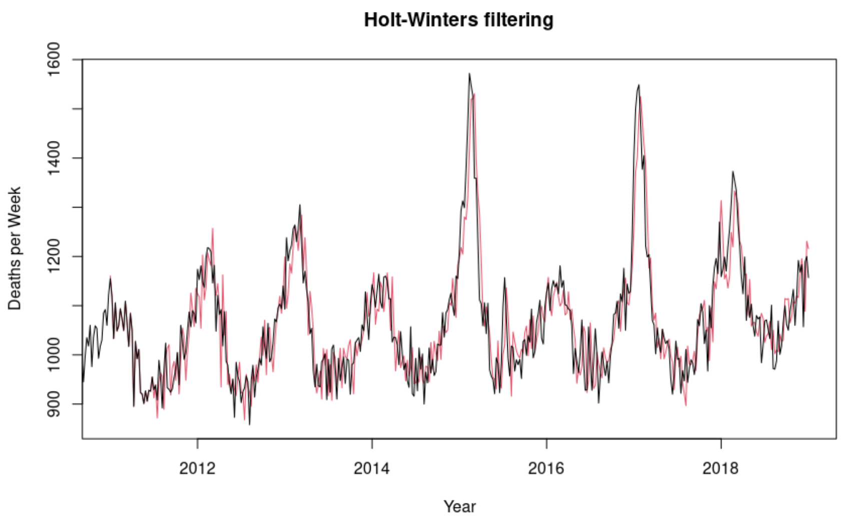

Figure 5.7: Smoothed Prediction on Training Data from 2010-2018

Following the same procedure to the SARIMA models, we plot the the residuals, their histogram and the ACF and note that there are no major spikes in the ACF, unlike the SARIMA family, suggesting a more stable model. The histogram peak at 0 is less significant than the SARIMA models, and performing a Shapiro–Wilk test at a 95% significance level produces the p-value 0.05066, showing that the Holt–Winters residuals follow a Gaussian distribution.

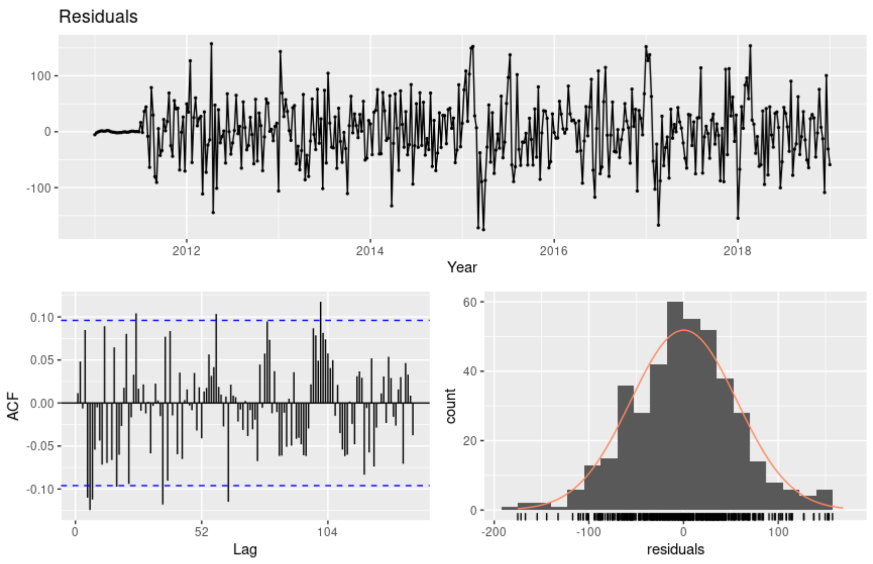

Figure 5.8: Holt–Winters Residuals, ACF, and residual histogram, showing a normal distribution.

Plotting the forecast, we notice a more modest peak compared to the ones forecasted by the SARIMA models. We also observe that the confidence regions are tighter than the previous two models.

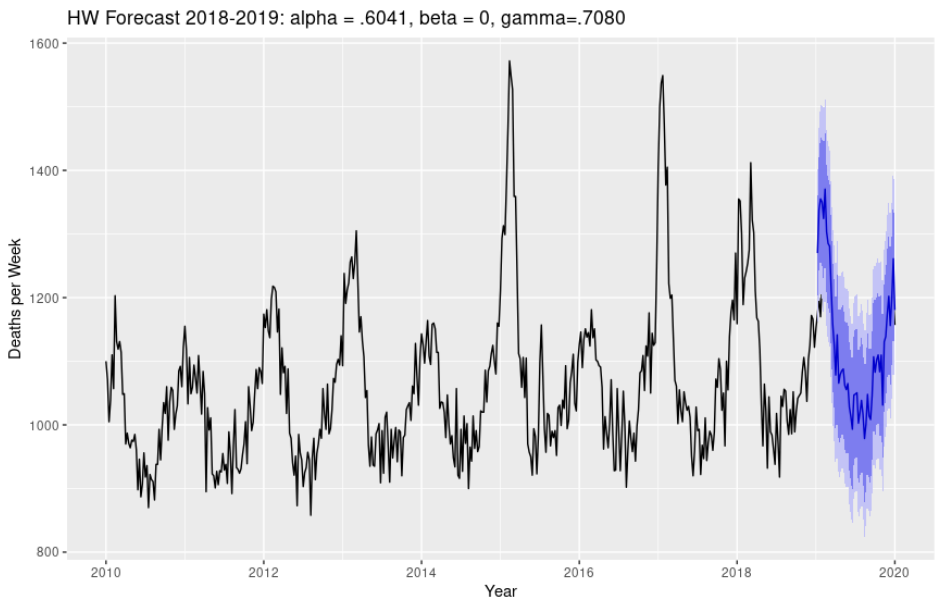

Figure 5.9: Holt–Winters Forecast. The dark blue confidence region represents the 80% percentile, light blue represents the 95% percentile.

The RMSE for the Holt–Winters forecast are as follows:

```
                      ME      RMSE      MAE        MPE     MAPE     MASE       ACF1 Theil's U
Training set  0.2736994  56.09318 42.99468 -0.1494921 4.000682 0.592589 0.01142327        NA
Test set     86.3129755 109.45382 90.06374  7.5213616 7.847319 1.241335 0.76019780  1.858238
```

showing that the model here performs the worst in both the training and testing sets.

5.3 Comparison

We now compare the three models seen in the previous sections and analyse their differences.

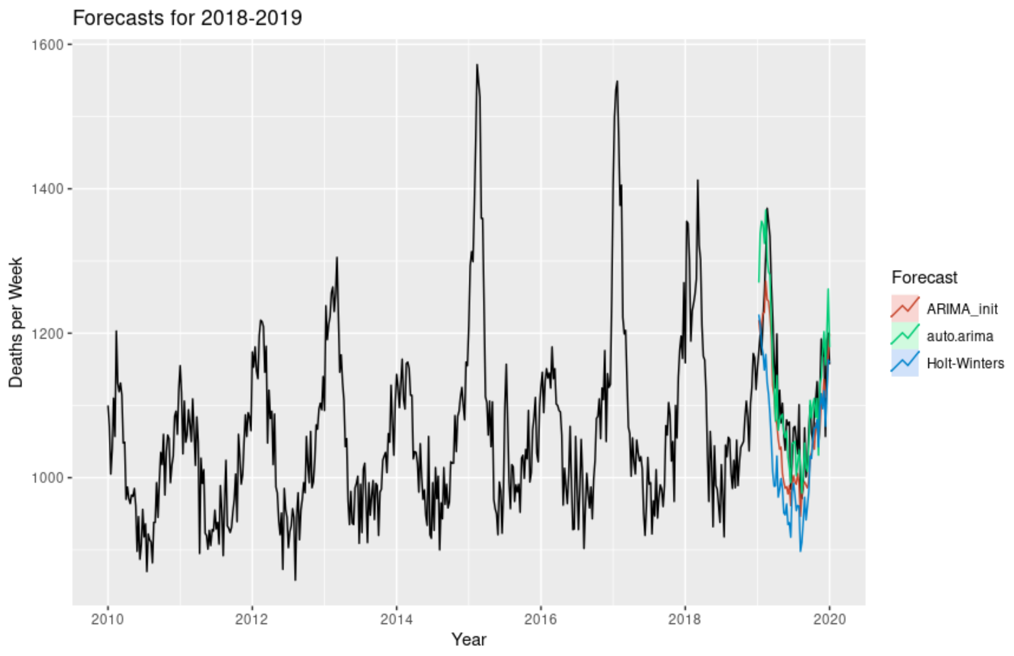

Figure 5.10: Comparison of all three methods, forecast from 2018-2019

```
                       Training set RMSE Test set RMSE
Initial Arima Forecast          46.37004      63.64347
Auto Arima Forecast             55.29622      55.12722
Holt-Winters Forecast           56.09318     109.45382
```

The automatically determined seasonal ARIMA model performs the best, with the Holt–Winters forecast performing the poorest.

We can also use the data collected from 2010 to 2019 to produce a second forecast for 2020, following an analogous procedure to above:

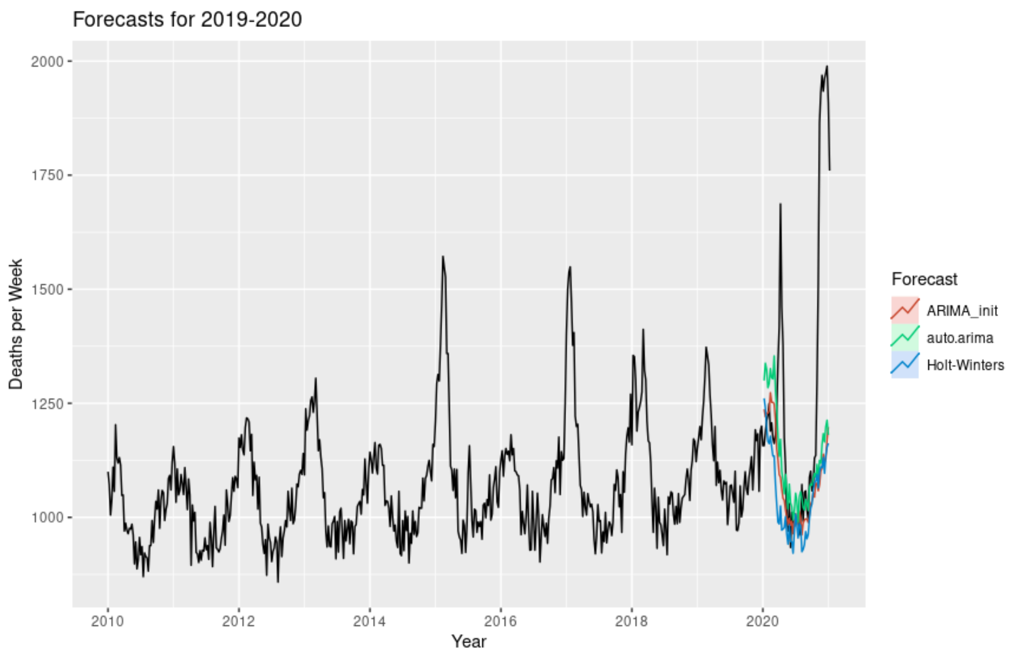

Figure 5.11: Comparison of all three methods, forecast from 2019-2020

Since all three models are based on a seasonal assumption, none of the models could successfully emulate the surge in deaths in the elderly caused by COVID-19.

5.4 Conclusion and Outlook

All three methods provide us with a good forecast for the Swiss deaths dataset from 2018-2019. However, real data is not always generated by a relatively simple model such as a seasonal ARIMA process. Furthermore, using the RMSE may not be the best indication of whether a model is accurate or not, and even within the scope of viewing the RMSE, we could also pose the question of whether we should minimise the one-step, two-step, or in our case, the 52-step MSE (p.319, [1]).

More complicated algorithms, such as Holt–Winters, are likely to be more flexible in practical forecasting problems. However, deciding which models to use is not inherently clear without performing the forecasts and comparing their respective errors.

Nonetheless, none of the models explored in this thesis were able to forecast the drastic change in deaths in 2020. Additional methods, such as choosing methods more suited towards epidemiology like the SIR model and adding parameters related to the reproduction number can be explored, and is beyond the scope of this thesis.

Bibliography

[1] P. J. Brockwell and R. A. Davis, *Introduction to Time Series and Forecasting*, 3rd ed. Springer, 2016.

[2] J. Brownlee. White noise time series with python. [Online]. Available: https://machinelearningmastery.com/white-noise-time-series-python/

[3] S. Palachy. Detecting stationarity in time series data. [Online]. Available: https://www.kdnuggets.com/2019/08/stationarity-time-series-data.html

[4] C. Chatfield, *The Analysis of Time Series: Theory and Practice*, 1st ed. Springer, 1975. [Online]. Available: https://doi.org/10.1007/978-1-4899-2925-9

[5] P. J. Brockwell and R. A. Davis, *Time Series: Theory and Methods*, 2nd ed. Springer, 2006.

[6] A. Coghlan, "A little book of r for time series," pp. 26–60, 2018. [Online]. Available: https://buildmedia.readthedocs.org/media/pdf/a-little-book-of-r-for-time-series/latest/a-little-book-of-r-for-time-series.pdf

[7] S. F. S. Office. Weekly number of deaths, 2010-2019. [Online]. Available: https://www.bfs.admin.ch/bfs/en/home/statistics/health/state-health/mortality-causes-death.assetdetail.12607335.html

[8] ——. Weekly number of deaths, 2010-2019. [Online]. Available: https://www.bfs.admin.ch/bfs/en/home/statistics/health/state-health/mortality-causes-death.assetdetail.16306577.html

[9] G. A. Rob J Hyndman. Forecasting: Principles and practice, 2nd ed. [Online]. Available: https://otexts.com/fpp2/

[10] M. W. S. Shapiro, "An analysis of variance test for normality," *Biometrika*, vol. 52, pp. 591–611, 1965. [Online]. Available: https://doi.org/10.1093/biomet/52.3-4.591

[11] R. J. Hyndman. https://robjhyndman.com/hyndsight/arimaconstants/. [Online]. Available: https://www.rdocumentation.org/packages/forecast/versions/8.13/topics/Arima

[12] ——. https://robjhyndman.com/hyndsight/arimaconstants/. [Online]. Available: https://otexts.com/fpp2/

[13] E. Parzen, "Ararma models for time series analysis and forecasting," *Journal of Forecasting*, vol. 1, pp. 67–82, 1982. [Online]. Available: https://doi.org/10.1002/for.3980010108

Appendix A

Code

The code used in Chapter 5 can be found here, and is loosely based on the procedures explored in [9] and [6]. All files can also be found under the github repository at `https://github.com/wayne-zeng/time-series-forecasting`. In case the links to the Swiss Federal Health Department do not work anymore, csv files from March 2021 are also included in the repository.

The function `checkresiduals()` from the library `forecast` plots a histogram of the forecast errors with an overlaid normal curve that has mean zero and the same standard deviation as the distribution of forecast errors.

A.1 Utilities

```r
# util.R by Wayne Zeng, March 2021
# Utilities for Bachelor Thesis

# stats, forecast and ggplot2 libraries required for SARIMA and Holt-Winters
library(stats)
library(forecast)
library(ggplot2)

# Import Swiss deaths data from 2020
df_20 <- read.csv("https://www.bfs.admin.ch/bfsstatic/dam/assets/16104419/appendix", sep=";")
# Import Swiss deaths data from 2010-2019
df_10_19 <- read.csv("https://www.bfs.admin.ch/bfsstatic/dam/assets/12607335/master", sep=";")

# Filtering out data for over 65s for 2010-2019, 2010-2018 and 2020.
old_10_19 <- subset(df_10_19, Age=="65+")
old_10_18 <- subset(old_10_19, CY!=2018)

# Data from 2020 has different format to previous years, so column titles were adjusted.
old_20 <- subset(df_20, Year==2020 & Age=="65+      ")
colnames(old_20) <- colnames(old_10_19)
```

```r

# Concatenate all data to one dataset
old <- rbind(old_10_19, old_20)

# Extracts deaths data and converts data to a time series object.
# frequency=52 to ensure that the period is over 1y=52w.
deaths_10_19 <- ts(old_10_19$NumberOfDeaths, start=2010, frequency=52)
deaths_10_18 <- ts(old_10_18$NumberOfDeaths, start=2010, frequency=52)
deaths <- ts(old$NumberOfDeaths, start=2010, frequency=52)
```

Listing A.1: Utility file used for later code

A.2 SARIMA

```r
# ARIMA.R by Wayne Zeng, March 2021
# Forecasting using Seasonal Arima for Bachelor Thesis

# Import Utilities File
source("util.R")

# Plot Time Series as well as ACF, PACF using ggtsdisplay
ggtsdisplay(deaths_10_18, xlab="Year", ylab="no. of Deaths per Week", main=
    "Swiss Weekly Deaths (65+) from 2010-2018")

# Perform differencing once, resulting in a zero-mean stationary time
    series
deaths_18_diff <- diff(deaths_10_18, differences=1)
ggtsdisplay(deaths_18_diff, xlab="Year", main="1-Differenced Swiss Deaths
    2010-2018")

# Initial guess using SARIMA(0,1,1)x(0,1,1). checkresiduals plots the time
    series,
# ACF and residual histogram.
fit_init <- Arima(deaths_10_18,order=c(0,1,1), seasonal=c(0,1,1))
fit_init
res_init <- residuals(fit_init)
checkresiduals(res_init, xlab="Year", main="Initial ARIMA Estimation of
    Residuals 2010-2018")
# Perform Shapiro-Wilk test to check for normality of noise
shapiro.test(res_init)
# forecasts using ARIMA model and plots the forecast with 80% and 95%
    quantiles
arima_forecast_init <- forecast(fit_init, h=52)
autoplot(window(deaths_10_19, start=2010), main="Forecast of ARIMA(0,1,1)
    (0,1,1)[52], 2018-2019") +
  autolayer(arima_forecast_init, PI=TRUE)
# Calculates RMSE between forecast and real data
accuracy(arima_forecast_init, deaths_10_19)

# Guess using auto.arima, results in modelling with SARIMA
fit_auto <- auto.arima(deaths_10_18)
```

```r
fit_auto
res_auto <- residuals(fit_auto)
checkresiduals(res_auto, xlab="Year", main="Automatic ARIMA Estimation of
    Residuals 2010-2018")
# Perform Shapiro-Wilk test to check for normality of noise
shapiro.test(res_auto)
# Forecast auto model
arima_forecast_auto <- forecast(fit_auto, h=52)
autoplot(window(deaths_10_19, start=2010), main="Forecast of ARIMA(1,0,0)
    (1,1,0)[52] with drift, 2018-2019") +
  autolayer(arima_forecast_auto, PI=TRUE) +  xlab("Year") + ylab("Deaths
    per Week")

# stepwise=FALSE and approximation=FALSE makes auto.arima look harder,
# however the code does not stop running. Execute at own risk.
# pre2020_auto2 <- auto.arima(pre2020_ts, stepwise=FALSE, approximation=
    FALSE)

# Calculates RMSE between forecast and real data
accuracy(arima_forecast_auto, deaths_10_19)
```

Listing A.2: Script used to generate seasonal ARIMA model forecasts

A.3 Holt–Winters

```r
# HoltWinters.R by Wayne Zeng, March 2021
# Forecasting using HW for Bachelor Thesis

# Import Utilities File
source("util.R")

# Forecast Using data from 2010-2018 as training data, 2019 as test data.

# Training
hw_forecast_18 <- HoltWinters(deaths_10_18)
hw_forecast_18
# Plot to see if smoothing looks correct
plot(hw_forecast_18, xlab="Year", ylab="Deaths per Week")
# View Residuals
hw_res <- residuals(hw_forecast_18)
checkresiduals(hw_res, xlab="Year", main="Holt-Winters Estimation of
    Residuals 2010-2018")
shapiro.test(hw_res)

# Forecast training data to 2019, plot data
hw_forecast_18_19 <- forecast(hw_forecast_18, h=52)
# autoplot(hw_forecast_18_19, xlab="Year", ylab="Deaths per Week")
autoplot(window(deaths_10_19, start=2010), main = "HW Forecast 2018-2019:
    alpha = .6041, beta = 0, gamma=.7080") +
  autolayer(arima_forecast_auto, PI=TRUE) + xlab("Year") + ylab("Deaths per
     Week")
```

```r

# Calculates RMSE between forecast and real data
accuracy(hw_forecast_18_19, deaths_10_19)
```

Listing A.3: Holt Winters Method

A.4 Comparisons

```r
# comparison.R by Wayne Zeng, March 2021
# Comparison of SARIMA and HW for Bachelor Thesis

# Import previous files
source("util.R")
source("ARIMA.R")
source("HoltWinters.R")

# Plot all three forecasts and original data onto plot
autoplot(window(deaths_10_19, start=2010)) +
  autolayer(arima_forecast_init, series="ARIMA_init", PI=FALSE) +
  autolayer(arima_forecast_auto, series="auto.arima", PI=FALSE) +
  autolayer(hw_forecast_18_19, series="Holt-Winters", PI=FALSE) +
  xlab("Year") + ylab("Deaths per Week") +
  ggtitle("Forecasts for 2018-2019") +
  guides(colour=guide_legend(title="Forecast"))

# Make accuracies matrix to calculate RMSE
accuracies <- rbind(accuracy(arima_forecast_init, deaths_10_19)[,"RMSE"],
                    accuracy(arima_forecast_auto, deaths_10_19)[,"RMSE"],
                    accuracy(hw_forecast_18_19, deaths_10_19)[,"RMSE"])
colnames(accuracies) <- c("Training set RMSE", "Test set RMSE")
rownames(accuracies) <- c("Initial Arima Forecast", "Auto Arima Forecast",
    "Holt-Winters Forecast")

accuracies


# Now make predictions for 2020 based on data from 2010-19,
# errors will be large, showing effect of COVID-19

### ARIMA ###

# Perform differencing once, resulting in a zero-mean stationary time
    series
deaths_19_diff <- diff(deaths_10_19, differences=1)
# Initial guess using SARIMA(0,1,1)x(0,1,1)
fit_init_19 <- Arima(deaths_10_19, order=c(0,1,1), seasonal=c(0,1,1))
fit_init_19
arima_forecast_init_19 <- forecast(fit_init_19, h=52)
# Guess using auto.arima, results in modelling with SARIMA
fit_auto_19 <- auto.arima(deaths_10_19)
fit_auto_19
```

```r
arima_forecast_auto_19 <- forecast(fit_auto_19, h=52)


## HW ##

# Training
hw_forecast_19 <- HoltWinters(deaths_10_19)
# Forecast training data to 2020, plot data
hw_forecast_19_20 <- forecast(hw_forecast_19, h=52)


# Plot all three forecasts and original data onto plot
autoplot(window(deaths, start=2010)) +
  autolayer(arima_forecast_init_19, series="ARIMA_init", PI=FALSE) +
  autolayer(arima_forecast_auto_19, series="auto.arima", PI=FALSE) +
  autolayer(hw_forecast_19_20, series="Holt-Winters", PI=FALSE) +
  xlab("Year") + ylab("Deaths per Week") +
  ggtitle("Forecasts for 2019-2020") +
  guides(colour=guide_legend(title="Forecast"))
```

Listing A.4: comparison of all three models

Appendix B

ARAR Method

The ARAR algorithm is an adapted version of the ARARMA algorithm, in which the idea is to apply automatically selected "memory-shortening" transformations (if necessary) to the data and then to fit an ARMA model to the transformed series. The ARAR algorithm we describe is a version of this in which the ARMA fitting step is replaced by the fitting of a subset AR model to the transformed data. More information can be found in Section 10.1 of [1] as well as [13]

This algorithm was originally considered for the thesis. However, the package `itsmr` used is outdated, provide very little flexibility. The frequency of the time series cannot be specified, unlike `forecast`, and the ARAR model was not able to produce any significant forecasts.

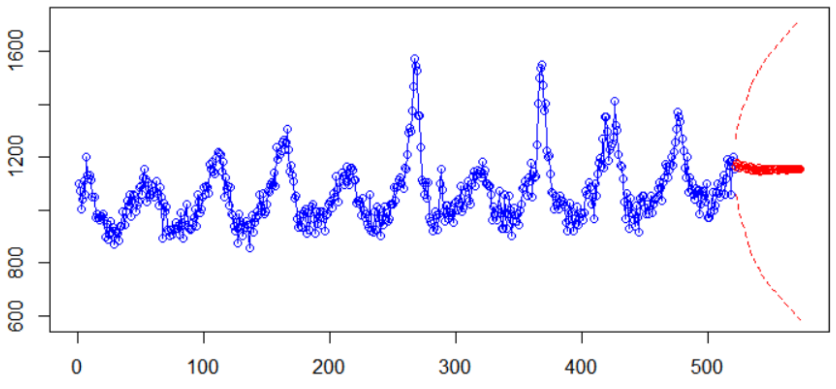

Figure B.1: ARAR Modelling of Deaths dataset age 65+ from 2010-2019 [7], failing to produce a meaningful forecast.

ETH

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

TIME SERIES: A COMPARATIVE STUDY OF FORECASTING METHODS

Authored by (in block letters):

For papers written by groups the names of all authors are required.

Name(s):	First name(s):
ZENG	WAYNE SIXING

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	Signature(s)
Zurich, 30th March 2021	[illegible]

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.