# Analyzing Bitcoin Time Series Data

1. **Introduction**

This semester has really brought my attention to the cryptocurrency market, especially as it relates to time series data analysis. New investors, massive corporations, and even people who merely enjoy keeping up with the latest trends, are investing their money in various crypto currencies via different brokers and exchanges in numerous crypto markets around the globe. Putting it simply, the crypto market has been on fire and something everyone’s heard about by this point. In recent times, the altcoin[[1]](#footnote-1) market has been doing particularly well, but we’ll discuss more of that a bit later. For now, we must draw our attention to the crypto currency that started it all - **Bitcoin** **(BTC)**.

The BTC blockchain network was created on January 3, 2009 by Satoshi Nakamoto[[2]](#footnote-2) as the first crypto currency on a blockchain network. Essentially, a blockchain network is a database where data is stored in blocks and chained together. New data is entered as its own new “block” and is paired, or “chained” together with another block in chronological order. Various types of information can be stored on a blockchain but its most common use thus far is acting as a ledger for transactions. For BTC, blockchain technology is being used in a decentralized way; that is, no single entity or group has total control, rather all users retain control collectively. Bitcoin was the first player in the game to effectively use blockchain technology in a unique way, and it’s allowed for more secure transactions, is on almost every single exchange (if not every), and has revolutionized the crypto industry in multiple, unforeseen ways.

This analysis focuses on time series data associated with monthly Bitcoin prices per share from October 1, 2014 to May 1, 2021. The data was obtained from *Yahoo Finance* and can be accessed via the following link: [BTC](https://finance.yahoo.com/quote/BTC-USD/history?period1=1412121600&period2=1619827200&interval=1mo&filter=history&frequency=1mo&includeAdjustedClose=true). Various time series methods that were discussed throughout the course of the semester will be utilized for this data analysis, competing models will be built and assessed on performance, and overall conclusions about BTC will be drawn.

1. **Methods**

First, we can take look at the realization, sample autocorrelation, and Parzen spectral density plots to try and analyze the data and describe the behavior. [Figure 1](#_Figure_1) is demonstrated below and represents the realization for BTC.

# Figure 1

Figure 1: BTC Realization of Price per Share vs. Time in Months

Chart, line chart, histogram

Description automatically generated

The realization above shows time in months on the x-axis and price per share of BTC on the y-axis, and as we can also see, it looks like BTC has more recently gone way up in value (i.e., it went to the moon). We can see that at certain time points of the data there are periods with drastic increases or decreases in Bitcoin prices per share. A large increase in price seems to occur between August 1, 2017 and December 1, 2017, and an even larger increase again between May 1, 2020 and May 1, 2021. The realization also seems to show that with each major increase there seems to be a steadier decrease, but also periods where prices seem to stabilize at new averages. Next, [Figure 2](#_Figure_2) below portrays the realization, sample autocorrelation, and spectral density plots for this data.

# Figure 2

Figure 2: BTC Realization, Sample Autocorrelation, and Spectral Density Plots

Chart, histogram

Description automatically generated

The figure above shows a sample autocorrelations plot with a rather steadily decreasing relationship between observations at time and time for values of larger than , and a little bit of sinusoidal behavior at the end. For instance, the correlation between times and is still positive and remains somewhat strong. Also, the spectral density plot appears to show a frequency peak at , and a much smaller frequency peak seems to be forming at about . The plot as a whole appears to gradually flatten out. Next, to see about stationarity, we must examine the factor tables before and after fitting a model. [Table 1](#_Table_1) below demonstrates a factor table for this data.

# Table 1

Table 1: Factor Table for BTC Time Series Data

|  |  |  |  |
| --- | --- | --- | --- |
| **Factor** | **Roots** | **Abs Recip.** | **Freq.** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

We can see from the factor table above that the model is stationary since the absolute reciprocal root values for its complex roots are less than one and the absolute value for its real root is greater than one. The realization demonstrates behavior that’s associated with the root that’s the closest to the unit circle, and the spectral density plot also shows a frequency at or near since the strongest root’s rather close to the unit circle.

Next, since the data seems to show both upward and downward linearly trending behavior, but especially upward trending behavior, the Cochrane-Orcutt and Woodward-Bottone-Gray tests were utilized to assess trends. We obtain p-values of for CO and for WBG, which means that both tests correctly identified underlying trends for the data. Thus, competing models were fit after evaluating the above plots, factor tables, and behavioral trends of the data, and results regarding these model fitting procedures are portrayed in the next section of this analysis.

1. **Results**

An ARMA model was first built and examined using the aic5.wge function to identify the top models for this data (using AIC), allowing to go from 0 to 12 and to go from 0 to 4. The factor table for the “best” model from the competing models can be seen below in [Table 2](#_Table_2). Technically, the function chose an ARMA(9,4) model to be the best in terms of AIC criteria, however the factor table for this model was a bit off since it showed two frequency peaks at , so I decided to stick with the second “best” model, which was the ARMA(3,2).

# Table 2

Table 2: Factor Table for ARMA(3,2)

|  |  |  |  |
| --- | --- | --- | --- |
| **Factor** | **Roots** | **Abs Recip.** | **Freq.** |
|  |  |  |  |
|  |  |  |  |

Next, we needed to check the whiteness of the residuals by inspecting the back cast residuals from the fitted model. [Figure 3](#_Figure_3) below demonstrates the plotted residuals, the residual sample autocorrelations, and the residual Parzen spectral density plot for the obtained model.

# Figure 3

Figure 3: Residual Plot, Autocorrelations, and Spectral Density for ARMA(3,2)

A picture containing chart

Description automatically generated

From the figure above we can see that the residuals don’t appear to be so random as there seems to be some pseudo-periodic behavior, but the sample autocorrelations stay within the 95% limit lines (except for one). The Ljung-Box test results using the ljung.wge function for K=24 and K=48 were also obtained. The outputs for both values of K can be seen in [Table 3](#_Table_3) and [Table 4](#_Table_4) below. As we can see, for both K=24 and K=48, the p-values are considerably greater than α=.05, so we don’t have evidence to reject the null hypothesis of white noise. Also, based on the plots in the figure and the conducted Ljung-Box test, it appears that the fitted model does a rather decent job whitening the residuals.

# Table 3

Table 3: Ljung-Box Test Result for ARMA(3,2)

|  |  |
| --- | --- |
| K | 24 |
| Chi Sqr. | 18.35721 |
| DF | 19 |
| P Value | 0.4987081 |

# Table 4

Table 4: Ljung-Box Test Result for ARMA(3,2)

|  |  |
| --- | --- |
| K | 48 |
| Chi Sqr. | 38.68585 |
| DF | 43 |
| P Value | 0.6587886 |

The model’s forecasting performance was also analyzed, and based on the forecasting performance plot represented in [Figure 4](#_Figure_4) below, and the model doesn’t seem to do an appropriate job forecasting since the last 10 steps aren’t forecasted quite well. Lastly, the obtained RMSE and MAD values are *10,010.86* and *9,133.65* for model forecasting, respectively.

# Figure 4

Figure 4: Forecasting Performance for ARMA(3,2)Chart

Description automatically generated

To obtain better models, transforming the data is another viable option. The factor table in the *Methods* section of this analysis was created using the overfitting procedure with , and from the table we can conclude that the unit root of can be included. The data was then transformed by first applying a seasonal transformation, and the transformed data was also differenced right after in order to fit another ARMA model by letting AIC select the top five model orders for the transformed data (p=0:15, q=0:2). AIC selected an ARMA(3,0) model to be the “top” model and so we fit it using “Burg” estimates to obtain a final model of: , where and , which is incredibly high. The factor table for this model can be seen in [Table 5](#_Table_5) below.

# Table 5

Table 5: Factor Table for ARMA(3,0)

|  |  |  |  |
| --- | --- | --- | --- |
| **Factor** | **Roots** | **Abs Recip.** | **Freq.** |
|  |  |  |  |
|  |  |  |  |

Again, we needed to check the whiteness of the residuals by inspecting the back cast residuals from the fitted model. [Figure 5](#_Figure_5) below demonstrates the plotted residuals, the residual sample autocorrelations, and the residual Parzen spectral density plot for the ARMA(3,0) model.

# Figure 5

Figure 5: Residual Plot, Autocorrelations, and Spectral Density for ARMA(3,0)

Chart, histogram

Description automatically generated with medium confidence

From [Figure 5](#_Figure_5) we can see that the residuals for this model appear to be more random than the previous model’s, and all the sample autocorrelations stay within the 95% limit lines this time. The Ljung-Box test results using the ljung.wge function for K=24 and K=48 were also obtained, and the outputs for both values of K can be seen in [Table 6](#_Table_6_1) and [Table 7](#_Table_7) below. As we can see, for both K=24 and K=48, the p-values are considerably greater than α=.05, so we don’t have evidence to reject the null hypothesis of white noise. Also, based on the plots in the figure and the conducted Ljung-Box test, it appears that the fitted model does a better job whitening the residuals.

# Table 6

Table 6: Ljung-Box Test Result for ARMA(3,0)

|  |  |
| --- | --- |
| K | 24 |
| Chi Sqr. | 11.75669 |
| DF | 21 |
| P Value | 0.9459546 |

# Table 7

Table 7: Ljung-Box Test Result for ARMA(3,0)

|  |  |
| --- | --- |
| K | 48 |
| Chi Sqr. | 29.02839 |
| DF | 45 |
| P Value | 0.9689315 |

The model’s forecasting performance was also analyzed and based on the forecasting performance plot represented in [Figure 6](#_Figure_6) below, and the model doesn’t seem to do an appropriate job forecasting since the last 10 steps aren’t forecasted quite well, but it does do a somewhat better job than the previous ARMA(3,2) model as the obtained RMSE and MAD values were slightly better at *7,132.22* and *7,115.18*, respectively.

# Figure 6

Figure 6: Forecasting Performance for ARMA(3,0)

Chart

Description automatically generated

1. **Conclusion**

ARMA(3,2) and ARMA(3,0) models were both fit to analyze Bitcoin time series data. Between the competing models, the ARMA(3,0) with the included unit root of performs better on the transformed and differenced data. The residuals for the ARMA(3,0) model appear to be more random than the ARMA(3,2) model’s residuals, and all the sample autocorrelations stay within the 95% limit lines as well. For both models, we don’t have evidence to reject the null hypothesis of white noise, but the ARMA(3,0) does a better job whitening the residuals. Lastly, although both models didn’t show great overall forecasting performances, the ARMA(3,0) model does slightly better, especially when comparing RMSE and MAD values.

Real world time series data can be very difficult to analyze, however practical and industry-changing conclusions can be made if done so correctly. For Bitcoin data, the sudden and massive price increases have been unprecedented, especially within the last year or so. However, before the most recent sudden price increases, the COVID pandemic had a drastic impact on Bitcoin prices and financial markets in general, and we can see that this is greatly evident for BTC within the data as well. Throughout its cycle, Bitcoin seems to have gone through “stagnant” phases where prices per share don’t change too much on average, however, massive and sudden price increases also seem to occur every few years or so. This can be attributed to various geopolitical and other economic factors, but one significant impact on BTC pricing has been a process called "Bitcoin halving." Essentially, new “Bitcoins” are introduced to the market by “Bitcoin mining,” which is a process that's done by verifying Bitcoin blocks or transactions, and miners can obtain 6.25 BTC as a reward by verifying a single block of transactions and adding it to the BTC network. The amount miners get paid is reduced by half approximately every four years, or after every 210,000 blocks are mined, which in turn causes prices to skyrocket as supply decreases and demand increases. This is also evident in the data where we see these massive jumps in prices followed by some sort of longing correction[[3]](#footnote-3) afterward.

Lastly, there were a few limitations to my analysis. For one, I was only able to obtain data from 2014 and after, so it would’ve been more impactful and helpful to have utilized BTC data from its creation until now. Also, instead of using monthly data I could have used weekly, or even daily data, but I was having issues downloading anything that wasn’t monthly data and so I had to persist with using it for this analysis, however having more data would have been more beneficial for revealing clearer and more insightful patterns. Moreover, the obtained models aren’t “perfect” by any means, but the results are practical and demonstrate what can be obtained from real-world time series data analysis, nonetheless. It would have also been interesting to incorporate more time series analysis methods for other cryptocurrency data, such as the altcoins Ethereum, Dogecoin, Cardano, etc., since trends can be quite different for various coins and it would have been intuitive to see how such information relates with BTC.

What we’re currently witnessing with BTC has never been seen before. BTC is currently on its greatest “bull run” amid a pandemic, and “halving” hasn’t even occurred yet (it’s expected to happen this year). It’s difficult to say what can happen to Bitcoin in the short-term. Will halving cause prices to skyrocket again? Is there another massive correction bound to happen before then? Maybe after? It may be difficult to predict the future, however overall data seems to show that Bitcoin may be here to stay for a while.

1. **Appendix (R Code)**

### Project ###

#Load packages

library(tswge)

library(zoo)

library(xts)

library(lubridate)

library(dplyr)

library(tidyr)

library(WDI)

library(keras)

library(anomalize)

library(devtools)

library(signal)

library(MASS)

library(dplyr)

library(tidyverse)

#Read in monthly data (August 2010 - May 2021)

btc.monthly = read.csv("BTC-USD.csv")

View(btc.monthly)

str(btc.monthly)

# Methods #

#Plot monthly data

realization = plotts.wge(ts((btc.monthly$Open),start = c(btc.monthly$Date == "2014-10-01"),frequency = 1))

#Realizations, Sample Autocorrelations, and Parzen Spectral Densities

plotts.sample.wge(btc.monthly$Open)

#Examine factor table

est.ar.wge(btc.monthly$Open,p=15,type='burg')

#Cochrane-Orcutt Tests for trends

co.wge(btc.monthly$Open) #p-value=2.197806e-05

#Woodward-Bottone-Gray test for trend

wbg.boot.wge(btc.monthly$Open) #p-value=0.01754386

# Modeling Results #

#Model 1

#Identify top candidate models for data

aic5.wge(btc.monthly$Open,p=0:12,q=0:4,type='aic') #ARMA(9,4), ARMA(3,2), ARMA(6,3)

#Factor table for top models

model1 = est.arma.wge(btc.monthly$Open,p=9,q=4,factor=TRUE) #ARMA(9,4)

model2 = est.arma.wge(btc.monthly$Open,p=3,q=2,factor=TRUE) #ARMA(3,2)

model3 = est.arma.wge(btc.monthly$Open,p=6,q=3,factor=TRUE) #ARMA(6,3)

#Chosen = ARMA(3,2)

#Check 'whiteness' of residuals

model2$res

#Plot residuals

plotts.sample.wge(model2$res,lag.max=48,arlimits=TRUE)

#Ljung-Box test results for K=24 and K=48

ljung.wge(model2$res,p=3,q=2) #K=24 is the default

ljung.wge(model2$res,p=3,q=2,K=48)

#K=24: p-value is 0.4987081

#K=48:p-value is 0.6587886

#Forecasting performance

model2.forecast=fore.arma.wge(btc.monthly$Open,phi=model2$phi,theta=model2$theta,n.ahead=10,lastn=TRUE,limits=FALSE)

model2.forecast$f

#RMSE

sqrt(mean((model2.forecast$se)^2)) #10010.86

#MAD

mean(abs(model2.forecast$se)) #9133.65

#Model doesn't perform too well!

#Model 2

#Fit an ARIMA model with (1-B^12) roots

d\_btc1=artrans.wge(btc.monthly$Open,phi.tr=c(rep(0,12)),plot=TRUE)

d\_btc2=artrans.wge(d\_btc1,phi.tr=1)

aic.wge(d\_btc2,p=0:10,q=0:2)

btc.model = est.arma.wge(d\_btc2,p=3,q=0)

#phi = 0.27586863, 0.07772535, 0.38366067

#var = 6021575

#Check 'whiteness' of residuals

btc.model$res

#Plot of residuals

plotts.sample.wge(btc.model$res,lag.max=48,arlimits=TRUE)

#Ljung-Box test results for K=24 and K=48

ljung.wge(btc.model$res,p=3,q=0) #K=24 is the default

ljung.wge(btc.model$res,p=3,q=0,K=48)

#K=24: p-value is 0.9459546

#K=48:p-value is 0.9689315

#Forecasting performance

btc.model.forecast=fore.arma.wge(btc.monthly$Open,phi=btc.model$phi,theta=btc.model$theta,n.ahead=10,lastn=TRUE,limits=FALSE)

btc.model.forecast$f

#RMSE

sqrt(mean((btc.model.forecast$se)^2)) #7132.217

#MAD

mean(abs(btc.model.forecast$se)) #7115.181

1. Crypto currencies other than Bitcoin (BTC), or a currency that distinguishes itself by providing new or additional capabilities, such as smart contracts. [↑](#footnote-ref-1)
2. Presumed to be a pseudonym for the person, or people, who created the original bitcoin network [↑](#footnote-ref-2)
3. A moderate decline in an asset's value [↑](#footnote-ref-3)