Stockholms universitet

Just a Little Bit
Forecasting the Bitcoin/USD exchange rate using ARIMA and Local Level models

Jakob Torgander

Kandidatuppsats i matematisk statistik
Bachelor Thesis in Mathematical Statistics

Kandidatuppsats 2019:26
Matematisk statistik
Juni 2019

www.math.su.se

Matematisk statistik
Matematiska institutionen
Stockholms universitet
106 91 Stockholm

Stockholm University

Mathematical Statistics
Stockholm University
Bachelor Thesis **2019:26**
http://www.math.su.se

Just a Little Bit
Forecasting the Bitcoin/USD exchange rate using ARIMA and Local Level models

Jakob Torgander*

June 2019

Abstract

In this thesis we will compare the abilities of the ARIMA and Local Level model classes to predict future values of the Bitcoin/USD exchange rate. In particular we will compare how these two models approach non-stationary time series, i.e. when the means and covariances of the series are assumed to be time dependent. We will also discuss how the Bitcoin price relates to the concept of market efficiency and the random walk hypothesis.

*Postal address: Mathematical Statistics, Stockholm University, SE-106 91, Sweden. E-mail: jakobtorgander@gmail.com. Supervisor: Mathias Millberg Lindholm Kristofer Lindensjö .

Acknowledgements

This is a Bachelor's thesis of 15 ECTS in Mathematical Statistics at the Department of Mathematics at Stockholm University. I would like to thank my supervisors Mathias Millberg Lindholm and Kristofer Lindensjö for their guidance and valuable suggestions. I would also like to thank my friends, family and fellow students for their support and advice.

Contents

1 Introduction

Any person who wants to plan for his or her future inevitably will face the question: "How similar will the future be to the past?" Although relevant for many fields, this question is particularly urgent in the world of finance, where the nature of financial investments raises the need for its actors to be able to make qualified predictions of the future. Time series analysis methods have been developed as a tool to help us in trying to answer this question.

During this thesis we will study how time series modeling can be used to forecast future US-dollar log prices of the digital cryptocurrency Bitcoin. In particular we will compare two time series models which differ significantly in their approach to modeling the *non-stationary* tendencies of the Bitcoin series, meaning that the series have time dependent mean values.

We will first show how the Bitcoin log prices can be modeled using the classic **Autoregressive Integrated Moving Average (ARIMA)** model, where the problem of a time dependent mean is solved using so called *differencing*. This model will thereafter be compared against the **Local Level** model, where the time dependent mean instead is modeled explicitly. This will also give us the opportunity to utilize a renowned technique from the field of signal processing, known as the **Kalman filter**.

Through our study we will see that the two model classes not only differ in their ways to technically deal with the problem of non-stationarity, but also in their philosophical approach to how future values can be forecasted. We will see how the ARIMA model focuses on using historical patterns to predict future values, whereas the Local Level model uses a more "memoryless" approach.

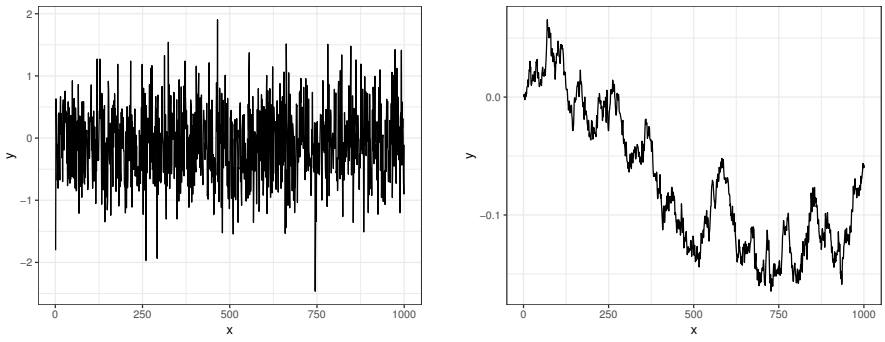


Figure 1: Examples of a stationary and non-stationary time series

When the forecasting abilities of these two models are put to test on the Bitcoin data set and evaluated using MSE, RMSE and MAE measures, we will see how the Local Level model gives a slightly better forecasting performance compared to the ARIMA model. We will however see that the Kalman filter in this situation will make our Local Level model to behave similarly to a **random walk** and that if our selected ARIMA(3,1,2) model is exchanged with an ARIMA(0,1,0), i.e a random walk model, the fitted Local Level and ARIMA models will show **equivalent** forecasting results.

Finally, we will see how splitting our given series into two parts and repeating our comparison, results in an even smaller difference between our models. Doing this we will also show how values past 2014-11-11 behave in accordance to the **random walk hypothesis** and discuss what consequences this might have on the forecasting abilities of our models.

2 The ARIMA Model

In this section we first describe the concepts leading up to our first model candidate: the ARIMA model. We then describe the necessary steps for model fitting and show how the model can be used to forecast future values. Unless otherwise stated, the following subsections are based on (Tsay, 2005, Chapter 2).

2.1 Autoregressive Moving Average Models (ARMA)

Let y_t be an observed value at time point t and p, q be nonnegative integers. An ARMA(p, q) model can then be written as:

$$y_t = \phi_0 + \sum_{i=1}^{p} \phi_i y_{t-i} + z_t - \sum_{i=1}^{q} \theta_i z_{t-i}, \tag{1}$$

where $\{z_t\}$ is a sequence of independent and identically distributed random variables with finite variance and mean, referred to as a *white noise* series. Throughout this thesis we will additionally assume z_t to be normally distributed with $E[z_t] = 0$ and $\text{Var}(z_t) = \sigma_a^2$. The series $\{z_t\}$ is in this case referred to as a *Gaussian* white noise series. The ARMA model describes the current value of y_t as a linear combination of its previous values and noise terms z_t, also known as *shocks*. The ARMA(p, q) model can in turn be thought of as a combination of a pure autoregressive model (AR) of order p, on the form

$$y_t = \phi_0 + \sum_{i=1}^{p} \phi_i y_{t-i} + z_t$$

and a moving average model (MA) of order q, written as

$$y_t = z_t - \sum_{i=1}^{q} \theta_i z_{t-i}.$$

This means that the ARMA class covers all AR(p)- and MA(q) models together with their combinations. The parameters $\phi_0, \ldots \phi_p, \theta_1, \ldots, \theta_q$ are typically estimated using **maximum likelihood estimation** according to (Box, Jenkins, & Reinsel, 2008, Chapter 7).

2.2 Stationarity/Non-stationarity

We here proceed to introduce the concept of stationarity. This is a central concept in time series modeling and will prove to be an important part for our future forecasting process.

A time series is said to be *strictly stationary* if the joint distribution of $(y_{t_1}, ..., y_{t_k})$ is identical to the distribution of $(y_{t_1+t}, ..., y_{t_k+t})$ for all t. Here k is an arbitrary positive integer and $(t_1, \ldots, t_k)$ are k positive integers. The joint distribution of $(y_{t_1}, ..., y_{t_k})$ is then said to be *time invariant*. As this condition often is considered too strict, a weaker definition of stationarity can instead be used. A time series is said to be *weakly stationary* if for all t:

- $E(y_t) = \mu$, where μ is a constant and
- $\text{Cov}(y_t, y_{t-l}) = \gamma_l$, only depending on l.

For our previously introduced ARMA model described in Eq. (1) it can be shown that a sufficient criterion for the model to be weakly stationary (see (Tsay, 2005, pp. 38-46)) is that all of the solutions to the so called *characteristic equation*

$$1 - \phi_1 x - \phi_2 x^2 - \cdots - \phi_p x^p, \tag{2}$$

should be less than one in absolute value. Under the assumption of weak stationarity the mean of the ARMA(p, q) model can be calculated in the following way:

$$\begin{aligned} \text{E}[y_t] &= \text{E}[\phi_0 + \sum_{i=1}^{p} \phi_i y_{t-i} + z_t - \sum_{i=1}^{q} \theta_i z_{t-i}] \\ &= \phi_0 + \text{E}[\sum_{i=1}^{p} \phi_i y_{t-i}] = \phi_0 + \sum_{i=1}^{p} \phi_i \, \text{E}[y_t], \end{aligned} \tag{3}$$

where the assumption of weak stationarity is used in the last equality. Note that we in this calculation also use that $\text{E}[z_t] = 0$ since we assume $\{z_t\}$ to be a Gaussian white noise series. This implies:

$$\text{E}[y_t] = \frac{\phi_0}{1 - \phi_1 - \cdots - \phi_p},$$

i.e. the mean of the series is a constant depending on the model parameters.

2.3 Forecasting

One of the main applications of time series modeling is to forecast future values. For our ARMA model this corresponds to calculating the n-step ahead forecast at horizon h defined as

$$\hat{y}_h(n) = \mathrm{E}[y_{h+n}|\mathcal{F}_h].$$

Here $\mathcal{F}_h$ represents the available information at time h, including previous observations, shocks and model parameter values. In this thesis we will restrict ourselves to only using the 1-step ahead forecast, which for the ARMA(p, q) model can be written as:

$$\begin{aligned}\hat{y}_h(1) &= \mathrm{E}[\phi_0 + \sum_{i=1}^{p} \phi_i y_{h+1-i} + z_h - \sum_{i=1}^{q} \theta_i z_{h+1-i}|\mathcal{F}_h] \\ &= \phi_0 + \sum_{i=1}^{p} \phi_i y_{h+1-i} - \sum_{i=1}^{q} \theta_i z_{h+1-i},\end{aligned} \tag{4}$$

using that $\mathrm{E}[a_i] = 0$ for $i > h$. This result can in turn be used to define the 1-step forecast error:

$$e_h(1) = y_{h+1} - \hat{y}_h(1).$$

Using Eq. (4) we get for the ARMA model $e_h(1) = z_t$, which later on will be useful when evaluating our fitted models.

2.4 Unit Root Non-stationarity

In Section 2.2 we could see how the calculation of the mean of the ARMA model was obtained using the assumption of weak stationarity, corresponding to the solutions of the characteristic equation (Eq. (2)) all being less than one in absolute value. We now consider the situation when at least one of the solutions are equal to one. The model is then referred to as being *unit root non-stationary*, for which the calculations in Eq. (3) no longer applies since the mean now is time dependent. To show how this problem can be solved we introduce the *back shift* operator B, which is defined such that

$$By_t = y_{t-1}.$$

It turns out that by applying B on the full series $\{y_t\}$ the unit root can in some situations be differenced away, with the resulting series now being stationary. This process is called *differencing* and may have to be iterated if more than one unit root exists in the model. To illustrate a situation where the back shift operator can be successfully used we consider the so called **random walk** model on the form:

$$y_t = y_{t-1} + z_t, \quad z_t \sim N(0, \sigma^2). \tag{5}$$

This model can in turn be seen as an ARMA(1,0) model with $\phi_0 = 0$, and $\phi_1 = 1$. Using our results from Section 2.2 we see that this model is unit non-stationary. This implicates that the calculations of the model mean $\mathrm{E}[y_t]$ collapses into:

$$\mathrm{E}[y_t] = \mathrm{E}[y_{t-1}],$$

since we no longer can use any assumptions of stationarity to simplify this further. To solve this problem we apply the back shift operator on Eq. (5) in the following way:

$$(1 - B)y_t = y_t - y_{t-1} = z_t.$$

We have here successfully differenced away the unit root term y_{t-1} with the resulting series now being clearly weakly stationary since both

$$\mathrm{E}[\Delta y_t] = \mathrm{E}[y_t - y_{t-1}] = \mathrm{E}[z_t] = 0,$$

$$\mathrm{Cov}(\Delta y_t, \Delta y_{t-l}) = \mathrm{Cov}(z_t, z_{t-l}) = 0$$

are independent of t. Differencing constitutes the main component of the ARIMA model which will be defined in the next section.

2.5 Autoregressive Integrated Moving Average Models (ARIMA)

Using our results from the previous sections we are now ready to define the first model candidate for our upcoming model comparison. Using the back shift operator B and the ARMA(p, q) model, the ARIMA(p, d, q) model can be defined in the following way:

Definition: We say that $\{y_t\}$ is an ARIMA(p, d, q) process if

$$y_t^* = (1 - B)^d y_t \tag{6}$$

follows an ARMA(p, q) process. Note that forecasting with the ARIMA model is done by first calculating the corresponding ARMA forecast according to Section 2.3 and then use Eq. (6) to express this forecast in terms of the original series $\{y_t\}$. For instance, the 1-step forecast $\hat{y}_h(1)$ for an ARIMA$(p, 1, q)$ model can be calculated in the following way:

$$\begin{aligned}\hat{y}_h(1) &= \mathrm{E}[y_{h+1}|\mathcal{F}_h] = \mathrm{E}[(y_{h+1} - y_h) + y_h|\mathcal{F}_h] \\ &= \mathrm{E}[y_{h+1}^* + y_h|\mathcal{F}_h] = \hat{y}_h^*(1) + y_h,\end{aligned}$$

where $\hat{y}_h^*(1)$ is the 1-step forecast of the corresponding ARMA(p, q) model. Also note that

$$\begin{aligned}e_h(1) &= y_{h+1} - \hat{y}_h(1) = y_{h+1} - (\hat{y}_h^*(1) + y_h) \\ &= y_{h+1}^* - \hat{y}_h^*(1) = z_t,\end{aligned}$$

according to our results from Section 2.3. This means that the 1-step forecast errors for the ARIMA$(p, 1, q)$ model are the same as for an ARMA(p, q) model.

3 The Local Level Model

In the previous sections we have shown how the ARIMA model uses differencing in order to model non-stationary time series. Following the derivations and notation of (Durbin & Koopman, 2012, Chapter 2) we will now present a different method for handling non-stationarity by considering the **Local Level** model, defined in the following way:

$$y_t = \alpha_t + \varepsilon_t, \quad \varepsilon_t \sim N(0, \sigma_\varepsilon^2), \tag{7}$$

$$\alpha_{t+1} = \alpha_t + \eta_t, \quad \eta_t \sim N(0, \sigma_\eta^2). \tag{8}$$

We here view Eq. (7) to be an *observed* process with observed value y_t and Eq. (8) to be an *unobserved* random walk with unobserved *state* α_t. If we consider α_t to be the mean of the model, the Local Level model is clearly non-stationary since α_t is time dependent. In the following sections we will describe how we, given an observed value y_t, can use *Kalman filtering* to recover α_t and remove the noise terms ε_t and η_t. We will then show how this can be used to forecast future values of our series.

3.1 Linear State Space Models

In order to introduce the Kalman filter we first need to introduce *Linear State Space Models* as described by (Durbin & Koopman, 2012, pp. 43-44). Let y_t be a $p \times 1$ vector of observations and α_t be an unobserved $m \times 1$ vector called the *state vector*. The general linear Gaussian state space model can then be written in the following way:

$$y_t = Z_t\alpha_t + \varepsilon_t, \qquad \varepsilon_t \sim N(0, H_t) \tag{9}$$

$$\alpha_{t+1} = T_t\alpha_t + R_t\eta_t, \qquad \eta_t \sim N(0, Q_t), \quad t = 1, \dots n \tag{10}$$

where the dimensions of the components are presented in Table 1 below. Returning to our Local Level model presentation in Eq. (7)-(8) we can observe how this model constitutes the most simple case of the linear state space model, with all p, m, r, Z_t, T_t and R_t equal to one, $H_t = \sigma_\varepsilon^2$ and $Q_t = \sigma_\eta^2$. Similarly to the previous section we view the state vector α_t to be the object of our primary interest and which we want to be distinguished from the observation vector y_t.

Vector		**Matrix**	
y_t	$p \times 1$	Z_t	$p \times m$
α_t	$m \times 1$	T_t	$m \times m$
ε_t	$p \times 1$	H_t	$p \times p$
η_t	$r \times 1$	R_t	$m \times r$
		Q_t	$r \times r$

Table 1: Vector and matrix dimensions for the general linear state space model

3.2 The Kalman Filter

This section introduces the Kalman filter together with its main algorithm, referred to as the **Kalman recursions**. Following (Durbin & Koopman, 2012, pp. 82-85) we here give a brief introduction to the recursions and how they relate to the forecasting of future values. As the derivation of the recursions is very heavy on notation and includes rather tedious calculations, the interested reader is referred to the appendix for a more complete derivation.

As previously mentioned, the main purpose of the Kalman filter is, given a vector of observations y_t and a linear state space form as described in Section 3.1, to calculate the state vector α_t. The basic idea behind this filtering process is to consider the *conditional* distribution of α_t given our previous observations $Y_t = (y_t, y_{t-1}, \dots, y_1)$. Using the assumptions of the linear state space model in Eq. (9)-(10) we can then assume $\alpha_t|Y_t$ and $\alpha_t|Y_{t-1}$ to be *normally* distributed with covariance matrices P_t and $P_{t|t}$ respectively. Our problem then reduces to the estimation of $\mathrm{E}[\alpha_t|Y_{t-1}] = a_t$ and $\mathrm{E}[\alpha_t|Y_t] = a_{t|t}$. Using these we can then use the observation equation

$$y_t = Z_t\alpha_t + \varepsilon_t, \quad \varepsilon_t \sim N(0, H_t)$$

from Section 3.1 to calculate $\mathrm{E}[\alpha_{t+1}|Y_t] = a_{t+1}$, which then represents the corresponding 1-step forecast of the model. This outlines the main purpose of the Kalman filter which somewhat loosely can be summarized into the following steps:

1. Use the distribution of $\alpha_t|Y_{t-1}$ from the previous step and calculate the *forecast error* $v_t = y_t - \mathrm{E}[y_t|Y_{t-1}]$,
2. Use the fact that $v_t|Y_{t-1}$ also is normally distributed to form the joint normal distribution of α_t and v_t given Y_{t-1},
3. Using rules for conditionally normal random variables, condition on v_t and calculate $\mathrm{E}[\alpha_t|Y_t] = a_{t|t}$ and $\mathrm{Var}(\alpha_t|Y_t) = P_{t|t}$,
4. Use the results from step 3 together with the observation equation to calculate $E[\alpha_{t+1}|Y_t] = a_{t+1}$ and $\mathrm{Var}(\alpha_{t+1}|Y_t) = P_{t+1}$.

The results from this can be summarized by the **Kalman recursions**, here defined as:

$$\begin{aligned} v_t &= y_t - Z_t a_t, & F_t &= Z_t P_t Z_t^T + H_t, \\ a_{t|t} &= a_t + P_t Z_t^T F_t^{-1} v_t, & P_{t|t} &= P_t - P_t Z_t^T F_t^{-1} Z_t P_t, \\ a_{t+1} &= T_t a_t + K_t v_t, & P_{t+1} &= T_t P_t (T_t - K_t Z_t)^T + R_t Q_t R_t^T, \end{aligned}$$

where $K_t = T_t P_t Z_t^T F_t^{-1}$ is referred to as the *Kalman gain*. Returning to our Local Level model the corresponding Kalman recursions can for this model be written as:

$$\begin{aligned} v_t &= y_t - a_t, & F_t &= P_t + \sigma_\varepsilon^2, \\ a_{t|t} &= a_t + K_t v_t, & P_{t|t} &= P_t(1 - K_t), \\ a_{t+1} &= a_t + K_t v_t, & P_{t+1} &= P_t(1 - K_t) + \sigma_\eta^2, \end{aligned} \tag{11}$$

where here $K_t = P_t / F_t$ (Durbin & Koopman, 2012, pp. 11-13). We can from Eq. (11) observe how the 1-step forecast a_{t+1} of the model can be thought of as a combination of the previous prediction a_t and a proportion K_t of how far off this prediction was from the actual observed value y_t.

As we have seen, the Kalman recursions have been derived under a normal distribution assumption. It can however be shown that the Kalman filter is considered to be a *minimum variance linear unbiased estimate* (MVLUE) even if the normal assumption does not hold (Durbin & Koopman, 2012, pp. 78-80), meaning that the filter has the smallest variance among all possible linear unbiased estimators.

3.3 Steady State Solution

An important property with the Kalman filter is that P_t will converge to a constant matrix $\bar{P}$, which according to (Durbin & Koopman, 2012, pp. 37-38) for the Local Level model is given as the solution to the equation

$$\bar{P} = \bar{P}\Big(1 - \frac{\bar{P}}{\bar{P} + \sigma_\varepsilon^2}\Big) + \sigma_\eta^2$$

If we let $x = \bar{P}/\sigma_\varepsilon^2$ and $q = \sigma_\eta^2/\sigma_\varepsilon^2$ the equation can be reduced to the quadratic equation

$$x^2 - xq - q = 0,$$

with the corresponding solution

$$x = \frac{(q + \sqrt{q^2 + 4q})}{2}.$$

Recalling the Kalman recursions for the Local Level model in Eq. (11) this result implicates that as P_t approaches $\bar{P}$ the magnitude of the Kalman gain

K_t eventually will be solely determined by the observation noise variance σ_ε^2. We will in our coming case study see how the calculation and analysis of $\bar{P}$ constitutes as a central part of the model *training* process, later to be described in Section 5.4.2.

3.4 Filter Initialization

As a result of the iterative nature of the Kalman Recursions (Section 3.2) the behavior of the algorithm in Eq. (11) is dependent on the initial values of a_1 and P_1. When these values are unknown, as often is the case, we can following (Durbin & Koopman, 2012, pp. 32) make use of so called *diffuse* initialization, which for the Local Level model means that we first let a_1 be an arbitrary value and then let the state variance P_1 tend to infinity. Using Eq. (11) we can see that after the first step of the Kalman filter we receive:

$$a_2 = a_1 + \frac{P_1}{P_1 + \sigma_\varepsilon^2}(y_1 - a_1) \to y_1,$$

$$P_2 = \frac{P_1}{P_1 + \sigma_\varepsilon^2}\sigma_\varepsilon^2 + \sigma_\eta^2 \to \sigma_\varepsilon^2 + \sigma_\eta^2,$$

as P_1 tends to infinity. We can after this proceed with the recursions as described in Section 3.2, now using the converged values of a_2 and P_2. Another alternative for initializing the Kalman filter for the Local Level model is to estimate α_1 by maximum likelihood theory (Durbin & Koopman, 2012, pp. 34). Note that we in this thesis will restrict ourselves to only using diffuse initialization.

3.5 Parameter Estimation

Recalling Eq. (11) we note that the Kalman recursions for the Local Level model include the noise variances σ_ε^2, σ_η^2, which hence need to be estimated before the filter can be used on our data. This can according to (Durbin & Koopman, 2012, pp. 34-35) be done using maximum likelihood estimation using the following expression for the log likelihood:

$$l(\sigma_\varepsilon^2, \sigma_\eta^2; y) = -\frac{n}{2}\log(2\pi) - \frac{1}{2}\sum_{t=1}^{n}\left(\log F_t + \frac{v_t^2}{F_t}\right), \tag{12}$$

where n is the sample size and $v_t = y_t - a_t$, $F_t = P_t + \sigma_\varepsilon^2$ are calculated using the Kalman recursions. However this again requires the initial states a_1 and P_1 to be known.

Following (Durbin & Koopman, 2012, pp. 35-36) a modified version of Eq. (12), using the diffuse initialization technique described in the previous section, can be written:

$$\begin{aligned} l(\sigma_\varepsilon^2, \sigma_\eta^2; y)_{\text{diffuse}} &= \lim_{P_1 \to \infty} \Big(l(\sigma_\varepsilon^2, \sigma_\eta^2; y) + \frac{1}{2} \log P_1 \Big) \\ &= -\frac{1}{2} \lim_{P_1 \to \infty} \Big(\log \frac{F_1}{P_1} + \frac{v_1^2}{F_1} \Big) - \frac{n}{2} \log(2\pi) - \frac{1}{2} \sum_{t=2}^{n} \Big(\log F_t + \frac{v_t^2}{F_t} \Big) \\ &= -\frac{n}{2} \log(2\pi) - \frac{1}{2} \sum_{t=2}^{n} \Big(\log F_t + \frac{v_t^2}{F_t} \Big). \end{aligned}$$

This is true since $F_1/P_1 \to 1$ and $v_1^2/F_1 \to 0$ as P_1 tends to infinity. This function can then be maximized numerically to provide us with estimations of σ_ε^2 and σ_η^2.

4 Model Selection & Diagnostics

In the following subsections we will introduce some tools that later on will be helpful for our coming model fitting and evaluation.

4.1 Augmented Dickey Fuller Test

We here present the Augmented Dickey-Fuller (ADF) test according to (Tsay, 2005, pp. 77). The test can be used to help us determine wether a time series can be considered stationary or not. Testing stationarity using the ADF-test corresponds to testing the existence of a unit root in an AR(p) process. This in turn corresponds to testing $H_0 : \beta = 1$, against $H_a : \beta < 1$ using the regression

$$y_t = c_t + \beta y_{t-1} + \sum_{i=1}^{p-1} \phi_i \Delta y_{t-i} + e_t,$$

where c_t is a deterministic function of t, e_t is an error term and $\Delta y_j = y_j - y_{j-1}$. Using least-squares estimation of β the hypothesis test can be carried out using the t-statistic

$$ADF = \frac{\hat{\beta} - 1}{\text{std}(\hat{\beta})},$$

where $\hat{\beta}$ denotes the least squares estimate of β.

4.2 Autocorrelation Function and the Ljung-Box Test

An important tool for model selection and diagnostics is the autocorrelation function (ACF) defined under the assumption of weak stationarity as

$$\rho_l = \frac{\text{Cov}(y_t, y_{t-l})}{\sqrt{\text{Var}(y_t)\,\text{Var}(y_{t-l})}} = \frac{\text{Cov}(y_t, y_{t-l})}{\text{Var}(y_t)} = \frac{\gamma_l}{\gamma_0},$$

(Tsay, 2005, pp. 31-32). The ACF function in turn constitutes the main component of the **Ljung-Box Portmanteau** test which is used to jointly test if multiple autocorrelations of y_t are zero. The Ljung-Box Portmanteau statistic is according to (Tsay, 2005, pp. 32-33) defined as

$$Q(m) = N(N+2) \sum_{l=1}^{m} \frac{\hat{\rho}_l^2}{N-l},$$

where N is the number of observations. The statistic can then be used to test $H_0 : \rho_1 = \cdots = \rho_m = 0$ against $H_a : \rho_i \neq 0$ for some $i \in \{1, \ldots, m\}$. It can be shown, under the assumption that $\{y_t\}$ are independent and identically distributed, that $Q(m)$ follows a chi-squared distribution with m degrees of

freedom. This means that H_0 is rejected if $Q(m) > \chi^2_\alpha$ for some appropriate quantile α. For the value of m (Tsay, 2005, pp. 33) suggests the choice of $m \approx \ln(N)$.

4.3 AIC, BIC & AICC

When comparing many models from the same model class the AICC, AIC and BIC criteria can be used as a measure to help us find a parsimonious model. The idea behind these criteria is to reward models with good fits and at the same time penalize complex, higher order models. Following (Held & Sabanés Bové, 2014, pp. 224-230), let $\boldsymbol{\theta}$ denote our model parameter vector of dimension k and $l(\hat{\boldsymbol{\theta}}_{ML})$ the maximized log likelihood function at the maximum likelihood estimate $\hat{\boldsymbol{\theta}}_{ML}$ of $\boldsymbol{\theta}$. Using this we first define the well known **Akaike's information criterion** as follows:

$$\text{AIC} = -2l(\hat{\boldsymbol{\theta}}_{ML}) + 2k \tag{13}$$

We then define the most common alternative to the AIC, called the **Bayesian information criterion**, in the following way:

$$\text{BIC} = -2l(\hat{\boldsymbol{\theta}}_{ML}) + 2k\log(N),$$

where N denotes the sample size. When fitting ARMA(p, q) models (Brockwell, Davis, & Calder, 2002, pp. 173) also suggests the use of a modified version of the AIC called the AICC which is defined as:

$$\text{AICC} = -2l(\hat{\boldsymbol{\theta}}_{ML}) + \frac{2(p+q+1)N}{(N-p-q-2)},$$

where here $k = \dim(\boldsymbol{\theta}) = p + q + 1$ and N again denoting the sample size. Returning to Eq. (13) we can observe that the AICC penalizes large order models harder than the AIC. This will according to (Brockwell et al., 2002) counteract the overfitting tendencies of the AIC. We can also see that both penalty factors are asymptotically equivalent as N tends to infinity.

4.4 Model Comparison Measures: MSE, RMSE & MAE

In the previous section we discussed how AICC, AIC and BIC can be used for model comparisons within the same model class. Since we in our coming case study want to compare fitted models from *different* model classes we need to introduce a set of new more "neutral" measures, which do not make any explicit judgements about the number of model parameters and instead primarily focus on the error terms of the fitted models. For this purpose we need to borrow some concepts and vocabulary from the field of statistical learning in order to formulate a procedure to compare our models.

Following (Friedman, Hastie, & Tibshirani, 2001, Chapter 7), the model comparison process can be conducted by first splitting our given data set into a *training* and *test* set. The training set will here represent our historical, known values for which we fit (train) our model, and the test set represents future unknown values for which we test the forecasting ability of our trained models.

Furthermore, let Y be our response variable, X a vector of explanatory variables and $\hat{f}(X)$ a prediction model estimated from our training set $\mathcal{T}$. The prediction performance for each model can then be assessed by applying each models to the test set and select the model that gives us the lowest *expected prediction error* defined as:

$$\text{Err} = \text{E}[L(Y, \hat{f}(X))] = \text{E}\left[\text{E}[L(Y, \hat{f}(X))|\mathcal{T}]\right], \tag{14}$$

for some *loss function* L. Since we are working with continuous data we will for our coming model comparison use the following loss functions:

$$L_{\text{MSE}}(Y, \hat{f}(X)) = (Y - \hat{f}(X))^2, \quad L_{\text{MAE}}(Y, \hat{f}(X)) = |Y - \hat{f}(X)|, \tag{15}$$

If we the then estimate the expected prediction error by:

$$\frac{1}{N}\sum_{i=1}^{N} L(y_i, \hat{f}(x_i)), \tag{16}$$

using the Law of Large numbers and inserting our loss functions from Eq. (15) in Equation (16) above, we then receive the following estimated expected prediction error measures

$$\text{MSE} = \frac{1}{N}\sum_{t=1}^{N}(y_i - \hat{f}(x_i))^2, \quad \text{MAE} = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{f}(x_i)|, \tag{17}$$

known as the estimated *mean square error* and *mean absolute error* respectively. Taking the square root of the mean squared error also gives us the unit corrected

$$\text{RMSE} = \sqrt{\text{MSE}},$$

known as *root mean square error*, now having the same unit as y_i which can be convenient when interpreting our results. For our time series purposes we will henceforth let $\hat{f}(x_t) = \text{E}[y_t|\mathcal{F}_{t-1}]$ according to our 1-step forecast definition in Section 2.3, which yields

$$\text{MSE} = \frac{1}{N}\sum_{t=1}^{N}(y_t - \text{E}[y_t|\mathcal{F}_{t-1}])^2, \quad \text{MAE} = \frac{1}{N}\sum_{t=1}^{N}|y_t - \text{E}[y_t|\mathcal{F}_{t-1}]|,$$

here instead indexing over time t.

5 Case Study

5.1 Data Description

Our model comparison will be based upon a data set containing the daily closing US-dollar (USD) prices of the digital cryptocurrency Bitcoin (BTC) during the period 2010-07-16 to 2019-01-25. Since its introduction in 2008 by acronym Satoshi Nakamoto, Bitcoin has served as a symbol for the digitalization of the modern society and has been a recurrent controversial topic in the public economic debate.

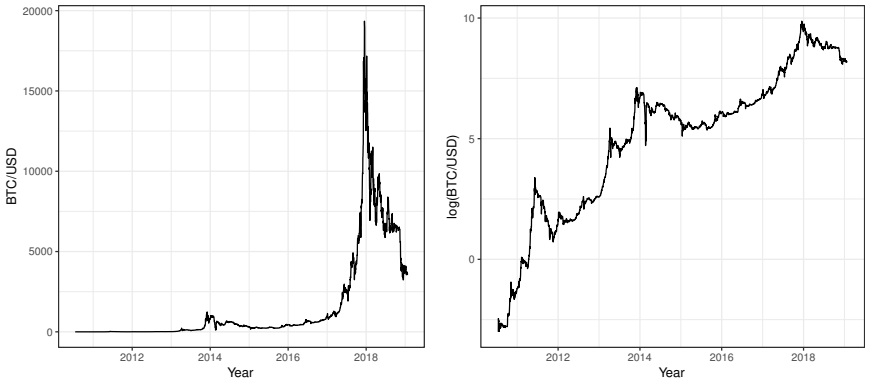
Bitcoin is an entirely decentralized virtual currency based on a peer-to-peer system. Individual coins are created through a so called *mining* process in which participants of the system compete to solve a mathematical problem while at the same time processing bitcoin transactions. According to (Antonopoulos, 2017, pp. 2) the Bitcoin technology can be summarized into the following four parts:

- A decentralized peer-to-peer network (the bitcoin protocol),
- A public transaction ledger (the blockchain),
- A set of rules for independent transaction validation and currency issuance (consensus rules),
- A mechanism for reaching global decentralized consensus on the valid blockchain (Proof-of-Work).

One important property of the Bitcoin protocol is that the mining function is regulated across the network in a way such that one new coin is created on average every 10 minutes. At the same time, the rate at which new coins are created is also regulated such that the total number of coins will reach a maximum of 21 million by the year 2140.

Besides being a representation of a brand new class of financial assets that are digital cryptocurrencies, one thing that makes the Bitcoin price a particular object of interest for mathematical modeling is its explosive historical development in combination with its drastically fluctuations. This can clearly be seen in Figure 2a where the Bitcoin price is plotted against time. We can here observe how the price ranges from an initial value of 0.04951 in 2010-07-16 to a maximum value of staggering 19345.49 USD in 2017-12-16. In combination with a standard deviation of 2972.353 USD from a mean value of 1565.745, it is clear that understanding the behavior of the Bitcoin price is an intriguing problem for any investor.

Figure 2



(a) Daily Bitcoin close prices in USD (b) Daily Bitcoin **log** prices in USD

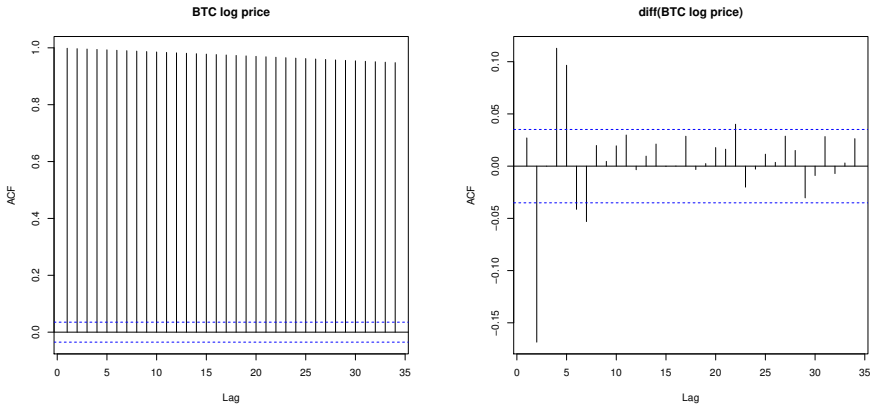
To compensate for the very large range and standard deviation of the price series we will throughout this case study instead conduct our analysis by taking the logarithm of the price series, as is also common practice in the financial literature (Campbell et al., 1997, pp. 11). The result of this can be seen in Figure 2b which now depicts a more controlled, bounded series. Another observation that can be made from both our plots in Figure 2 is that prices before year 2014 seems to behave differently compared to prices after year 2014. These observations will prove to be useful later during our modeling process.

We also note that the two price series are clearly non-stationary, with time dependent mean values. This can also be seen by studying the corresponding ACF-plot of the log price series depicted in Figure 3a, which clearly shows the slowly decaying behavior which according to (Brockwell et al., 2002, pp. 187-188) is characteristic for a non-stationary series.

5.2 The Random Walk Hypothesis

Before applying our model candidates on the Bitcoin data set introduced in the previous section, we first need to introduce the concept of the **random walk hypothesis**. The hypothesis states that stock prices evolve in accordance with a random walk and constitutes a central concept in the finance literature. The random walk hypothesis can in turn be seen as a variant of the concept of **market efficiency**, where a capital market is said to be efficient if it reflects all relevant information in determining security prices (Campbell et al., 1997, pp. 20). This concept is commonly paraphrased as that there under market efficiency is impossible to "beat" the market since all securities are priced correctly.

Figure 3



(a) ACF plot corresponding of the BTC log prices

(b) ACF plot corresponding of the BTC log returns

If we let p_t denote the price of a financial asset at time t, (Campbell et al., 1997) then states that the simplest version of the random walk hypothesis is the case when the dynamics of $\{p_t\}$ are given by:

$$p_t = \mu + p_{t-1} + \varepsilon_t, \quad \varepsilon_t \sim \text{IID}(0, \sigma^2), \tag{18}$$

where μ is the expected price change called *drift*, and $\text{IID}(0, \sigma^2)$ denotes ε_t being independently and identically distributed random variables with mean 0 and variance σ^2. Since this version of the hypothesis often is considered unrealistic for financial asset prices over longer time spans, the assumptions of i.i.d. distributed increments is usually relaxed, leading to what Campbell refers to as the random walk 3 model or **RW3**. This version of the random walk hypothesis instead assumes that the increments are dependent but *uncorrelated*. Specifically, the model assumes that

$$\text{Cov}(\varepsilon_t, \varepsilon_{t-k}) = 0, \text{ for all } k, \quad \text{Cov}(\varepsilon_t^2, \varepsilon_{t-k}^2) \neq 0, \text{ for some } k.$$

One important property with the random walk model is that we when calculating the corresponding 1-step forecast as described in Section 2.3 receive:

$$\hat{p}_h(1) = \text{E}[p_{h+1}|\mathcal{F}_h] = \text{E}[p_h + \varepsilon_h|\mathcal{F}_h] = p_h,$$

meaning that the forecast of the next value of the series is just its *current* value. This property makes the random walk model to be considered **unforecastable**. Because of this, testing the plausibility of the random walk hypothesis for our Bitcoin log price series is crucial in order to determine if applications of our Local Level and ARIMA candidates on this data are to be considered meaningful.

To do this we first observe that by applying the back shift operator B on Eq. (18) we receive:

$$\Delta p_t = (1 - B)p_t = \varepsilon_t \Rightarrow$$

$$\text{Cov}(\varepsilon_t, \varepsilon_{t-k}) = \text{Cov}(\Delta p_t, \Delta p_{t-k}).$$

This means that testing autocorrelation between the noise terms ε_t corresponds to testing autocorrelations for the differenced log price series $\{\Delta p_t\}$, in the finance literature referred to as the **log return** series (Campbell et al., 1997, pp. 11).

Testing autocorrelations for the differenced series can then easily be carried out using the the Ljung-Box test from Section 4.2, which for our data gives a *significant* result. The implication of this is that we can reject the corresponding null hypothesis of zero autocorrelation, which in turn provides evidence against the RW3. Another indication of this can also be observed in Figure 3b, where the ACF-plot of the log returns shows signs of significant correlations at lag 2,4,5, and 6. We can from this conclude that RW3 does not seem to hold for the full data set, opening up the possibility for that application of our ARIMA and Local Level models for this data could be useful.

5.3 Software

Before carrying out our model fitting and testing process we here briefly mention our use of computer software for this thesis. Our modeling is made entirely in the R-language. For the ARIMA modeling part we have, besides the basic packages, also included the use of the "Arima"- and "auto.arima"-functions from the 'forecast' package developed by (Hyndman, Khandakar, et al., 2007). We have also used the adf-function from the package 'tseries'. The code for the Kalman filtering is written by the author using R base functions and is tested to give similar results as the dlm-package when applied to the Bitcoin data.

5.4 Model Fitting

We are now ready to train our Local Level and ARIMA models on the Bitcoin data set. For this purpose we use the procedure as described in Section 4.4 by first splitting our given data set into a training and test set. We will here use a 70/30% split where the training set will consist of the 2180 first observations and test set will consist of the remaining 935 observations. Note that the order between the observations are maintained throughout this splitting process.

5.4.1 ARIMA

The goal of this section is to fit an appropriate ARIMA model to our data set. This is done using (Tsay, 2005, Chapter 2) with the purpose of finding a model that best will predict future data. Recall from Section 2.5 that $\{y_t\}$ follows an ARIMA(p, d, q) process if the d times differenced series $\{y_t^*\}$ follows an ARMA(p, q) process, i.e.

$$y_t^* = (1 - B)^d y_t = \phi_0 + \sum_{i=1}^{p} \phi_i y_{t-i}^* + z_t - \sum_{i=1}^{q} \theta_i z_{t-i}.$$

Our model selection process will be done using the following three steps:

1. Determining d, i.e the number of differencing needed to make the resulting model weakly stationary.
2. For $0 \leq p, q \leq M$, fit all ARIMA(p, d, q) models using ML-estimation and select the model with the lowest AICC value.
3. Conducting residual diagnostics for the selected model using the 1-step forecast errors from our selected model.

For step 1 we apply the Augmented Dickey Fuller test as described in Section 4.1 using the R adf.test function. Application of the test gives us a significant result for $d = 1$ but a nonsignificant result for $d = 2$, resulting in the selection of $d = 1$ as our differencing term.

We then proceed to step 2 and determine the p and q-values of the model. For this we use the R auto.arima function which for a max order $M = 5$ gives that $p = 3$ and $q = 2$ produces the model with the lowest AICC value. Repeating the process instead using the AIC and BIC criteria also results in the selection of $p = 3$ and $q = 2$, meaning that our model selection is independent of the choice of information criterion.

We finally proceed to the last step of our model selection process by controlling the model assumptions. We know from Section 2.1 that if our model holds for our data the series $\{z_t\}$ should resemble a Gaussian white noise series. Since z_t can be estimated with

$$e_t(1) = y_{t+1} - \hat{y}_t(1)$$

as shown in Section 2.5, the model assumptions can be tested using residual series $\{e_t(1)\}$. We first test the assumption of uncorrelated residuals by an application of the Ljung-Box test described in Section 4.2. Application of the test here gives us a *non-significant* result, meaning that we henceforth believe that the residuals are serially *uncorrelated*. Inspection of the corresponding QQ-plot (Figure 7 in the appendix) however shows signs of a more heavy tailed distribution than a normal distribution. Although making it implausible for the shocks z_t to be normally distributed, this result should not have a significant effect on our model's forecasting ability, since the main focus of our study is on *prediction* rather than *description*.

After this we can now consider the ARIMA model selection process completed, resulting in the selection of an **ARIMA(3,1,2)** model on the following form:

$$\begin{aligned} y_t^* &= y_t - y_{t-1} \\ &= 0.8666y_{t-1}^* - 0.7040y_{t-2}^* + 0.1989y_{t-3}^* + z_t - 0.8333z_{t-1} + 0.4973z_{t-2}. \end{aligned}$$

5.4.2 Local Level Model

We now turn to the fitting of the Local Level model. Recall from Section 3 that this model can be written

$$\begin{aligned} y_t &= \alpha_t + \varepsilon_t, \;\; \varepsilon_t \sim N(0, \sigma_\varepsilon^2) \\ \alpha_{t+1} &= \alpha_t + \eta_t, \;\; \eta_t \sim N(0, \sigma_\eta^2). \end{aligned}$$

Once again the fitting process can be decomposed into three steps, which for the Local Level model are as follows:

1. Estimate σ_ε^2 and σ_η^2 using ML-estimation through a first application of the Kalman filter on the training set.

2. Using the results from step 1, make another application of the Kalman filter on the training set and from this extract the last filtered state a_t and state variance P_t.

3. Using the forecast errors from step 2, conduct model diagnostics.

We first carry out the requested ML-estimation in step 1 using a first application of the Kalman filter according to Section 3.5. This results in the following variance estimations:

$$\hat{\sigma}_\varepsilon^2 = 0.000001607844, \ \hat{\sigma}_\eta^2 = 0.00587909. \tag{19}$$

From this result we can observe that the variance belonging to the observation noise is very close to zero. Returning to our description of the Local Level model in Eq. (7) this means that ε_t will be close to a constant, which in turn will have a significant effect on the behavior of the filtering process. This can be seen by recalling the Kalman Recursions from Section 3.2, which for the Local Level model could be written:

$$\begin{aligned} v_t &= y_t - a_t & F_t &= P_t + \sigma_\varepsilon^2 \\ a_{t|t} &= a_t + K_t v_t & P_{t|t} &= P_t(1 - K_t) \\ a_{t+1} &= a_t + K_t v_t & P_{t+1} &= P_t(1 - K_t) + \sigma_\eta^2, \end{aligned}$$

where $K_t = \frac{P_t}{F_t} = \frac{P_t}{P_t + \sigma_\varepsilon^2}$. If we now let σ_ε^2 tend to zero it follows that K_t will tend to one and as a result of this:

$$P_{t+1} \to \sigma_\eta^2, \quad a_{t+1} \to a_t + v_t = y_t \tag{20}$$

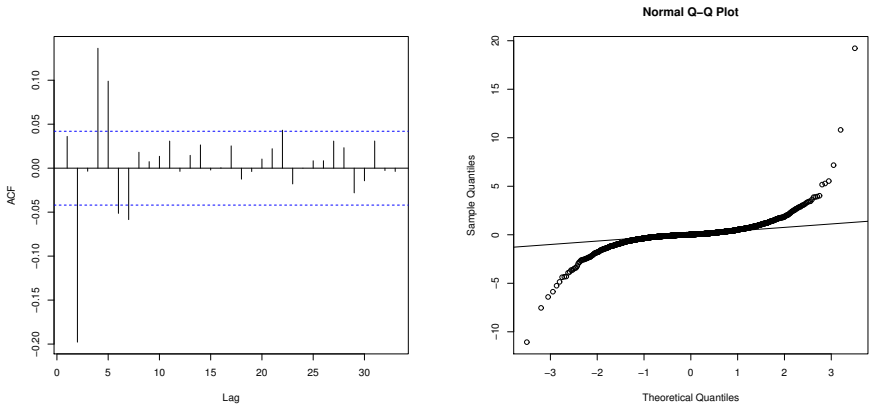
as $\sigma_\varepsilon^2 \to 0$. This means that for a small value of σ_ε^2 the influence of v_t on a_{t+1} will be maximized and the state variance P_t will be solely determined by σ_η^2. This implies that the 1-step forecast of the Local Level model collapses into the 1-step forecast belonging to the random walk model described in Section 5.2. Although hypothesis testing of our calculated σ_ε^2 in Eq. (19) indicates that we cannot reject that the true parameter is equal to zero, we will however proceed with the parameter included in the model, again considering the predictive objective of our study.

Using our variance estimates in Eq. (19) we then proceed to the second step of the training process and make another application of the Kalman filter on our training set. By extracting the resulting values for K_t and P_t we note that both K_t and P_t here have converged for $t \geq 3$, which is a clear consequence of our small $\hat{\sigma}_\varepsilon^2$ value. With a converged value of 0.005880697 we also note that P_t is very close to our estimation of σ_η^2 in Eq. (19), which is an expected result. We can also observe how the converged value of K_t will give us the following 1-step forecast:

$$\begin{aligned} a_{t+1} &= a_t + K_t v_t \\ &= a_t + 0.9997267(y_t - a_t) = 0.9997267 y_t + 0.0002733 a_t. \end{aligned}$$

This means that the dominating part of our prediction will be based on our current observation but including a small adjustment with respect to our previous prediction a_t.

Figure 4



(a) Residual ACF-plot for the trained Local Level model

(b) Residual QQ-plot for the trained Local Level model

Moving on to model diagnostics we follow (Durbin & Koopman, 2012, pp. 38) and test wether the standardized residuals

$$e_t = \frac{v_t}{\sqrt{F_t}}$$

can be considered to be normally distributed and serially uncorrelated. We first test the autocorrelation using another Ljung-Box test. In contrast to our ARIMA fitting we here receive a **significant** result, implying that the null hypothesis of uncorrelated standardized residuals here can be rejected. This can also be confirmed by inspecting the ACF-plot of the standardized residuals in Figure 4a, which shows signs of significant correlations at lag 2,4,5,6 and 7. We can also observe another sign of model misspecification by inspecting the QQ-plot of the residuals in Figure 4b. Similarly to our ARIMA model candidate the plot shows clear signs of a more heavy tailed distribution than a N(0,1) distribution, making assumptions of normal distributed residuals implausible.

Since our model diagnostics collectively speaks against that our training data could have been generated by a Local Level model as described by Eq. (7)-(8), we now have to ask ourselves what implications will this have for our future possibilities to use this model for forecasting? Fortunately we can here recall from Section 3.2 that the Kalman filter is considered to be MVLUE even though the error terms are not normally distributed. This together with our earlier argument that our main focus of our study here is on prediction rather than description means that we can move on with our model to the forecasting part of our case study.

5.5 Model Comparison

Using the results from Section 5.4.1 and 5.4.2 we are now ready to test which of the models that best will forecast future values of the Bitcoin log price series. We therefore apply our trained ARIMA(3,1,2) and Local Level models on the test set and for each data point in this set calculate the corresponding 1-step ahead forecasts as described in Section 2.3 and 3.2. Using these forecasts we then calculate the corresponding 1-step forecast errors and form the test quantities **MSE, RMSE** and **MAE** as described in Section 4.4. Since we in Section 5.4.2 observed that our trained Local Level model showed behavior similar to a random walk we here choose to also include a random walk model in our comparison.

The results are summarized in Table 1, where we firstly can observe how the performance of the Local Level model seems to be slightly better than the ARIMA$(3, 1, 2)$ model, regardless of the choice of loss function. Not surprisingly, we can also observe that the results from the Local Level model is practically identical to the results of the random walk model. Since the random walk model is contained within the ARIMA class as an ARIMA(0,1,0) model we can from this conclude that our ARIMA training process seems to have resulted in an overfitted model.

Recalling our training process in Section 5.4.2, where we could observe our Local Level model to show signs of misspecification in contrast to the significantly better specified ARIMA model, we next wish to control if the choice of proportion between the training and test set has any effect on our results before making any definitive statements about the ARIMA(3,1,2) model. We therefore repeat the training and testing process when varying the proportions between the training and test set.

The results from the above process can be observed in Figure 5 where we can see how the Local Level model performs better than the ARIMA(3,1,2) model for up to a 40/60% proportion between the training and test set. We can however observe how the results from the two models are equivalent for test set proportions higher than 60%. Finally, we can in Figure 5 observe how the ARIMA model seems to break down as the test set proportion approaches 100% while the Local Level model remains stable throughout all

Model	MSE	RMSE	MAE
ARIMA(3,1,2)	0.001966050	0.04434017	0.02995355
Local Level	0.001830771	0.04278751	0.02861004
Random walk	0.001830523	0.04278461	0.02859944

Table 2: Test error measures for the fitted Local Level, ARIMA and random walk models

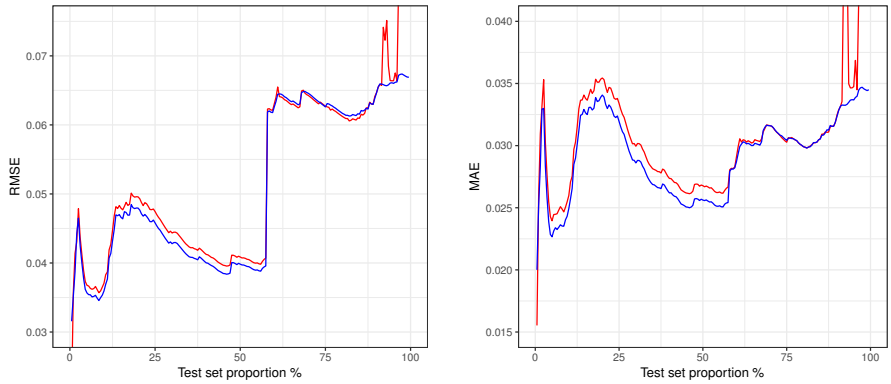


Figure 5: RMSE and MAE comparison between the Local Level model (blue) and ARIMA(3,1,2) model (red) for varying proportion sizes between the test and training set.

train/test set proportions. Although test set proportions approaching 100% are not very useful in practice, this result does however illustrate how the fast convergence of our trained Local Level model discussed in Section 5.4.2, here makes it perform reasonably well on smaller training sets.

Repeating this process with the ARIMA(3,1,2) model replaced by the (0,1,0) model results in the corresponding RMSE and MAE comparison plots now showing no visible difference between the Local Level and ARIMA model (See Figure 8 in the appendix). From this we can conclude that in terms of ARIMA modeling there seems to be no benefits for using a more complex model than a random walk when forecasting future values of the Bitcoin/USD log price series.

Before considering our model comparison to be completed we first want to address the sudden and radical increase in RMSE that can be observed in the middle of the left plot in Figure 5. It seems that for train/test set proportions over 42.5/57.5%, corresponding to a split point around the date 2014-03-01, our models seem to perform much worse. This observation brings us back to our initial data analysis in Section 5.1, where we in Figure 2 noted that the price series showed different behaviors for values before year 2014, compared to values after 2014.

It turns out that splitting our series at the date 2014-11-11 and then repeating the model fitting and forecasting process on the resulting two sub-series, the differences between our two models decrease significantly and also result in less overfitted ARIMA fittings. The results from the corresponding forecast evaluation can be found in Table 3-4 in appendix and show how

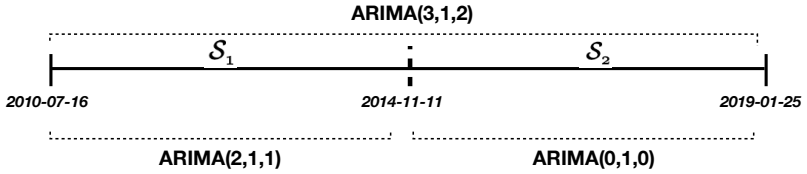


Figure 6: Splitting of the Bitcoin log price series and the resulting fitted ARIMA models for each sub-series

the error measures for the sub-series with values after 2014-11-11 denoted by $\mathcal{S}_2$, are very similar to the corresponding values for the *full* series shown in Table 2, whereas the error measures for the sub-series with values up to 2014-11-11 denoted by $\mathcal{S}_1$ are considerably higher in comparison.

Another important consequence of the above splitting process is that our ARIMA fitting for $\mathcal{S}_2$ this time also results in a random walk. Using this result we return to our discussion in Section 5.2 where we showed how inspections of the autocorrelations of the log return series for the full data set provided clear evidence against the random walk hypothesis (RW3). As both our Local Level and ARIMA model fitting for log prices after 2014-11-11 has resulted in some variation of a random walk we now have a reason to reevaluate our assumptions about the RW3. We therefore generate another Ljung-Box test for each of our two differenced sub-series, testing the null hypothesis of zero autocorrelation. As these tests now give us a **significant** result for $\mathcal{S}_2$ but still a non-significant result for $\mathcal{S}_1$ we therefore believe the random walk hypothesis to be **true** for $\mathcal{S}_2$ but not for $\mathcal{S}_1$. The same conclusions can also be reached by inspecting the corresponding ACF-plots in Figure 9 shown in appendix.

5.6 Conclusion

We can from this point declare the model comparison process to be completed, resulting in the following conclusions:

- The forecast performance of the Local Level model is slightly better than the ARIMA(3,1,2) when fitted to the full Bitcoin data set.
- The differences in performance between the models are very small if we instead consider the split data set with values before and after 2014-11-11 respectively.
- None of our fitted models outperforms a random walk.
- The random walk hypothesis seems to hold true for data past 2014-11-11 but not for data prior to this.

6 Discussion

The objective of this thesis has been to compare how Local Level and ARIMA models can be used to forecast future Bitcoin prices. For this purpose we have showed how these models can be fitted on a training set with daily closing log prices between 2010-07-16 and 2019-01-25. We were able to observe how this process resulted in the selection of an ARIMA(3,1,2) model and a Local Level model closely resembling a random walk. Applying these trained models on a test set and evaluating the corresponding 1-step forecasts we could then see a slightly stronger forecasting performance from the Local Level model compared to the ARIMA(3,1,2) model.

We could thereafter discover that, by splitting the original data at date 2014-11-11 and repeating the comparison process, the differences between the models would decrease significantly. Here we could also observe, for log prices past 2014-11-11, how the ARIMA and Local Level model fitting both resulted in a well specified random walk, hereby confirming the random walk hypothesis for this sub-series. Conversely, for values prior to 2014-11-11 we concluded that, although a model predicting data better than a random walk could not be found, significant autocorrelation of the log returns here provides evidence *against* the hypothesis. In conclusion, our case study shows that when considering the full data, set the Local Level model gives a slightly better performance than the corresponding ARIMA model, but equivalent results when instead considering the split data set.

Further, from an economic point of view our results show how Bitcoin for log prices after 2014-11-11 has come to adapt the behavior of a regular financial asset. This means that for the future development of the series the random walk hypothesis can be assumed to hold true, with future values as a result being considered unforecastable. Focusing on our modeling process, we conclude that the most noticeable difference between our Local Level and ARIMA modeling is how the Local Level model continuously interpreted the series as a random walk, whereas the ARIMA fitting tended to overlook the corresponding (0,1,0) model in favor of more complex models. Since we also could observe that none of our considered models would outperform a random walk in terms of forecasting, we can from this conclude that our ARIMA modeling constantly suffered from the risk of overfitting when information criteria was used for validation. It is possible that using different validation methods for the ARIMA model could have resulted in a better predictive model. We will here however instead provide a simpler, intuitive argument for why the risk of overfitting is inevitable when applying ARIMA models on financial problems similar to the one we have studied:

Belonging to a more complex model class the ARIMA model is able to use historical data in a more involved way, enabling it to detect more intricate patterns while using reasonably few parameters. However, unless we have good reasons to believe that future data actually will follow the same model as previous data, these historical patterns discovered by the ARIMA model may be more or less irrelevant when extrapolating into the future. During our case study we noted that fitting the ARIMA on a split data set would result in two radically different models, which in turn makes it implausible that the full data set could be seen as generated from the same model. In contrast, the Local Level model did not suffer from the problem of overfitting and also appear to have handled the ambivalence regarding the random walk hypothesis in a more accurate way.

One central component of this thesis has been to compare how our models approach non-stationary time series. Our results here indicate that, when 1-step forecast is used, explicitly modeling the time dependent mean using the Local Level model give similar result to the ARIMA's approach of instead considering a d times differenced series. Given this result, the simpler form of the Local Level model here shows a clear advantage in terms of interpretability, compared to the considerably more complex ARIMA model. We can from this see how this becomes especially relevant in situations when the differenced series do not have a specific interpretation such as the log returns, which may result in ARIMA models that are significantly harder to interpret.

Finally, we can see that the results from our case study opens up for a number of options for further research. Using our split data set as a starting point a natural continuation of the analysis for the part of the data where the random walk hypothesis seem to hold would be to proceed with volatility modeling by considering ARCH or GARCH effects according to (Tsay, 2005, Chapter 3). This would also serve as a suitable opportunity to investigate the possible t-distribution of the ARIMA and Local Level model residuals that we observed during our model fitting process. For the other data set we could instead consider the addition of exogenous variables to our models to see if we in that way can capture some of the behavior of the period before the series start to behave like a random walk.

References

Antonopoulos, A. M. (2017). *Mastering bitcoin: Programming the open blockchain*. O'Reilly Media, Inc.

Box, G. E., Jenkins, G. M., & Reinsel, G. C. (2008). *Time series analysis: forecasting and control*. John Wiley & Sons.

Brockwell, P. J., Davis, R. A., & Calder, M. V. (2002). *Introduction to time series and forecasting* (Vol. 2). Springer.

Campbell, J. Y., Champbell, J. J., Campbell, J. W., Lo, A. W., Lo, A. W.-C., & MacKinlay, A. C. (1997). *The econometrics of financial markets*. princeton University press.

Durbin, J., & Koopman, S. J. (2012). *Time series analysis by state space methods*.

Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning* (Vol. 1) (No. 10). Springer series in statistics New York.

Held, L., & Sabanés Bové, D. (2014). *Applied statistical inference*. Springer.

Hyndman, R. J., Khandakar, Y., et al. (2007). *Automatic time series for forecasting: the forecast package for r* (No. 6/07). Monash University, Department of Econometrics and Business Statistics.

Tsay, R. S. (2005). *Analysis of financial time series* (Vol. 543). John Wiley & Sons.

7 Appendix

7.1 Derivation of the Kalman Filter

We here provide a more extensive derivation of the Kalman filter, resulting in the Kalman Recursions introduced in Section 3.2. In order to do this we first need to introduce the following Lemma:

Lemma 1

Let X and Y be jointly normally distributed variables with

$$E\begin{pmatrix} X \\ Y \end{pmatrix} = \begin{pmatrix} \mu_X \\ \mu_Y \end{pmatrix}, \quad \text{Var}\begin{pmatrix} X \\ Y \end{pmatrix} = \begin{bmatrix} \Sigma_{XX} & \Sigma_{XY} \\ \Sigma_{XY}^T & \Sigma_{YY} \end{bmatrix},$$

where Σ_{YY} is a nonsingular matrix. The conditional distribution of $X|Y$ is then normal with

$$\begin{aligned} E(X|Y) &= \mu_X + \Sigma_{XY}\Sigma_{YY}^{-1}(Y - \mu_Y), \\ Var(X|Y) &= \Sigma_{XX} - \Sigma_{XY}\Sigma_{YY}^{-1}\Sigma_{XY}^T. \end{aligned}$$

Proof: see (Durbin & Koopman, 2012, pp. 77-78).

Derivation of the Kalman filter

Using Lemma 1 we here provide a derivation of the Kalman filter analogously with (Durbin & Koopman, 2012, pp. 82-85). Recall that the general linear Gaussian state space model is given by

$$\begin{aligned} y_t &= Z_t\alpha_t + \varepsilon_t, \quad \varepsilon_t \sim N(0, H_t) \\ \alpha_{t+1} &= T_t\alpha_t + R_t\eta_t, \quad \eta_t \sim N(0, Q_t), \quad t = 1, \dots n \end{aligned} \tag{21}$$

Let Y_{t-1} be the set of past observations $y_1 \dots, y_{t-1}$ for $t = 2, 3...$ If we then assume that the initial state $\alpha_1 \sim N(a_1, P_1)$, where a_1, P_1 are known we then proceed inductively to derive the conditional distribution of α_t given Y_t and from this extract

$$a_{t|t} = \text{E}[\alpha_t|Y_t], \quad P_{t|t} = \text{Var}(\alpha_t|Y_t)$$

called the filtered state and variance. Using this together with Eq. (21) we can then calculate

$$a_{t+1} = \text{E}[\alpha_{t+1}|Y_t], \quad P_{t+1} = \text{Var}(\alpha_{t+1}|Y_t)$$

which completes the induction step. The main idea behind deriving the distribution of $\alpha_t|Y_{t-1}$ is to incorporate the one step forecast errors v_t defined as

$$v_t = y_t - \text{E}[y_t|Y_{t-1}].$$

Using Eq. (21) this expression can be written as:

$$v_t = y_t - \mathrm{E}[Z_t\alpha_t + \varepsilon_t|Y_{t-1}] = y_t - Z_t a_t.$$

Rewriting this equality we get $y_t = v_t + Z_t a_t$ which means that if we know v_t and Y_{t-1} we can calculate y_t and $Y_t = (y_t, y_{t-1}, ...)$. Consequently:

$$p(\alpha_t|Y_t) = p(\alpha_t|Y_{t-1}, v_t) = \frac{p(\alpha_t, v_t|Y_{t-1})}{p(v_t|Y_{t-1})},$$

using Bayes formula. Since it can be shown that both $\alpha_t|Y_{t-1}$ and $v_t|Y_{t-1}$ are normally distributed it follows that they also are jointly normally distributed and that we can apply **Lemma 1**. This yields:

$$a_{t|t} = \mathrm{E}[\alpha_t|Y_{t-1}] + \mathrm{Cov}(\alpha_t, v_t|Y_{t-1})\,\mathrm{Var}(v_t|Y_{t-1})^{-1}(v_t - \mathrm{E}[v_t|Y_{t-1}])] \quad (22)$$

$$P_{t|t} = \mathrm{Var}(\alpha_t|Y_{t-1}) - \mathrm{Cov}(\alpha_t, v_t|Y_{t-1})\,\mathrm{Var}(v_t|Y_{t-1})^{-1}\,\mathrm{Cov}(\alpha_t, v_t|Y_{t-1})^T$$

Thus, it only remains to calculate the unknown means and variances:

$$\begin{aligned}
\mathrm{E}[v_t|Y_{t-1}] &= \mathrm{E}[y_t - Z_t a_t|Y_{t-1}] \\
&= \mathrm{E}[Z_t a_t + \varepsilon_t - Z_t a_t|Y_{t-1}] = 0, \\
\mathrm{Cov}(\alpha_t, v_t|Y_{t-1}) &= \mathrm{E}[\alpha_t(Z_t\alpha_t + \varepsilon_t - Z_t a_t)^T|Y_{t-1}] \\
&= \mathrm{E}[\alpha_t(\alpha_t - a_t)^T Z_t^T|Y_{t-1}] = P_t Z_t^T, \\
\mathrm{Var}(v_t|Y_{t-1}) &= \mathrm{Var}(Z_t\alpha_t + \varepsilon_t - Z_t a_t|Y_{t-1}) = Z_t P_t Z_t^T + H_t.
\end{aligned}$$

If we set this last expression for $\mathrm{Var}(v_t|Y_{t-1})$ equal to F_t and insert the above identities back into Eq. (22) we arrive at

$$a_{t|t} = a_t + P_t Z_t^T F^{-1} v_t \quad (23)$$

$$P_{t|t} = P_t - P_t Z_t^T F_t^{-1} Z_t P_t. \quad (24)$$

To complete the induction step, observe that

$$\begin{aligned}
a_{t+1} &= \mathrm{E}[T_t\alpha_t + R_t\eta_t|Y_t] = T_t\,\mathrm{E}(\alpha_t|Y_t) = T_t a_{t|t}, \\
P_{t+1} &= \mathrm{Var}(T_t\alpha_t + R_t\eta_t|Y_t) \\
&= T_t\,\mathrm{Var}(\alpha_t|Y_t)T_t^T + R_t Q_t R_t^T = T_t P_{t|t} T_t^T + R_t Q_t R_t^T.
\end{aligned}$$

Letting $K_t = T_t P_t Z_t^T F_t^{-1}$ and substituting the above results into Eq. (23)-(24) finally yields

$$\begin{aligned}
a_{t+1} &= T_t a_{t|t} = T_t a_t + K_t v_t, \\
P_{t+1} &= T_t P_{t|t} T_t^T + R_t Q_t R_t^T = T_t P_t (T_t - K_t Z_t) + R_t Q_t R_t^T.
\end{aligned}$$

This completes the induction step and we have now derived the Kalman Recursions:

$$\begin{aligned}
v_t &= y_t - Z_t a_t, & F_t &= Z_t P_t Z_t^T + H_t \\
a_{t|t} &= a_t + P_t Z_t^T F_t^{-1} v_t, & P_{t|t} &= P_t - P_t Z_t^T F_t^{-1} Z_t P_t \\
a_{t+1} &= T_t a_t + K_t v_t, & P_{t+1} &= T_t P_t (T_t - K_t Z_t)^T + R_t Q_t R_t^T.
\end{aligned}$$

7.2 Additional Figures and Tables

Model	MSE	RMSE	MAE
ARIMA(2,1,1)	0.01203462	0.1097024	0.04393368
Local Level	0.01192288	0.1091920	0.04416067
Random walk	0.01192285	0.1091918	0.04415827

Table 3: Test error measures for the fitted Local Level, ARIMA and random walk models for values before 2014-11-11

Model	MSE	RMSE	MAE
ARIMA(0,1,0)	0.002193875	0.04683882	0.03279620
Local Level	0.002204582	0.04695298	0.03288663

Table 4: Test error measures for the fitted Local Level, ARIMA and random walk models for values after 2014-11-11

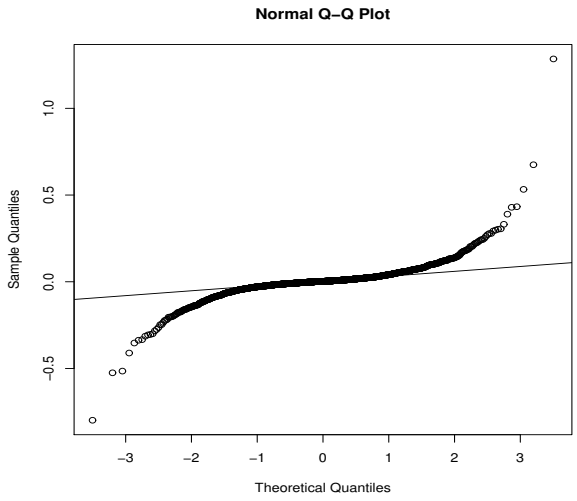


Figure 7: Residual QQ-plot from the fitted ARIMA(3,1,2) model

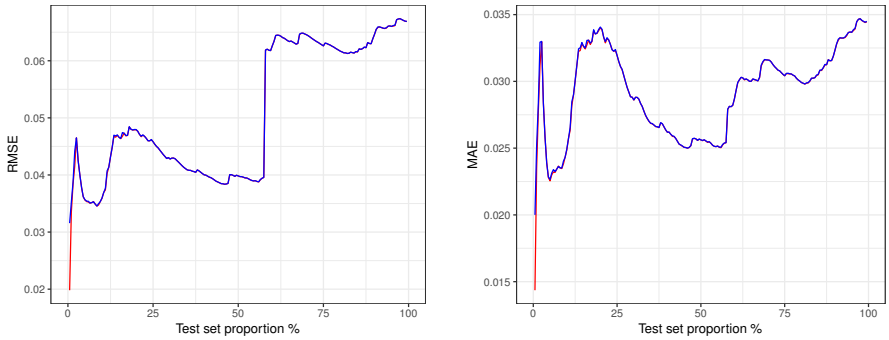


Figure 8: RMSE and MAE comparison between the Local Level model (blue) and ARIMA(0,1,0) model (red) for varying proportion sizes between the test and training set

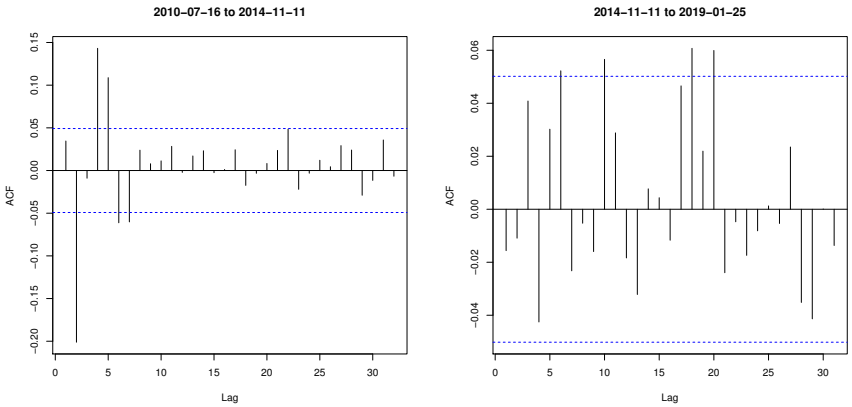


Figure 9: ACF plots of the log returns prior (left) and after (right) to 2014-11-11