Modelling Cryptocurrency exchange rates using ARMA Models

Applied Macroeconometrics Assignment I

Rishabh Patil | 2021A7PS0464H

Contents

```r
knitr::opts_chunk$set(tidy.opts = list(width.cutoff = 60), tidy = TRUE)
```

```r
# install.packages('quantmod')
```

```r
library(quantmod)
```

```
## Loading required package: xts

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: TTR

## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

Description of Dataset

For our study, we'd be analyzing the exchange rates of the cryptocurrency Bitcoin, more precisely the weekly returns.

The variable of concern here is the price(in USD) of Bitcoin (returns on it).

Since there is a general notion of cryptocurrencies having a 4 year cycle, we'd be looking at 8-10 year data. And to make our analysis feasible, instead of a daily chart, we would be analyzing a weekly rolling average for our time period of 10 years.

The data

```r
btc_df <- getSymbols("BTC-USD", src = "yahoo", from = Sys.Date() -
    365 * 10, to = Sys.Date(), auto.assign = FALSE)
print(head(btc_df))
```

```
##            BTC-USD.Open BTC-USD.High BTC-USD.Low BTC-USD.Close BTC-USD.Volume
## 2014-09-17      465.864      468.174     452.422       457.334       21056800
## 2014-09-18      456.860      456.860     413.104       424.440       34483200
## 2014-09-19      424.103      427.835     384.532       394.796       37919700
## 2014-09-20      394.673      423.296     389.883       408.904       36863600
## 2014-09-21      408.085      412.426     393.181       398.821       26580100
## 2014-09-22      399.100      406.916     397.130       402.152       24127600
##            BTC-USD.Adjusted
## 2014-09-17          457.334
## 2014-09-18          424.440
## 2014-09-19          394.796
## 2014-09-20          408.904
## 2014-09-21          398.821
## 2014-09-22          402.152
```

Candlestick Chart (daily pricing)

```r
chartSeries(btc_df, name = "BTC-USD", subset = "last 120 months")
```

```
## Warning in last.xts(structure(c(465.864013671875, 456.859985351562,
## 424.102996826172, : requested length is greater than original
```

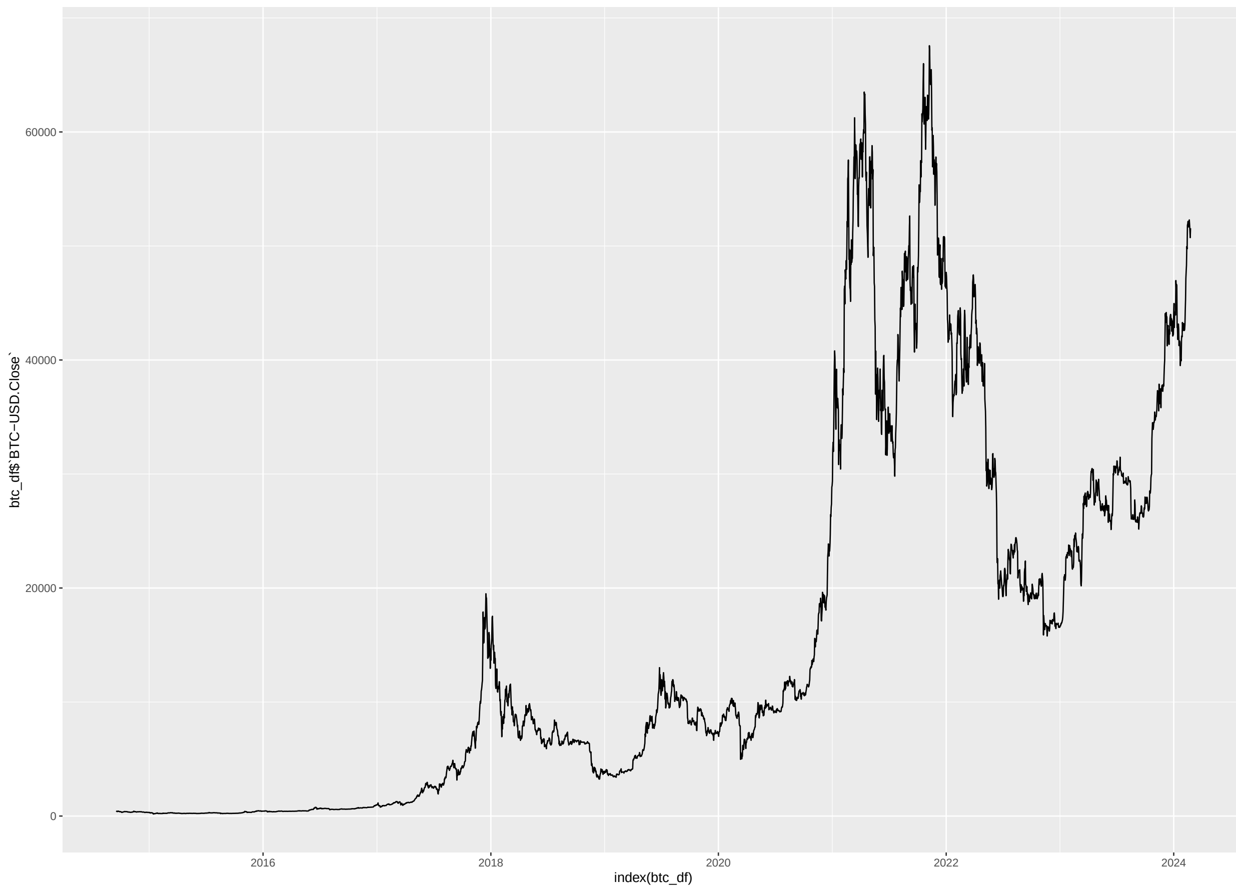
Line Graph(daily pricing)

```r
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.2.3
```

```r
ggplot(data = btc_df, aes(y = btc_df$`BTC-USD.Close`, x = index(btc_df)),
    group = 1) + geom_line()
```

```
## Don't know how to automatically pick scale for object of type <xts/zoo>.
## Defaulting to continuous.
```



Clustering Month-wise avg price:

```
# install.packages('xts')
```

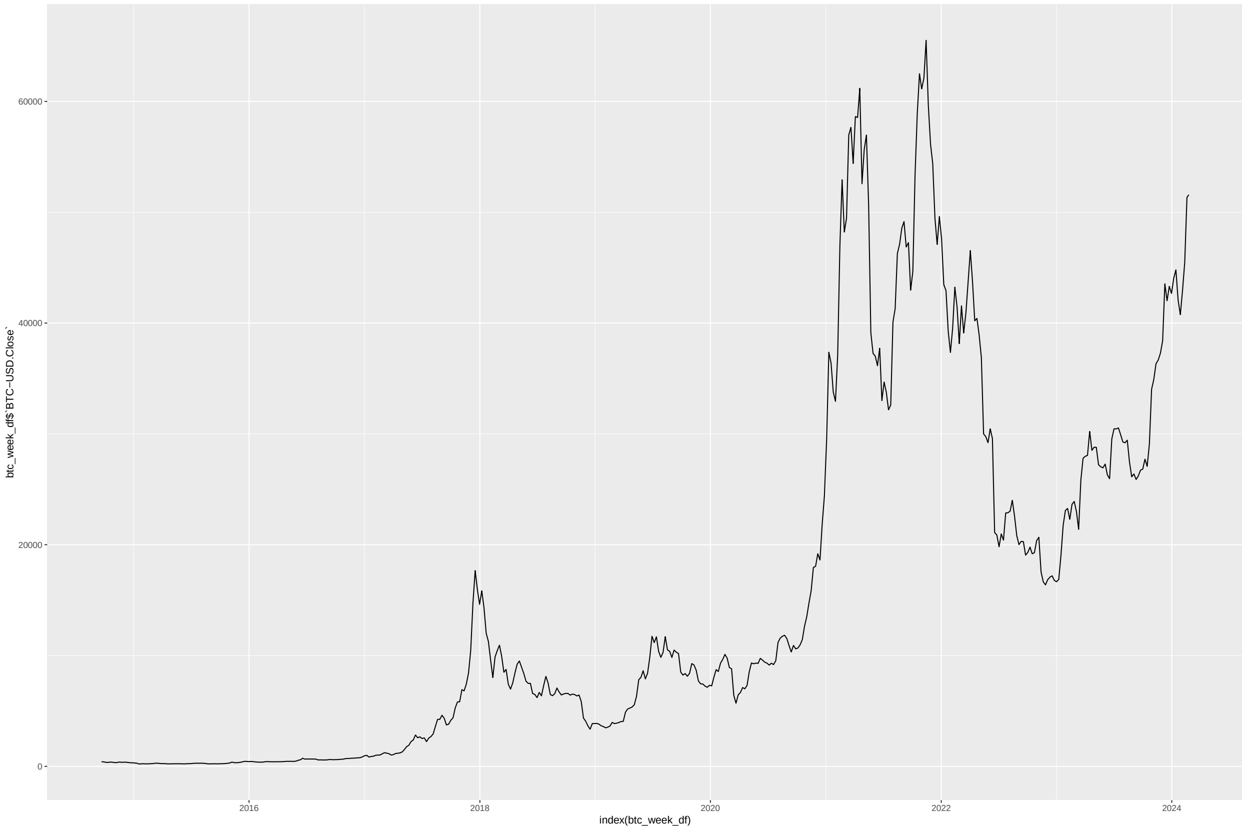
```
library(xts)
btc_week_df <- apply.weekly(btc_df, FUN = mean)
head(btc_week_df)
```

```
##            BTC-USD.Open BTC-USD.High BTC-USD.Low BTC-USD.Close BTC-USD.Volume
## 2014-09-21     429.9170     437.7182    406.6244      416.8590       31380680
## 2014-09-28     410.6507     418.6690    399.3771      407.6926       26681800
## 2014-10-05     369.7743     376.7210    353.2071      361.4266       39522557
## 2014-10-12     346.9274     363.3089    337.5679      355.2346       48736115
## 2014-10-19     389.0103     397.7904    380.4106      390.4799       22414581
## 2014-10-26     372.2030     377.1116    362.5564      367.3164       16241686
##            BTC-USD.Adjusted
## 2014-09-21         416.8590
## 2014-09-28         407.6926
## 2014-10-05         361.4266
## 2014-10-12         355.2346
## 2014-10-19         390.4799
## 2014-10-26         367.3164
```

Line Graph

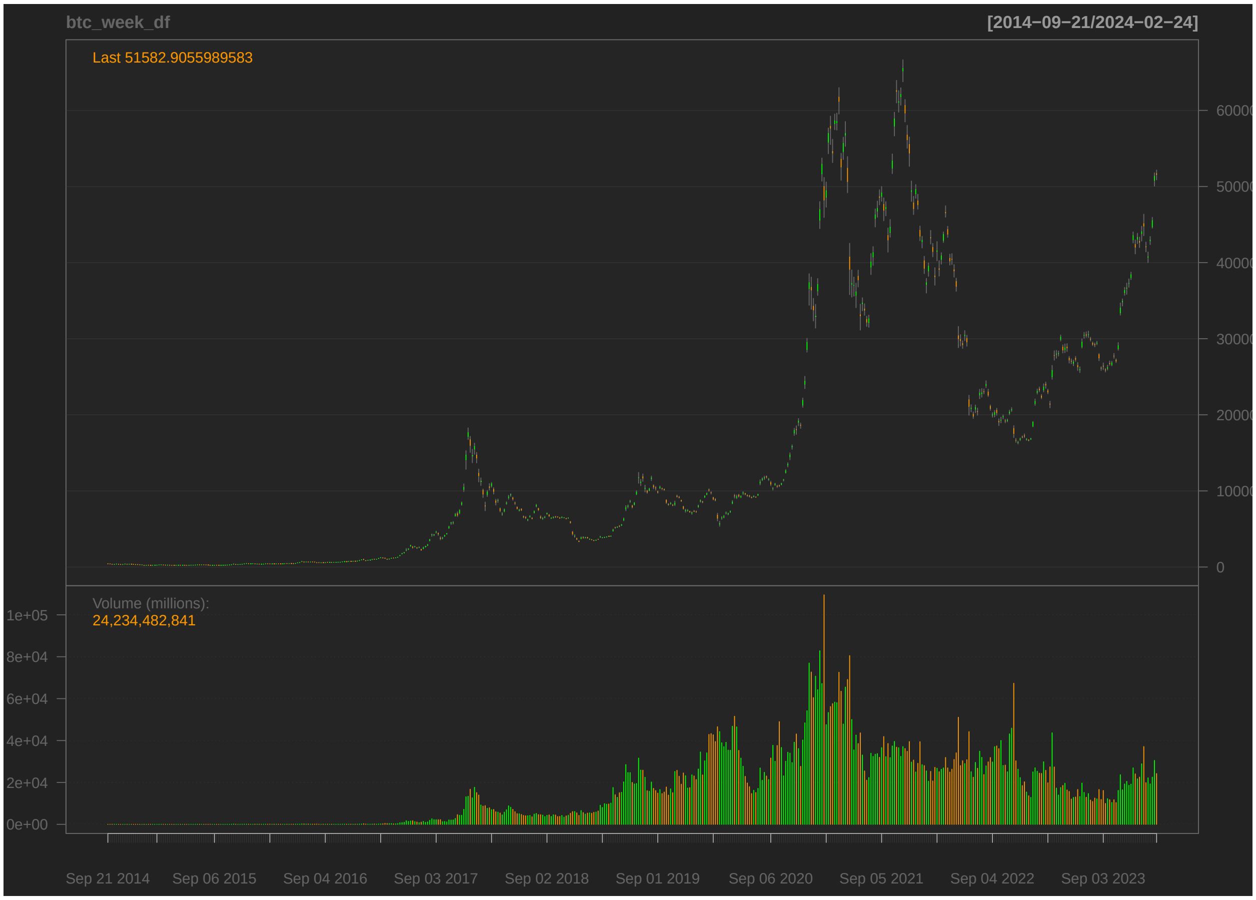
```
ggplot(data = btc_week_df, aes(y = btc_week_df$`BTC-USD.Close`,
    x = index(btc_week_df)), group = 1) + geom_line()
```

```
## Don't know how to automatically pick scale for object of type <xts/zoo>.
## Defaulting to continuous.
```



Candlestick Graph

```
chartSeries(btc_week_df)
```



Returns

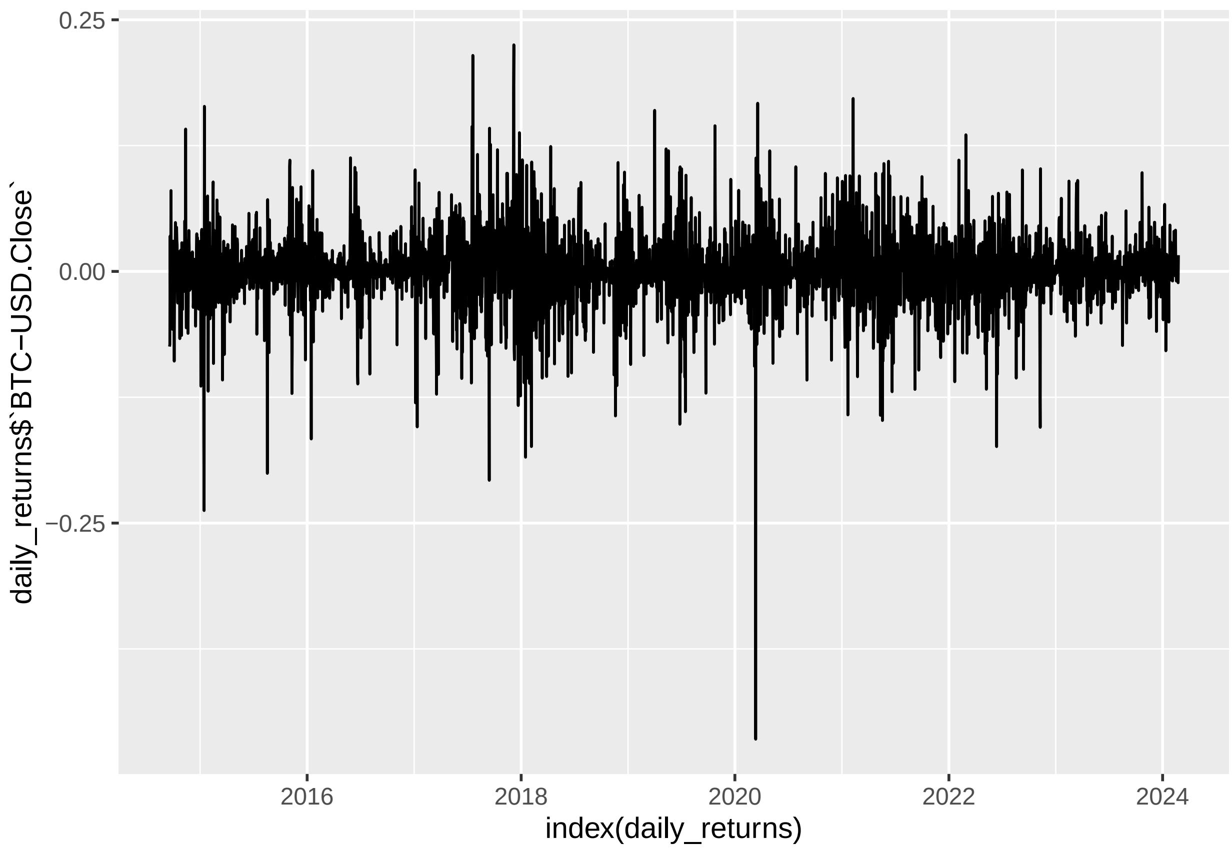
For plotting the returns, we use this formula:

$$R = \frac{S(t+1) - S(t)}{S(t)} \text{ or plotting the log return: } RLt = ln\left(\frac{S(t+1)}{S(t)}\right)$$

```
daily_returns <- log(btc_df/lag(btc_df, 1))
ggplot(data = daily_returns, aes(y = daily_returns$`BTC-USD.Close`,
    x = index(daily_returns)), group = 1) + geom_line()
```

```
## Don't know how to automatically pick scale for object of type <xts/zoo>.
## Defaulting to continuous.
```

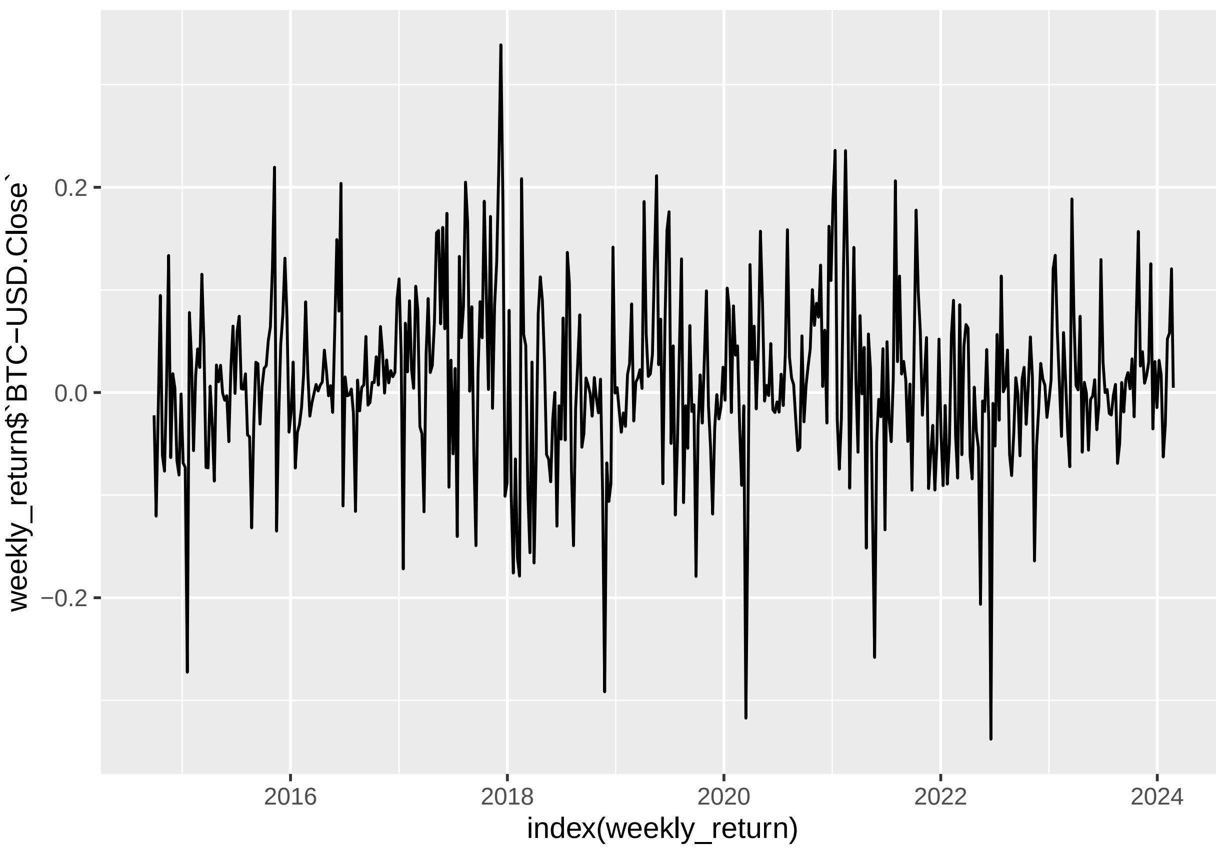
```
## Warning: Removed 1 row containing missing values (`geom_line()`).
```




```r
weekly_return <- log(btc_week_df$`BTC-USD.Close`/lag(btc_week_df$`BTC-USD.Close`,
    1))
ggplot(data = weekly_return, aes(y = weekly_return$`BTC-USD.Close`,
    x = index(weekly_return)), group = 1) + geom_line()
```

```
## Don't know how to automatically pick scale for object of type <xts/zoo>.
## Defaulting to continuous.
```

```
## Warning: Removed 1 row containing missing values (`geom_line()`).
```



This is the basis of our analysis

Visual Interpretation

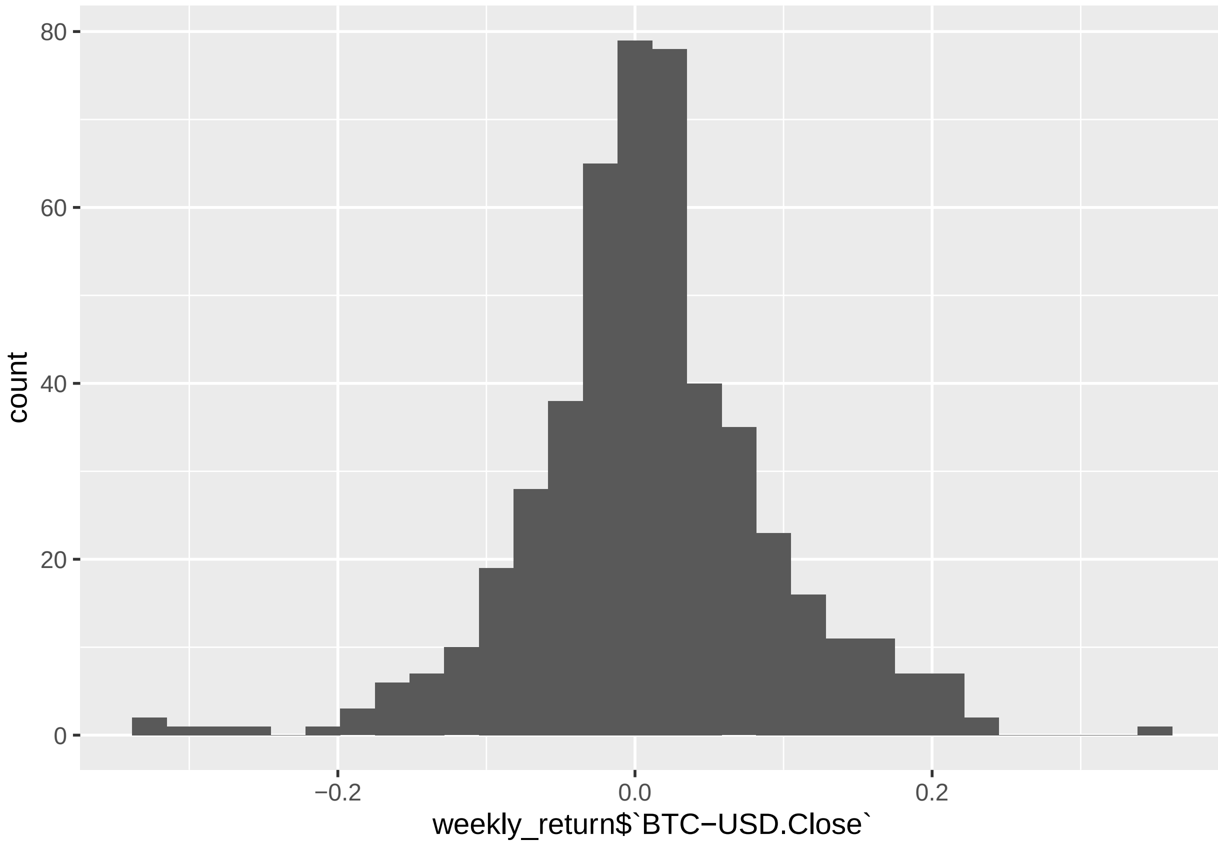
From the above log returns chart we can see that the long run expectation for weekly return is 0

$$\mathbb{E}[y_t] = 0$$

```
ggplot(weekly_return, aes(weekly_return$`BTC-USD.Close`)) + geom_histogram()
```

```
## Don't know how to automatically pick scale for object of type <xts/zoo>.
## Defaulting to continuous.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

```
## Warning: Removed 1 rows containing non-finite values (`stat_bin()`).
```

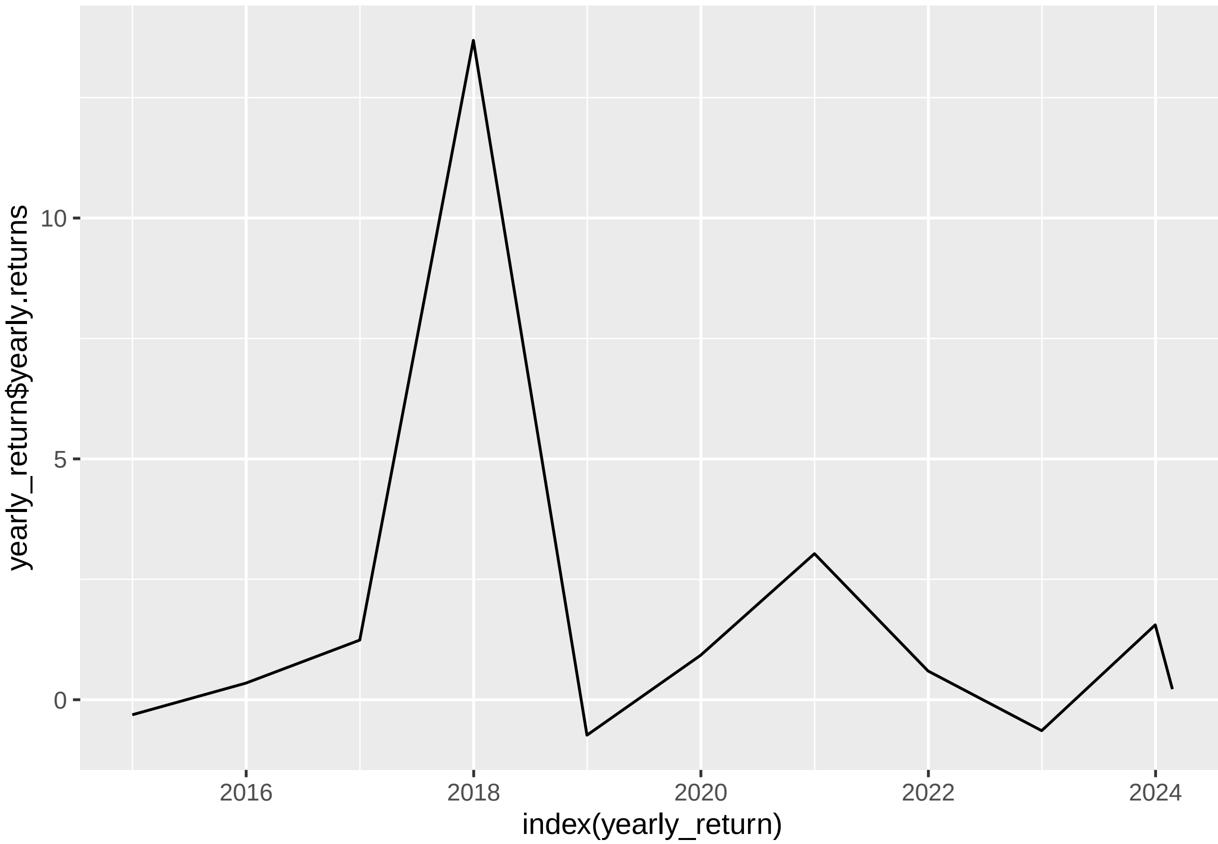


The data looks normally distributed with slight outliers.

looking for seasonality:

```
yearly_return <- yearlyReturn(btc_df)
ggplot(data = yearly_return, aes(y = yearly_return$yearly.returns,
    x = index(yearly_return)), group = 1) + geom_line()
```

```
## Don't know how to automatically pick scale for object of type <xts/zoo>.
## Defaulting to continuous.
```



We do see that there is a 4-year cycle apparent from the yearly returns, that is, the returns rise up for a year, then reach peak very steeply the next and a steep decline sets in ending the cycle.

But apart from the yearly return analysis the weekly returns appear stationery.

The objective of this study is to model the weekly returns with appropriate ARMA model(s) and predict the trend (forecast it).

Estimation and Holdback dataset

Estimation: from 2014-09-21 to 2022-09-21 - *train data*

Holdback: from 2022-09-28 to 2024-02-23 - *test data*

(~75/25 split) (split post cycle)

```
btc_week_est <- btc_week_df[index(btc_week_df) >= as.Date("2014-09-21") &
    index(btc_week_df) <= as.Date("2022-09-21"), ]

btc_week_hb <- btc_week_df[index(btc_week_df) >= as.Date("2022-09-22") &
    index(btc_week_df) <= as.Date("2024-02-23"), ]
```

```
chartSeries(btc_week_est)
```

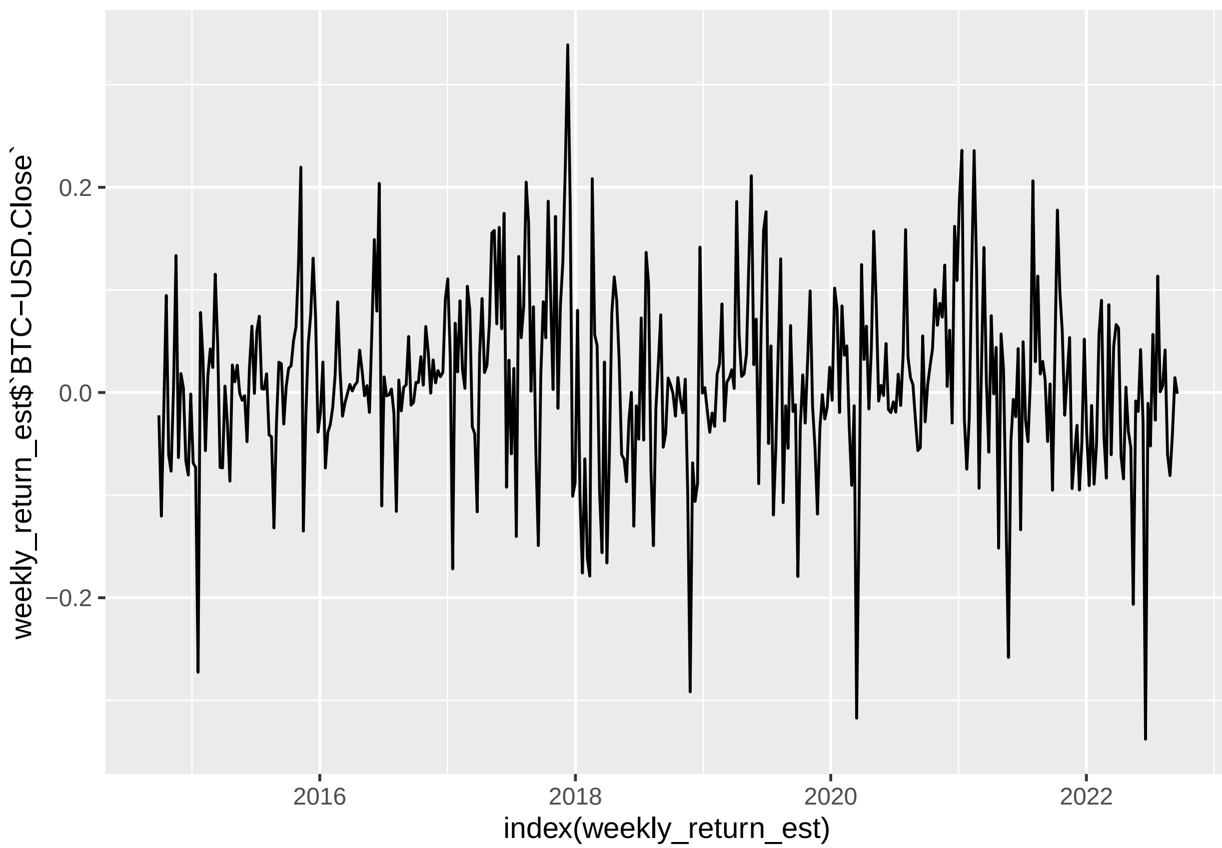

```r
chartSeries(btc_week_hb)
```


```r
weekly_return_est <- weekly_return[index(weekly_return) >= as.Date("2014-09-21") &
    index(weekly_return) <= as.Date("2022-09-21"), ]
weekly_return_hb <- weekly_return[index(weekly_return) >= as.Date("2022-09-22") &
    index(weekly_return) <= as.Date("2024-02-23"), ]
```

```r
ggplot(data = weekly_return_est, aes(y = weekly_return_est$`BTC-USD.Close`,
    x = index(weekly_return_est)), group = 1) + geom_line()
```

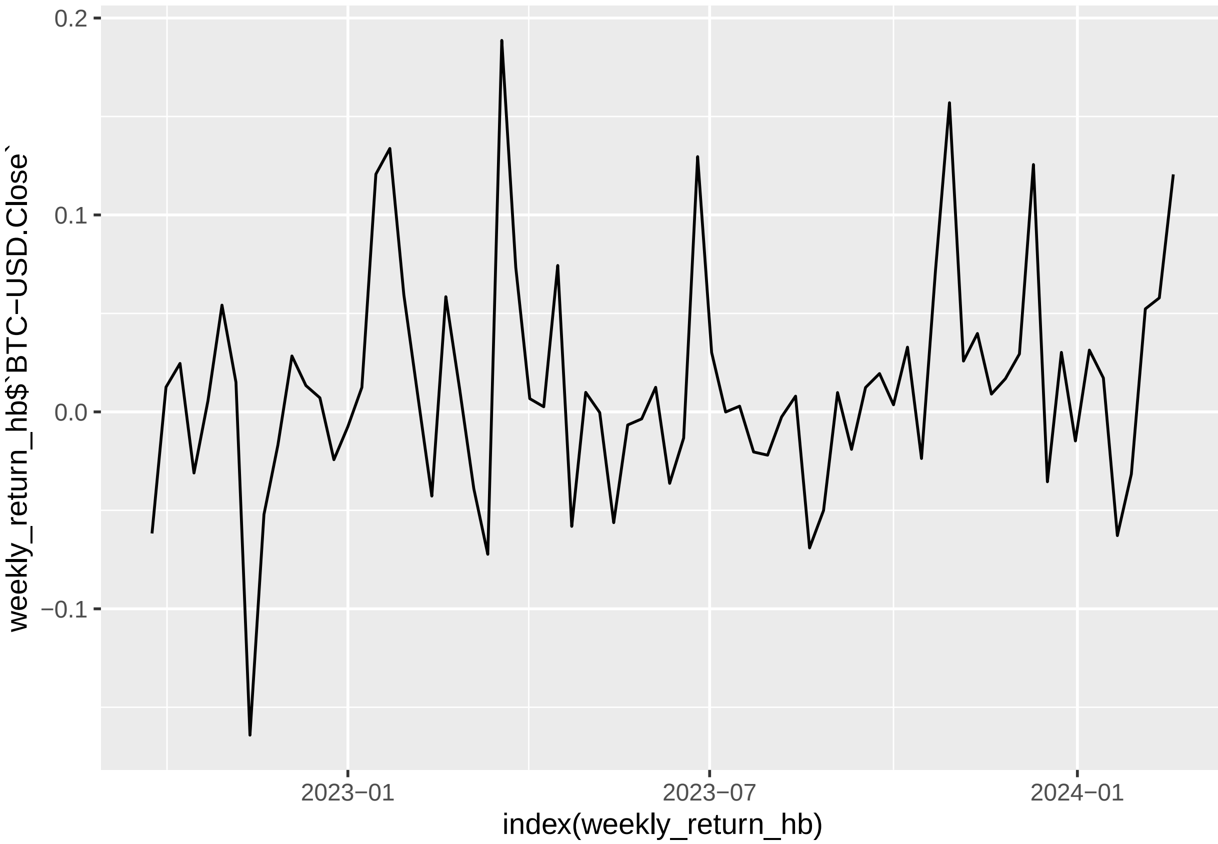
```
## Don't know how to automatically pick scale for object of type <xts/zoo>.
## Defaulting to continuous.

## Warning: Removed 1 row containing missing values (`geom_line()`).
```




```r
ggplot(data = weekly_return_hb, aes(y = weekly_return_hb$`BTC-USD.Close`,
    x = index(weekly_return_hb)), group = 1) + geom_line()
```

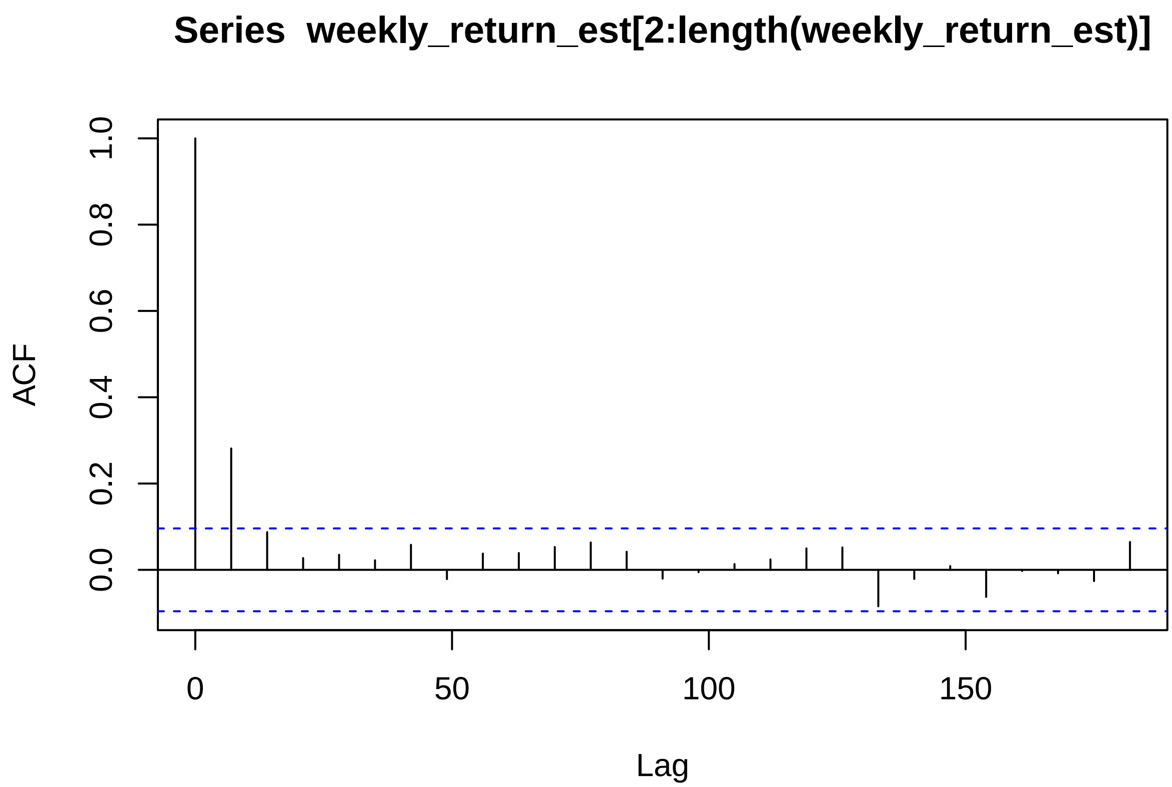
```
## Don't know how to automatically pick scale for object of type <xts/zoo>.
## Defaulting to continuous.
```



Model fitting and model selection

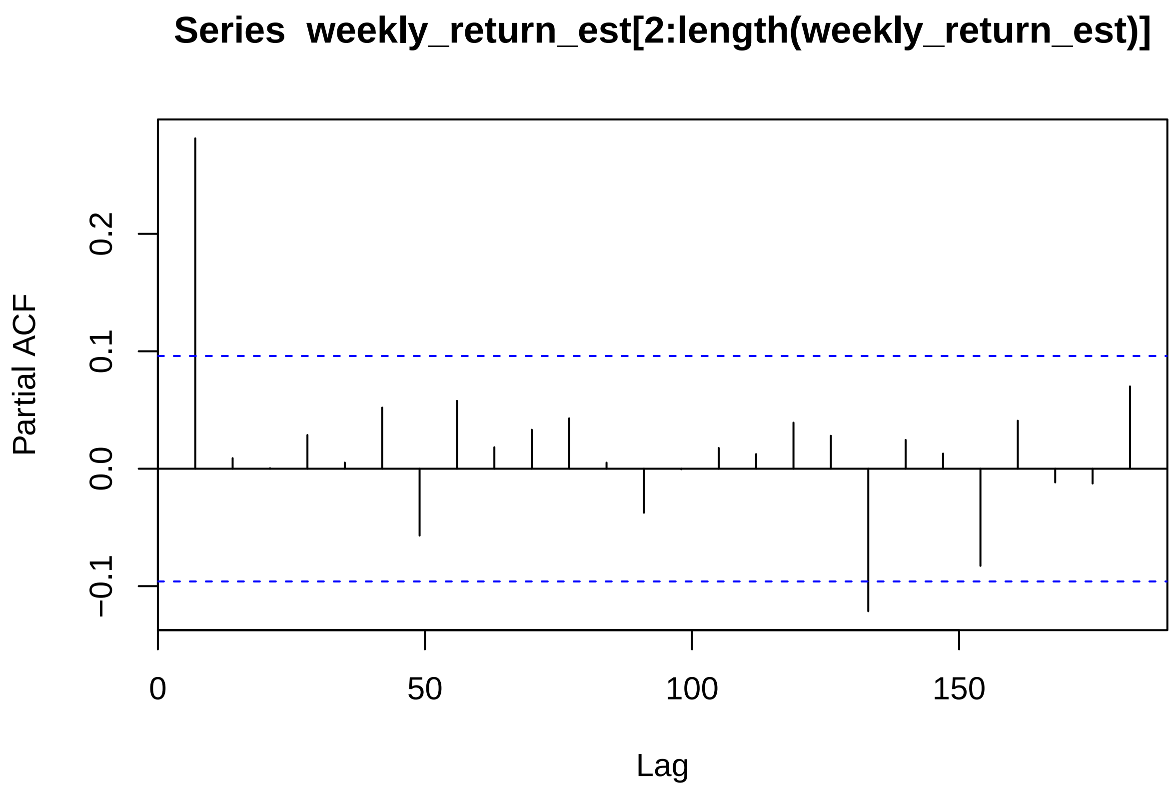
Analyzing ACF and PACF

```
acf(weekly_return_est[2:length(weekly_return_est)])
```



We see that the *ACF is smoothly decaying* which means its more *likely to be an AR process*

```
pacf(weekly_return_est[2:length(weekly_return_est)])
```



There is no continuous decay in PACF, the value abruptly falls after first lag. so we can assume it to be a AR process. **Note:** *The 19th lag is also significant, suggesting some seasonality. But we will ignore that for now.*

Since one lag is significant, AR(1) model can be tested. We will also test ARMA(1,1) and ARMA(1,2) and ARMA(2,1)

Testing ARMA models

AR(1)

```
ar1 <- arima(weekly_return_est$`BTC-USD.Close`, order = c(1,
    0, 0))
summary(ar1)
```

```
##           Length Class  Mode
## coef        2    -none- numeric
## sigma2      1    -none- numeric
## var.coef    4    -none- numeric
## mask        2    -none- logical
## loglik      1    -none- numeric
## aic         1    -none- numeric
## arma        7    -none- numeric
## residuals 418    ts     numeric
```

```
## call        3    -none- call
## series      1    -none- character
## code        1    -none- numeric
## n.cond      1    -none- numeric
## nobs        1    -none- numeric
## model      10    -none- list
```

Coefficients Test The coefficients have some standard error but can be considered insignificant. Statistically testing the significance:

```r
library(lmtest)
```

```
## Warning: package 'lmtest' was built under R version 4.2.3
```

```r
coeftest(ar1)
```

```
##
## z test of coefficients:
##
##            Estimate Std. Error z value Pr(>|z|)
## ar1       0.2807682  0.0469217  5.9838 2.18e-09 ***
## intercept 0.0092963  0.0056354  1.6496  0.09902 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

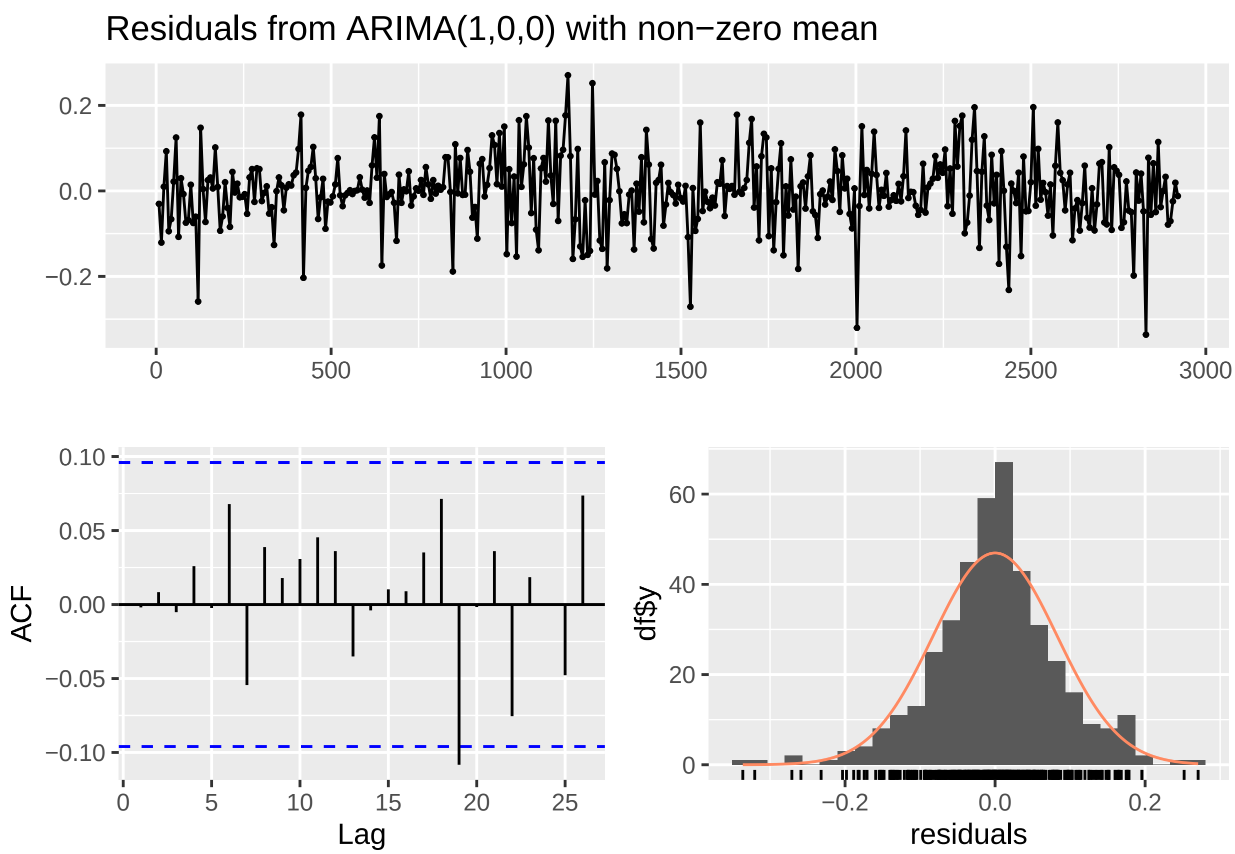
The coefficients are statistically significant. (The Intercept has a lower significance threshold, but can be fitted in for the model.)

```r
library("forecast")
```

Residual Check

```
## Warning: package 'forecast' was built under R version 4.2.3
```

```r
checkresiduals(ar1)
```



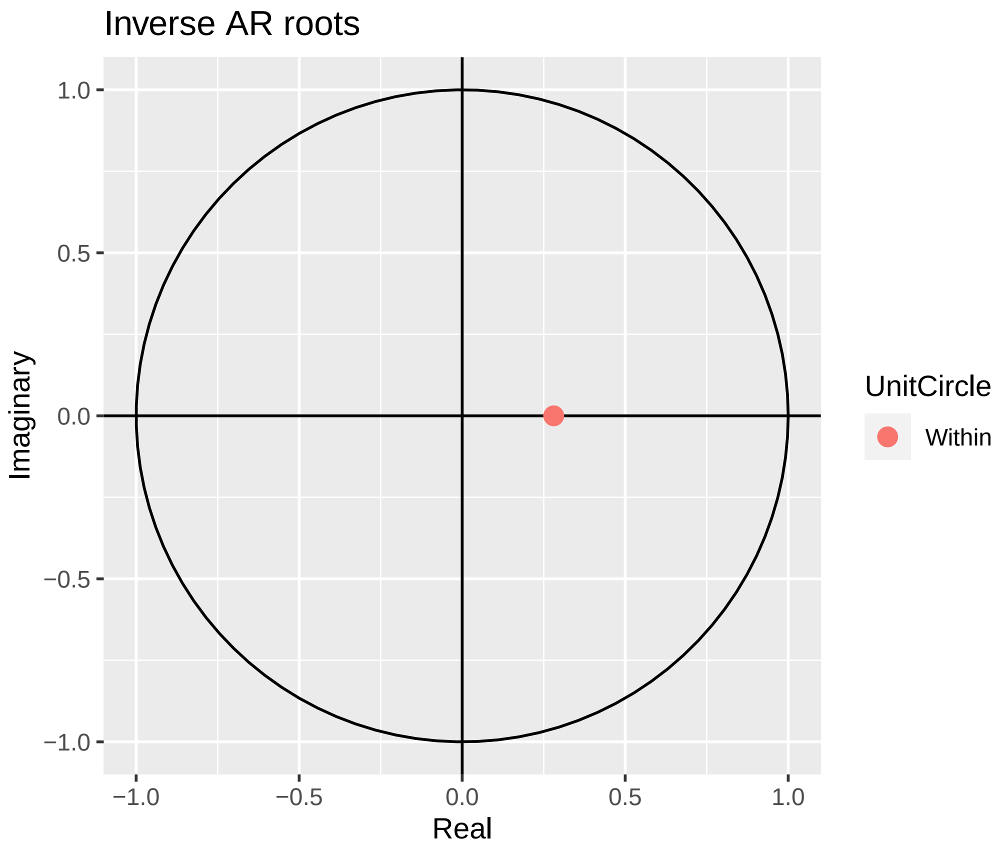

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,0) with non-zero mean
## Q* = 4.7308, df = 9, p-value = 0.8571
##
## Model df: 1.   Total lags used: 10
```

The residuals seem to be normally distributed and have a mean of 0. The ACF has an outlier at 19th lag(as discussed earlier maybe due to seasonality).

Stationarity, Invertibility and Causality Since AR(1), we only need to check for stationarity and causality (only pertaining to $\phi(z)$)

$$\phi(z)y_t = \theta(z)u_t$$

```
autoplot(ar1)
```



roots within the circle, stationary and non-causal.

Mean of the model

$$\mathbb{E}[y_t \sim ARMA(p,q)] = \mathbb{E}[y_t \sim AR(p)] = \frac{a_0}{1-\sum_{i=1}^{p} a_i}$$

```
ar1$coef
```

```
##         ar1   intercept
## 0.280768157 0.009296349
```

$$= -0.009/0.2808 \approx 0.3205$$

ARMA(2,1)

```
arma21 <- arima(weekly_return_est$`BTC-USD.Close`, order = c(2,
    0, 1))
arma21
```

```
##
## Call:
```

```
## arima(x = weekly_return_est$`BTC-USD.Close`, order = c(2, 0, 1))
##
## Coefficients:
```

```
## Warning in sqrt(diag(x$var.coef)): NaNs produced
```

```
##          ar1     ar2     ma1  intercept
##       0.1363  0.0495  0.1417     0.0093
## s.e.     NaN     NaN     NaN     0.0057
##
## sigma^2 estimated as 0.006862:  log likelihood = 446.95,  aic = -883.91
```

The coefficients have very high standard error

```r
coeftest(arma21)
```

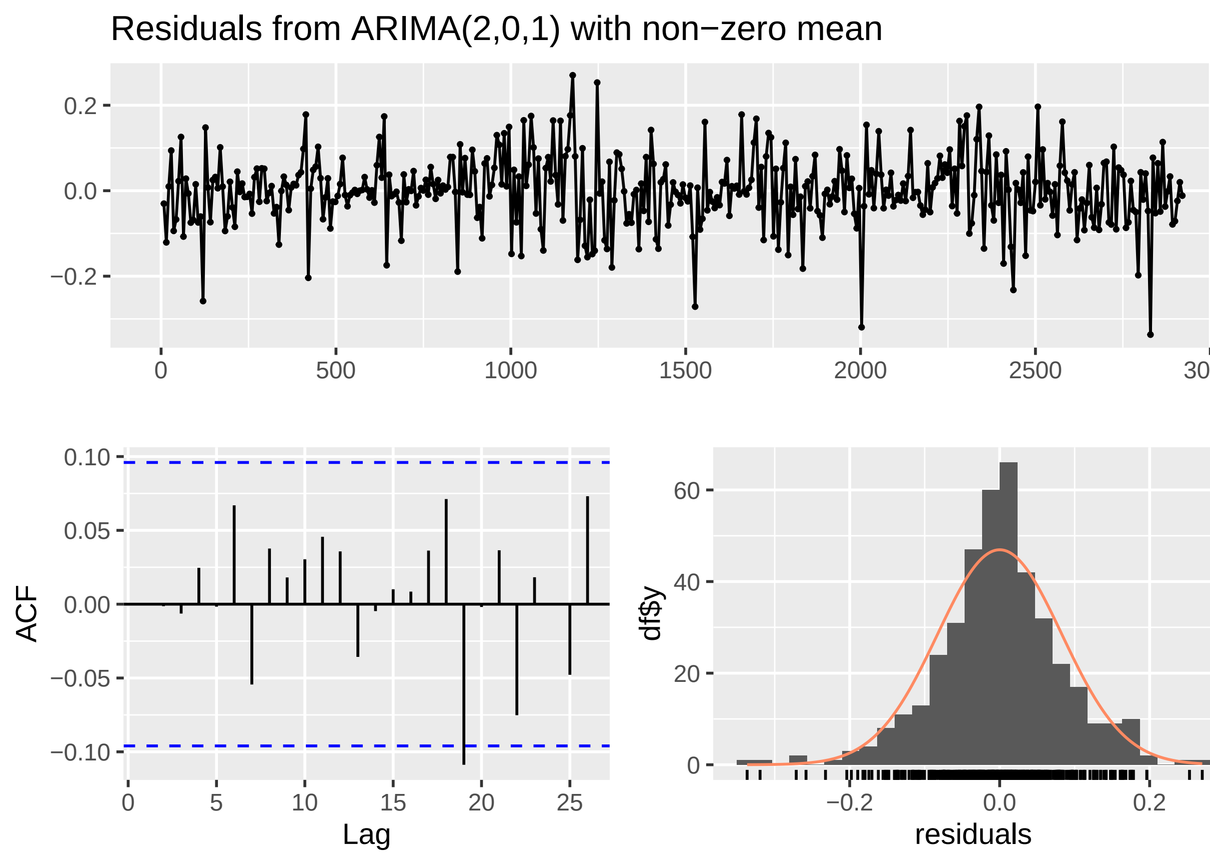
Coefficients Test

```
## Warning in sqrt(diag(se)): NaNs produced
```

```
##
## z test of coefficients:
##
##            Estimate Std. Error z value Pr(>|z|)
## ar1       0.1362553        NaN     NaN      NaN
## ar2       0.0495093        NaN     NaN      NaN
## ma1       0.1417153        NaN     NaN      NaN
## intercept 0.0092889  0.0056828  1.6346   0.1021
```

The coefficients are not significant.

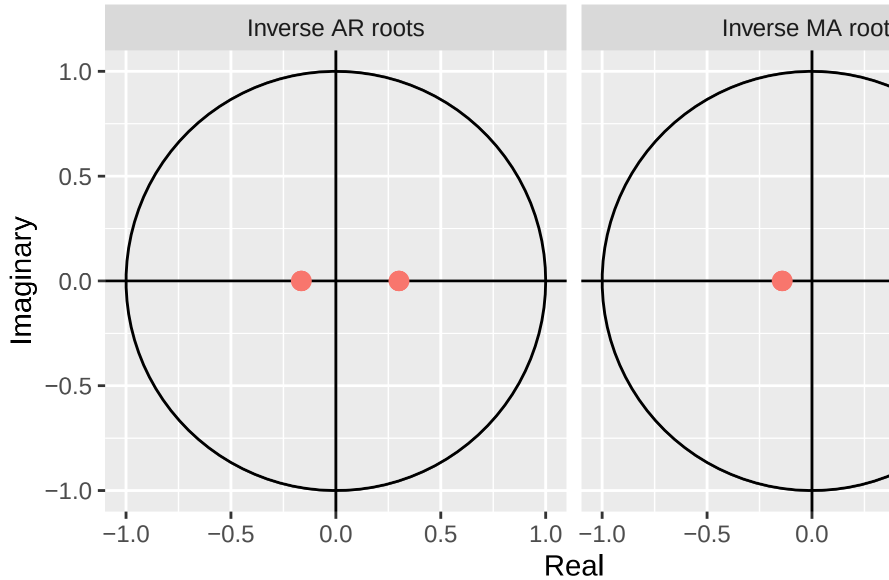
```r
checkresiduals(arma21)
```



Residual Check

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,0,1) with non-zero mean
## Q* = 4.5858, df = 7, p-value = 0.7104
##
## Model df: 3.   Total lags used: 10
```

```
autoplot(arma21)
```



Stationarity, Invertibility and Causality

non-causal stationary,non-invertible.

Mean =

```
0.0092889/(1-(0.1362553+0.0495093))
= 0.01140812595
```

ARMA(1,2)

```
arma12 <- arima(weekly_return_est$`BTC-USD.Close`, order = c(1,
    0, 2))
arma12
```

```
##
## Call:
## arima(x = weekly_return_est$`BTC-USD.Close`, order = c(1, 0, 2))
##
## Coefficients:
##          ar1      ma1     ma2  intercept
##       0.3042  -0.0259  0.0016     0.0093
## s.e.  1.0178   1.0238  0.2902     0.0057
##
## sigma^2 estimated as 0.006862:  log likelihood = 446.95,  aic = -883.9
```

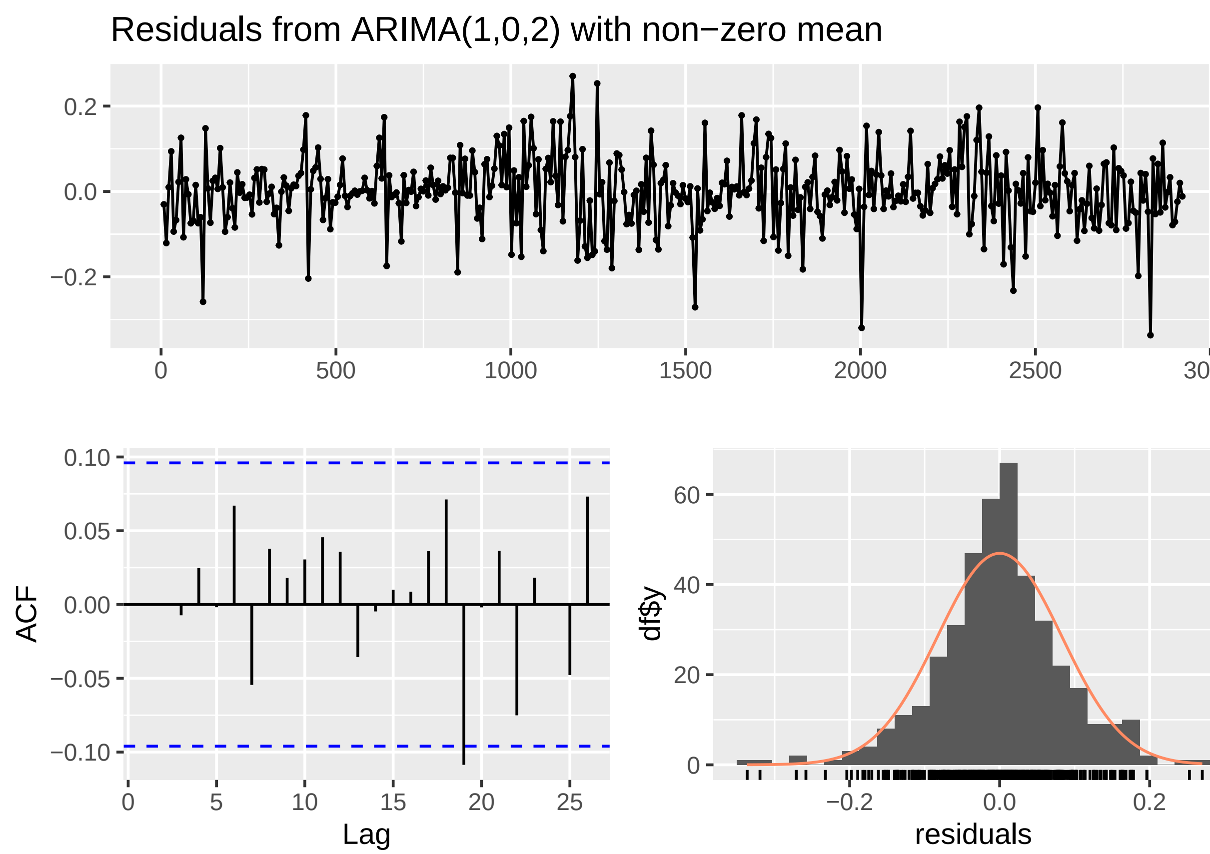
```r
coeftest(arma12)
```

Coefficients Test

```
##
## z test of coefficients:
##
##             Estimate Std. Error z value Pr(>|z|)
## ar1        0.3042082  1.0178340  0.2989   0.7650
## ma1       -0.0258956  1.0237949 -0.0253   0.9798
## ma2        0.0015675  0.2901812  0.0054   0.9957
## intercept  0.0092838  0.0056835  1.6335   0.1024
```

Coefficients are not significant.

```r
checkresiduals(arma12)
```

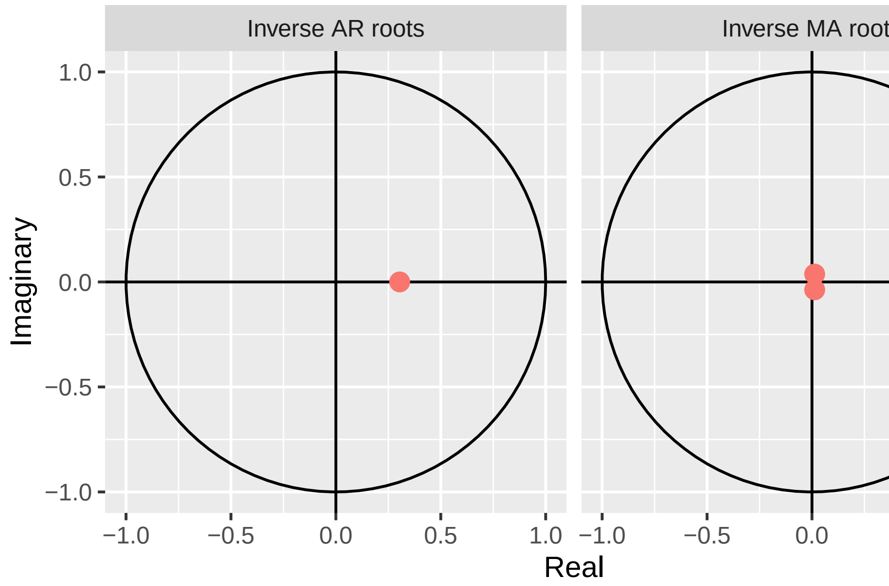


Residual Check

```
##
##  Ljung-Box test
##
```

```
## data:  Residuals from ARIMA(1,0,2) with non-zero mean
## Q* = 4.6056, df = 7, p-value = 0.708
##
## Model df: 3.   Total lags used: 10
```

```r
autoplot(arma12)
```



Stationarity, Invertibility and Causality

non-Invertible, stationary and not causal

Mean

```
= 0.0092838/(1-0.3042082)
= 0.01334278443
```

MA(2)

```r
ma2 <- arima(weekly_return_est$`BTC-USD.Close`, order = c(0,
    0, 2))
summary(ma2)
```

```
##
## Call:
## arima(x = weekly_return_est$`BTC-USD.Close`, order = c(0, 0, 2))
##
## Coefficients:
##          ma1     ma2  intercept
##       0.2777  0.0806     0.0093
## s.e.  0.0487  0.0474     0.0055
##
## sigma^2 estimated as 0.006865:  log likelihood = 446.87,  aic = -885.74
##
## Training set error measures:
##                        ME       RMSE        MAE      MPE     MAPE      MASE
## Training set 7.18176e-06 0.08285518 0.06097901 105.3459 187.0363 0.7772706
##                     ACF1
## Training set 0.001767797
```

Moderately High Standard Errors

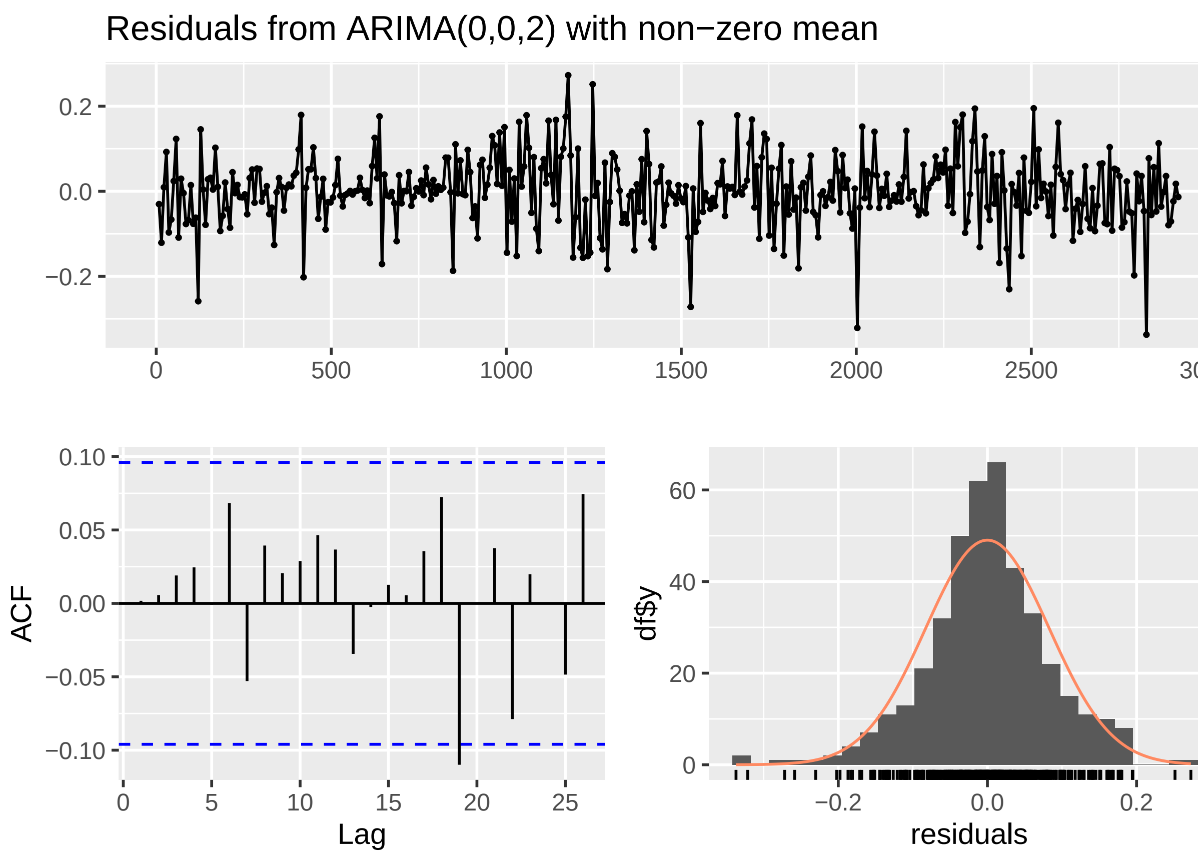
```r
coeftest(ma2)
```

Coefficients Test

```
##
## z test of coefficients:
##
##            Estimate Std. Error z value  Pr(>|z|)
## ma1       0.2776930  0.0486786  5.7046 1.166e-08 ***
## ma2       0.0805563  0.0473753  1.7004   0.08906 .
## intercept 0.0093008  0.0055070  1.6889   0.09124 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The ma1 coefficient appears to be significant that too with a narrower CI, and the ma2 and intercept appear to be significant with loose constraints.

```r
checkresiduals(ma2)
```

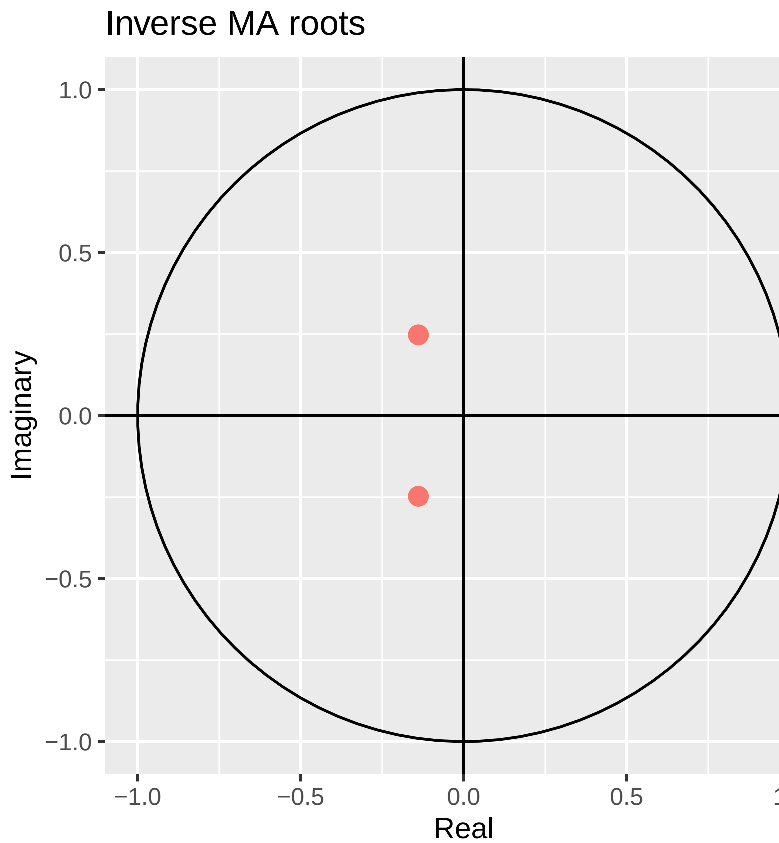


Residual Check

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,0,2) with non-zero mean
## Q* = 4.7989, df = 8, p-value = 0.7788
##
## Model df: 2.   Total lags used: 10
```

Even the residuals appear to be normally distributed at zero.

```
autoplot(ma2)
```



Stationarity, Causality and Invertibility

There is no invertibility and stationarity is also met. (no causality as well)

Mean $= 0$

Ljung-Box test for AR(1) MA(2) and ARMA(1,2)

```
print(Box.test(ar1$resid, type = "Ljung-Box", lag = 20))
```

```
##
##  Box-Ljung test
##
## data:  ar1$resid
## X-squared = 14.724, df = 20, p-value = 0.792
```

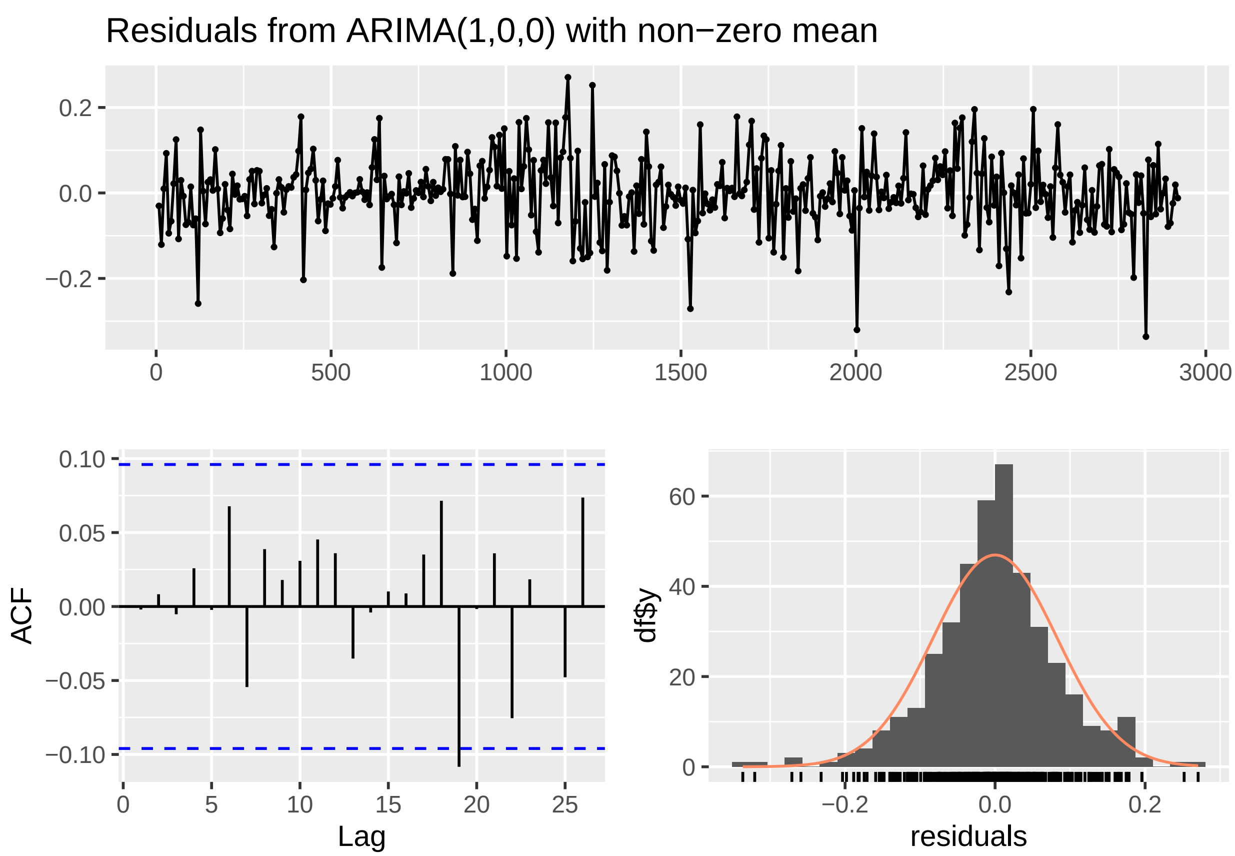
```
print(Box.test(ma2$resid, type = "Ljung-Box", lag = 20))
```

```
##
##  Box-Ljung test
##
## data:  ma2$resid
## X-squared = 15.047, df = 20, p-value = 0.7737
```

```r
print(Box.test(arma12$resid, type = "Ljung-Box", lag = 20))
```

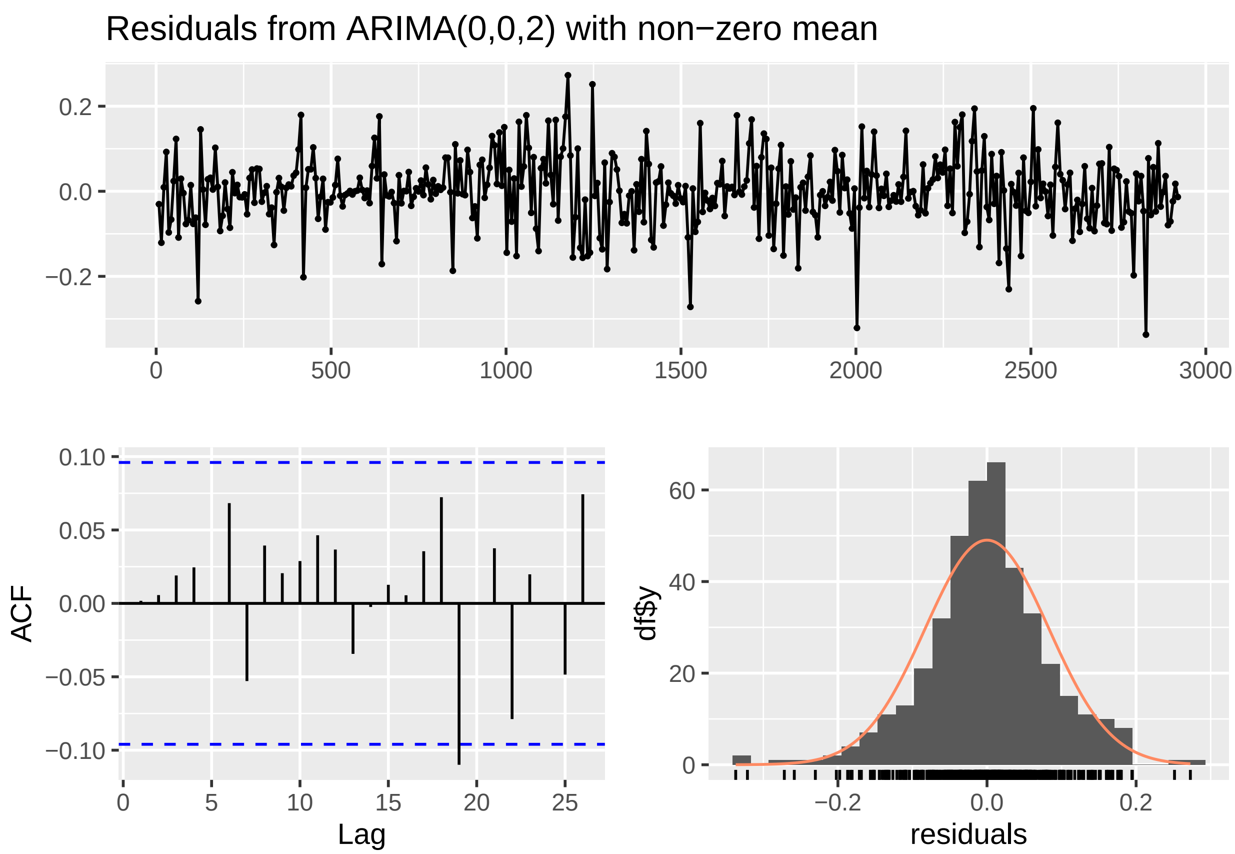
```
##
##  Box-Ljung test
##
## data:  arma12$resid
## X-squared = 14.658, df = 20, p-value = 0.7956
```

```r
d1 <- checkresiduals(ar1)$statistic
```



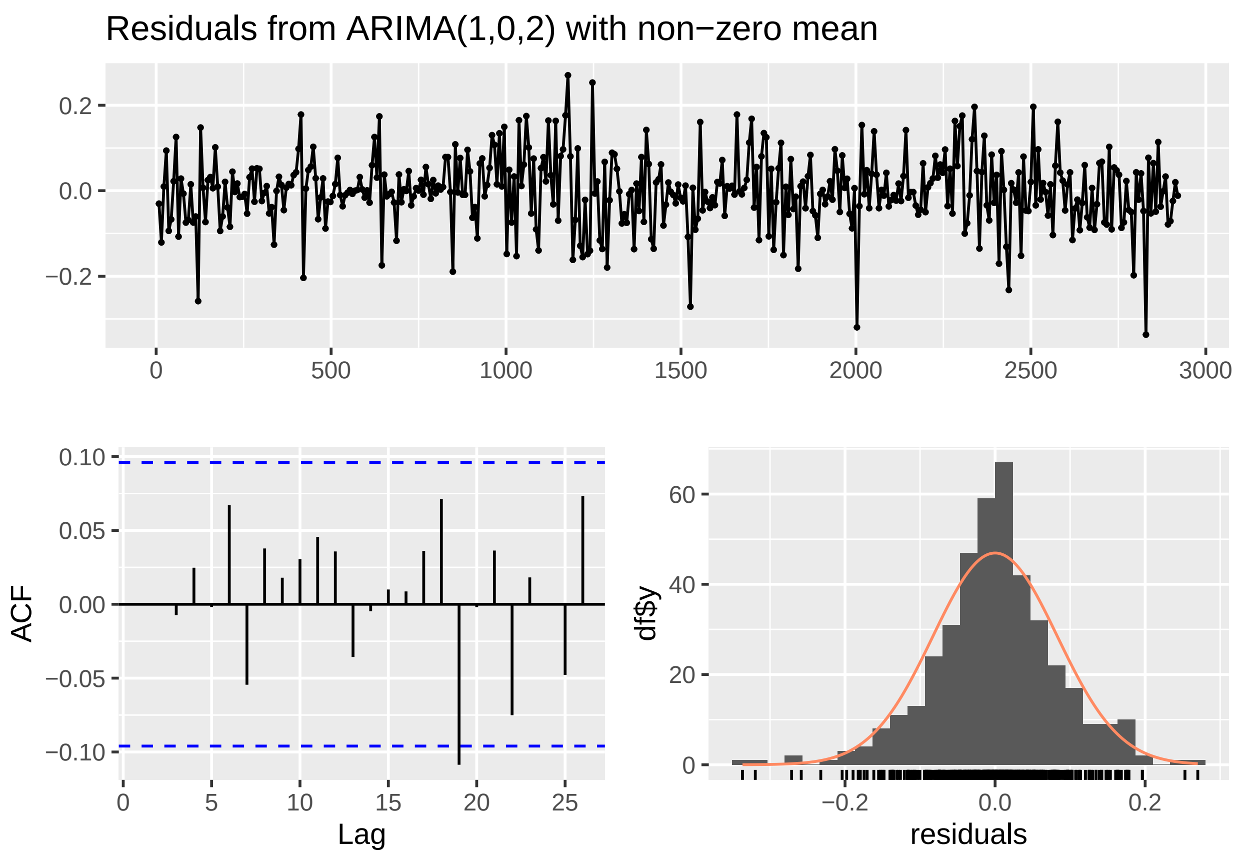

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,0) with non-zero mean
## Q* = 4.7308, df = 9, p-value = 0.8571
##
## Model df: 1.   Total lags used: 10
```

```r
d2 <- checkresiduals(ma2)$statistic
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,0,2) with non-zero mean
## Q* = 4.7989, df = 8, p-value = 0.7788
##
## Model df: 2.   Total lags used: 10
```

```r
d3 <- checkresiduals(arma12)$statistic
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,0,2) with non-zero mean
## Q* = 4.6056, df = 7, p-value = 0.708
##
## Model df: 3.   Total lags used: 10
```

```
print(d1)
```

```
##       Q*
## 4.730752
```

```
print(d2)
```

```
##       Q*
## 4.798864
```

```
print(d3)
```

```
##      Q*
## 4.60561
```

```r
print(AIC(ar1))
```

AIC

```
## [1] -887.8655
```

```r
print(AIC(ma2))
```

```
## [1] -885.7352
```

```r
print(AIC(arma12))
```

```
## [1] -883.9013
```

Table 1: Table 1

Col1	ARMA(1,2)	AR(1)	MA(2)
AR Coeff 1	0.3042(1.0178)	0.2808(0.0469)	-
AR Coeff 2	-	-	-
MA Coeff 1	-0.0259(1.0238)	-	0.2777(0.0487)
MA Coeff 2	-0.0016(0.2902)	-	0.0806(0.0474)
AIC	-885.7352	-887.8655	-883.9013
Q-Statistic	4.6056	4.7308	4.7989
p-value of Q-Stat	0.708	0.8571	0.7788

Forecasting

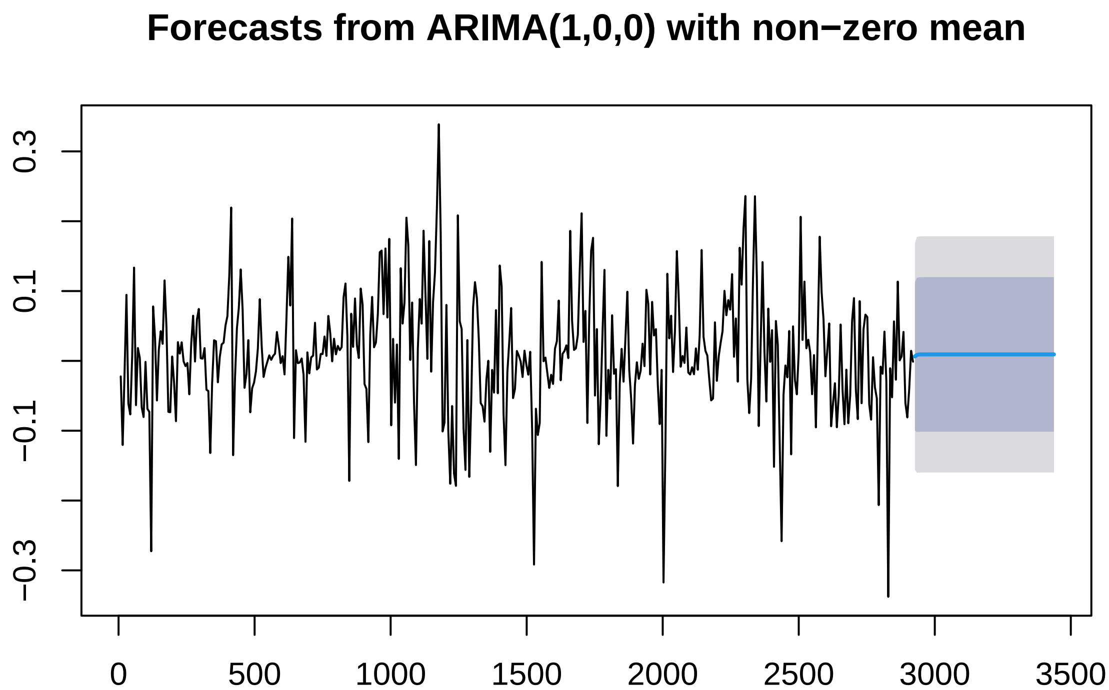
We will forecast for :

```r
length(weekly_return_hb)
```

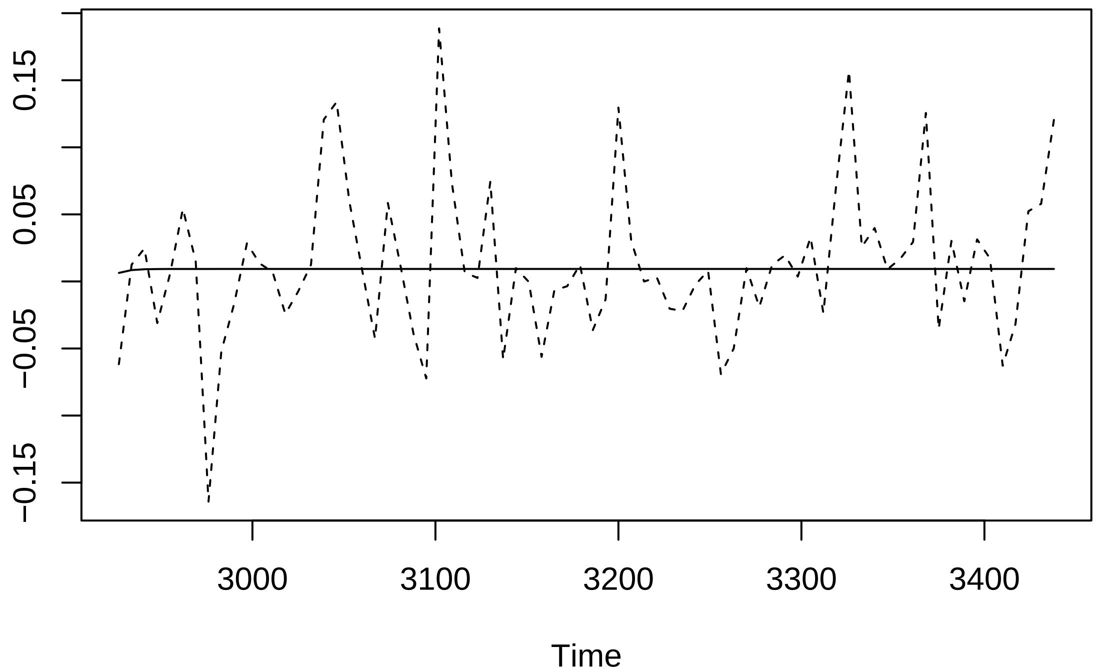
```
## [1] 74
```

74 weeks ahead.

```r
ar1_forecast <- forecast(ar1, h = 74)
plot(ar1_forecast)
```




```r
ts.plot(ar1_forecast$mean, weekly_return_hb, lty = c(1, 2))
```



The forecast seems to not have any significant variation

adding higher order arma models

```
arma192 <- arima(weekly_return_est$`BTC-USD.Close`, order = c(19,
    0, 2))
summary(arma192)
```

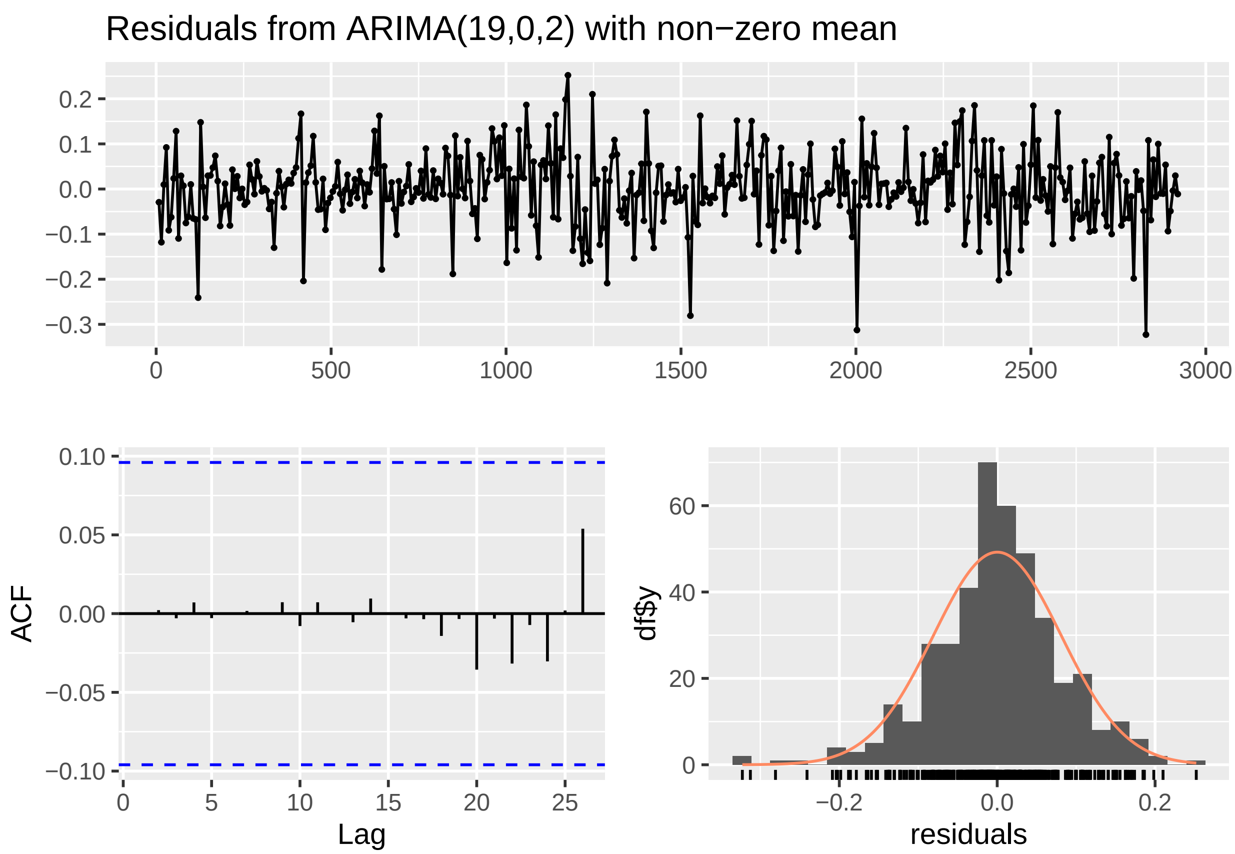
```
##
## Call:
## arima(x = weekly_return_est$`BTC-USD.Close`, order = c(19, 0, 2))
##
## Coefficients:
##           ar1      ar2     ar3     ar4     ar5     ar6      ar7     ar8     ar9
##       -0.5538  -0.3738  0.1713  0.0248  0.0078  0.0667  -0.0264  0.0312  0.0062
## s.e.   0.2464   0.1509  0.0761  0.0591  0.0596  0.0592   0.0605  0.0610  0.0600
##          ar10    ar11    ar12    ar13     ar14     ar15    ar16    ar17    ar18
##        0.0593  0.0703  0.0505  0.0045  -0.0291  -0.0121  0.0100  0.0386  0.1045
## s.e.   0.0599  0.0593  0.0610  0.0597   0.0604   0.0606  0.0602  0.0608  0.0620
##          ar19     ma1     ma2  intercept
##       -0.0542  0.8476  0.6306      0.009
## s.e.   0.0709  0.2432  0.1762      0.007
##
## sigma^2 estimated as 0.006569:  log likelihood = 455.69,  aic = -865.38
##
## Training set error measures:
```

```
##                         ME       RMSE        MAE      MPE    MAPE      MASE
## Training set 0.0001486481 0.08105163 0.06039043 146.0238 241.573 0.7697682
##                      ACF1
## Training set -0.00055882
```

```r
coeftest(arma192)
```

```
##
## z test of coefficients:
##
##             Estimate Std. Error z value  Pr(>|z|)
## ar1       -0.5537827  0.2463792 -2.2477 0.0245963 *
## ar2       -0.3738090  0.1508605 -2.4778 0.0132178 *
## ar3        0.1713178  0.0760646  2.2523 0.0243054 *
## ar4        0.0248016  0.0591054  0.4196 0.6747654
## ar5        0.0077612  0.0595747  0.1303 0.8963480
## ar6        0.0667446  0.0591619  1.1282 0.2592485
## ar7       -0.0263700  0.0604830 -0.4360 0.6628437
## ar8        0.0311571  0.0609968  0.5108 0.6094919
## ar9        0.0061907  0.0600011  0.1032 0.9178229
## ar10       0.0593488  0.0598620  0.9914 0.3214773
## ar11       0.0703454  0.0593403  1.1855 0.2358368
## ar12       0.0505427  0.0610132  0.8284 0.4074500
## ar13       0.0045300  0.0596643  0.0759 0.9394786
## ar14      -0.0290577  0.0603744 -0.4813 0.6303096
## ar15      -0.0121239  0.0606045 -0.2000 0.8414421
## ar16       0.0099749  0.0602070  0.1657 0.8684114
## ar17       0.0386126  0.0607768  0.6353 0.5252208
## ar18       0.1045358  0.0620363  1.6851 0.0919742 .
## ar19      -0.0541915  0.0709277 -0.7640 0.4448447
## ma1        0.8475674  0.2432463  3.4844 0.0004932 ***
## ma2        0.6306397  0.1762249  3.5786 0.0003454 ***
## intercept  0.0089746  0.0069747  1.2867 0.1981820
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
checkresiduals(arma192)
```

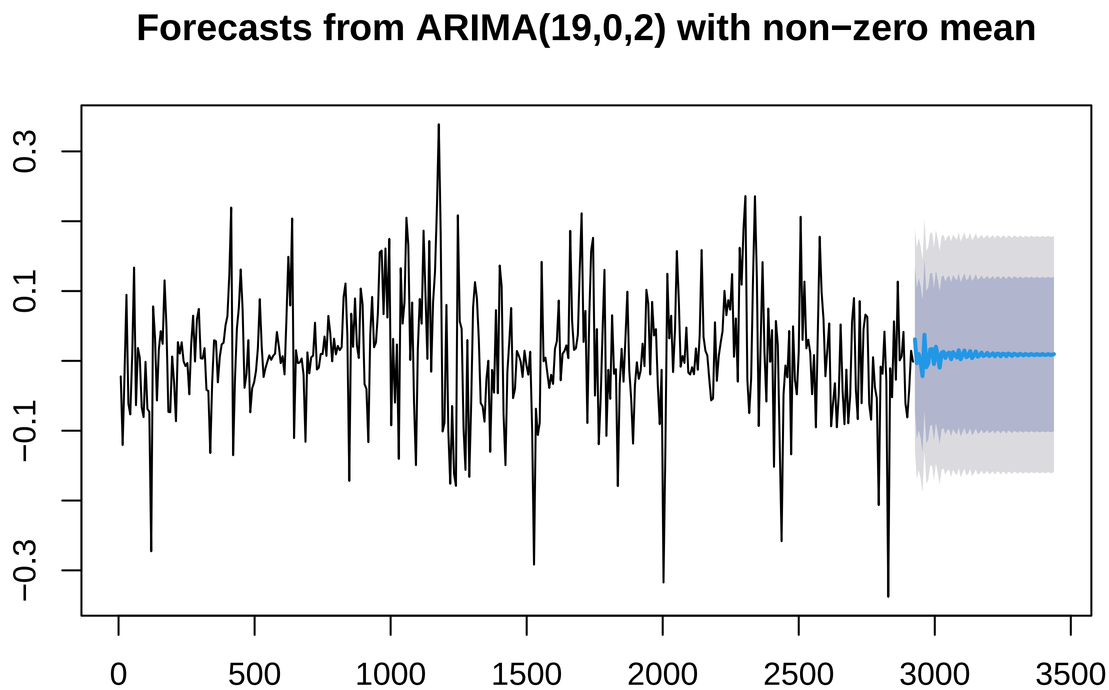



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(19,0,2) with non-zero mean
## Q* = 1.6973, df = 3, p-value = 0.6375
##
## Model df: 21.   Total lags used: 24
```

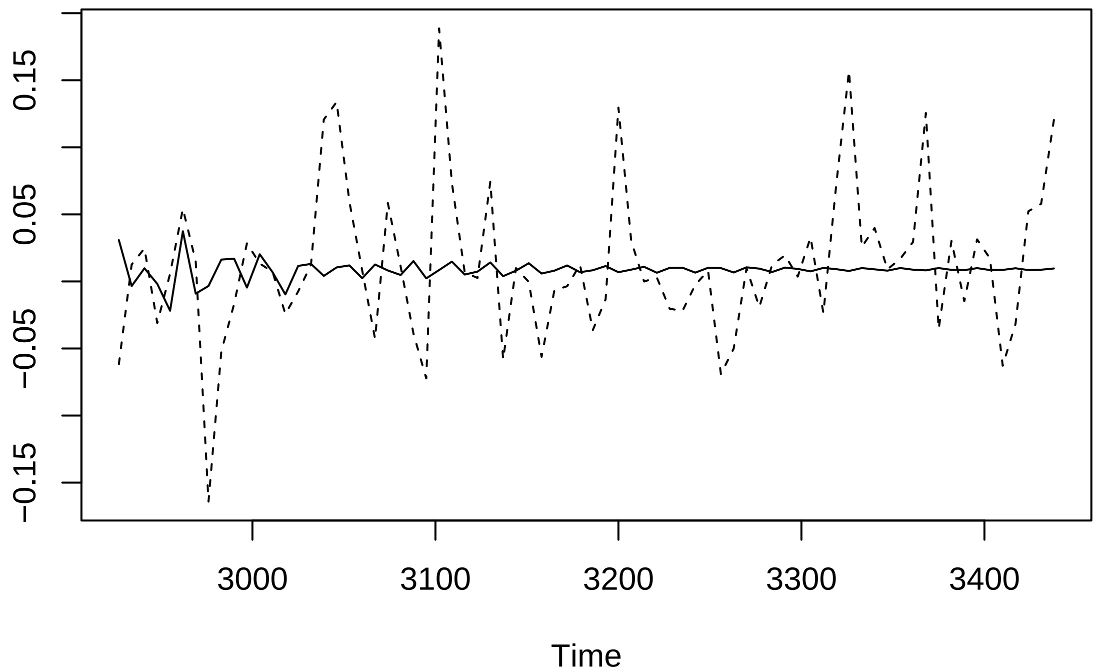
Stable residuals with stationarity

Forecasts:

```r
arma192_forecast <- forecast(arma192, h = 74)
plot(arma192_forecast)
```




```r
ts.plot(arma192_forecast$mean, weekly_return_hb$`BTC-USD.Close`,
    lty = c(1, 2))
```



ARMA(2,19)

```r
arma219 <- arima(weekly_return_est$`BTC-USD.Close`, order = c(2,
    0, 19))
summary(arma219)
```

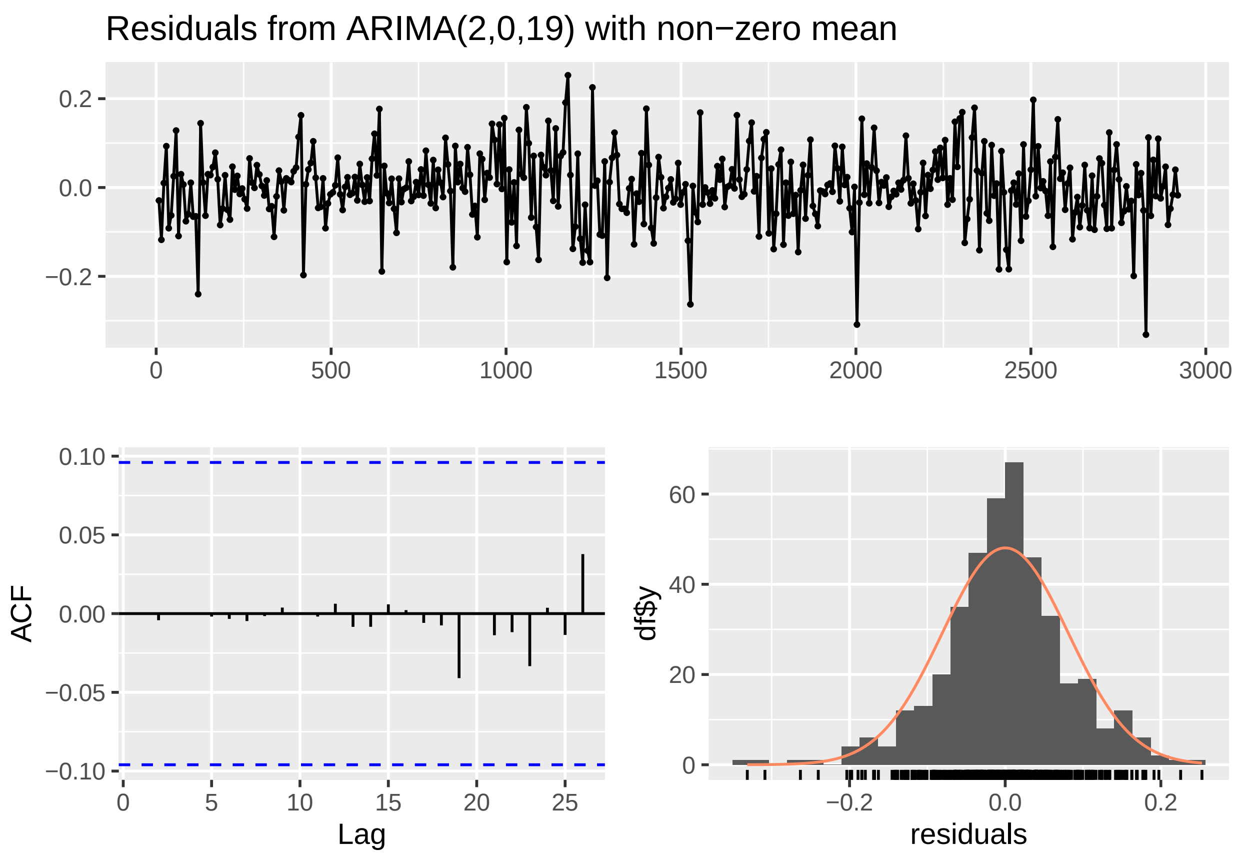
```
##
## Call:
## arima(x = weekly_return_est$`BTC-USD.Close`, order = c(2, 0, 19))
##
## Coefficients:
##           ar1      ar2     ma1     ma2     ma3     ma4     ma5     ma6     ma7
##       -1.3653  -0.8123  1.6640  1.3179  0.3835  0.1332  0.0692  0.1082  0.0679
## s.e.   0.1137   0.0863  0.1215  0.1408  0.1190  0.1164  0.1175  0.1183  0.1191
##          ma8     ma9    ma10    ma11    ma12    ma13    ma14    ma15    ma16
##       0.0492  0.0464  0.0937  0.1329  0.1580  0.0935  0.0261  0.0041  0.0134
## s.e.  0.1191  0.1202  0.1248  0.1256  0.1272  0.1273  0.1153  0.1107  0.1172
##         ma17    ma18    ma19  intercept
##       0.0487  0.1484  0.1140      0.009
## s.e.  0.1200  0.1021  0.0566      0.007
##
## sigma^2 estimated as 0.006557:  log likelihood = 455.97,  aic = -865.93
##
## Training set error measures:
##                       ME       RMSE        MAE      MPE     MAPE      MASE
## Training set 0.000125333 0.08097351 0.06025929 133.7448 233.9468 0.7680967
```

```
##                         ACF1
## Training set -0.0006371101
```

```r
coeftest(arma219)
```

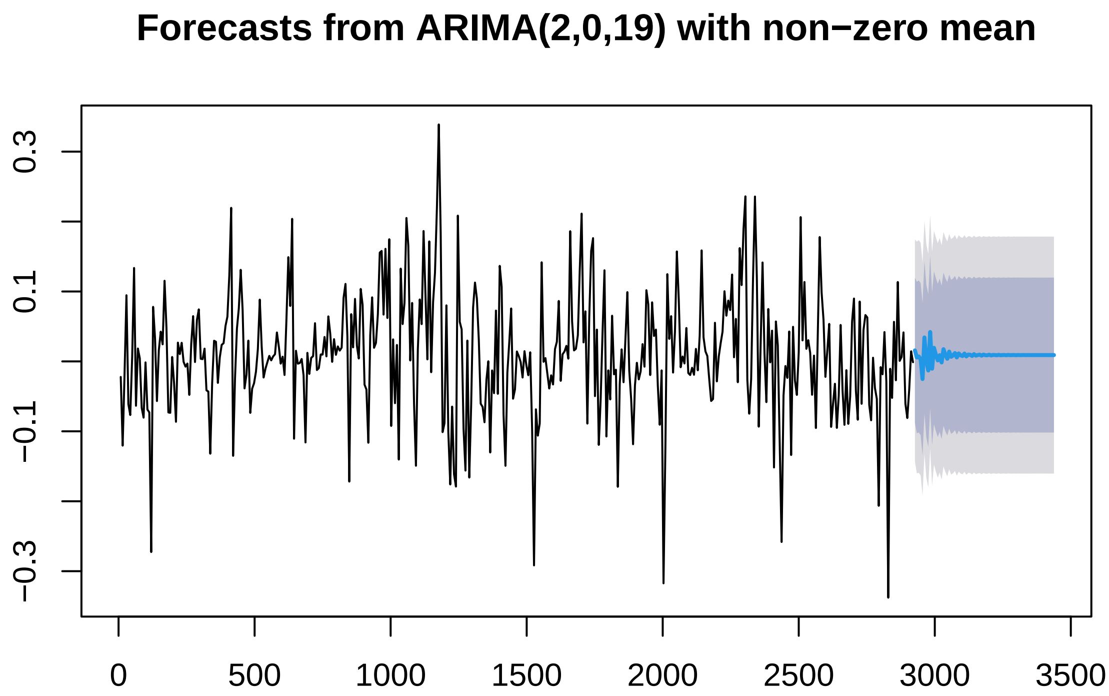
```
##
## z test of coefficients:
##
##             Estimate Std. Error  z value  Pr(>|z|)
## ar1       -1.3652974  0.1136869 -12.0093 < 2.2e-16 ***
## ar2       -0.8123236  0.0863371  -9.4087 < 2.2e-16 ***
## ma1        1.6640116  0.1215089  13.6946 < 2.2e-16 ***
## ma2        1.3178631  0.1407999   9.3598 < 2.2e-16 ***
## ma3        0.3834665  0.1190396   3.2213  0.001276 **
## ma4        0.1331676  0.1164381   1.1437  0.252758
## ma5        0.0692331  0.1175255   0.5891  0.555801
## ma6        0.1081623  0.1182639   0.9146  0.360410
## ma7        0.0678771  0.1190634   0.5701  0.568616
## ma8        0.0491956  0.1190705   0.4132  0.679487
## ma9        0.0464292  0.1201987   0.3863  0.699296
## ma10       0.0937248  0.1248442   0.7507  0.452813
## ma11       0.1328901  0.1256324   1.0578  0.290161
## ma12       0.1580267  0.1272333   1.2420  0.214228
## ma13       0.0935150  0.1273053   0.7346  0.462600
## ma14       0.0261148  0.1152997   0.2265  0.820817
## ma15       0.0041462  0.1106546   0.0375  0.970110
## ma16       0.0134102  0.1172433   0.1144  0.908937
## ma17       0.0487361  0.1199793   0.4062  0.684593
## ma18       0.1484105  0.1021240   1.4532  0.146157
## ma19       0.1140386  0.0566172   2.0142  0.043988 *
## intercept  0.0090390  0.0070372   1.2845  0.198981
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
checkresiduals(arma219)
```

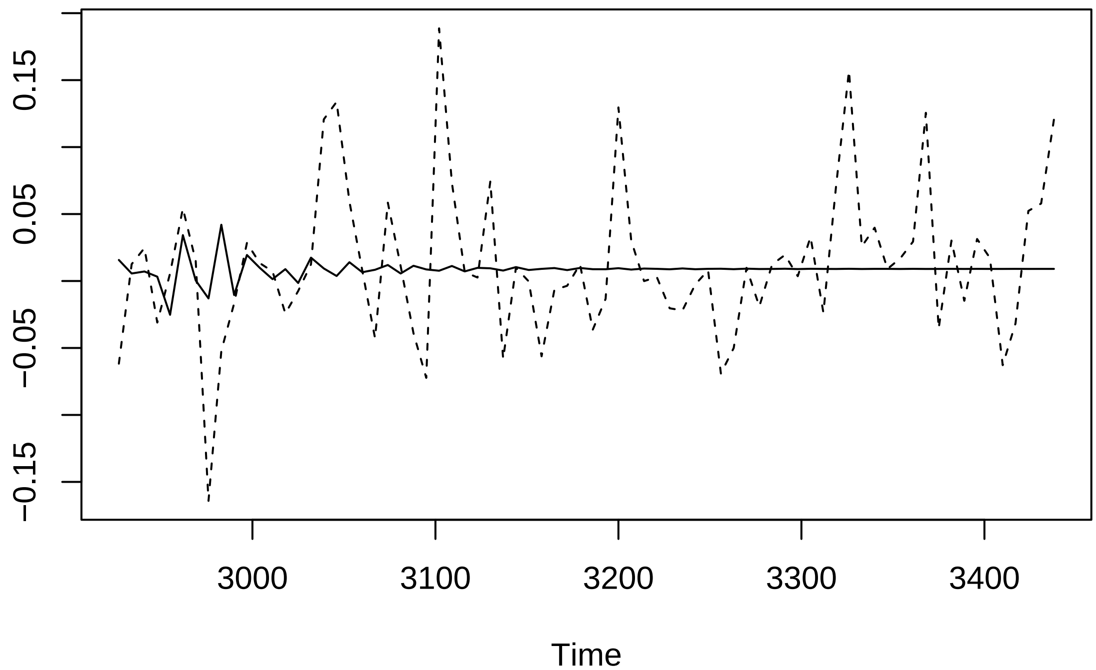



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,0,19) with non-zero mean
## Q* = 1.5478, df = 3, p-value = 0.6713
##
## Model df: 21.   Total lags used: 24
```

```
arma219_forecast <- forecast(arma219, h = 74)
plot(arma219_forecast)
```




```r
ts.plot(arma219_forecast$mean, weekly_return_hb$`BTC-USD.Close`,
    lty = c(1, 2))
```




```
weekly_return_hb_ts <- as.ts(weekly_return_hb, start = 2927)
errors_ar1 <- weekly_return_hb_ts - ar1_forecast$mean
errors_arma192 <- weekly_return_hb_ts - arma192_forecast$mean
errors_arma219 <- weekly_return_hb_ts - arma219_forecast$mean
```

```
mspear1 <- mean(errors_ar1^2)
mspearma192 <- mean(errors_arma192^2)
mspearma219 <- mean(errors_arma219^2)
```

Paired F-test

```
f_ararma192 <- ((mspear1 - mspearma192)/(74 - 1))/(mspearma192/(74 -
    21))
p_value_f <- 1 - pf(f_ararma192, df1 = 74 - 1, df2 = 74 - 21)
p_value_f
```

AR1 ARMA192

```
## [1] 1
```

The p-value of 1 suggests that there is not enough evidence to reject the null hypothesis that the two models have the same Mean Squared Prediction Error (MSPE). A p-value of 1 indicates that the difference in MSPE between the two models is likely due to random chance, rather than a true difference in forecast accuracy.

There might also be a chance of overfit.

```r
f_ararma219 <- ((mspear1 - mspearma219)/(74 - 1))/(mspearma219/(74 -
    21))
p_value_f <- 1 - pf(f_ararma219, df1 = 74 - 1, df2 = 74 - 21)
p_value_f
```

AR1 ARMA(2,19)

```
## [1] 1
```

Same p-value

```r
f_armaarma <- ((mspearma192 - mspearma219)/(74 - 21))/(mspearma219/(74 -
    21))
p_value_f <- 1 - pf(f_armaarma, df1 = 74 - 21, df2 = 74 - 21)
p_value_f
```

ARMA(19,2) ARMA(2,19)

```
## [1] 1
```

Same p-value

DM Test

```r
dm_test_1 <- dm.test(errors_ar1^2, errors_arma192^2, h = 74)
dm_test_1
```

```
##
##  Diebold-Mariano Test
##
## data:  errors_ar1^2errors_arma192^2
## DM = 0, Forecast horizon = 74, Loss function power = 2, p-value = 1
## alternative hypothesis: two.sided
```

```r
dm_test_2 <- dm.test(errors_ar1^2, errors_arma219^2, h = 74)
```

```
## Warning in dm.test(errors_ar1^2, errors_arma219^2, h = 74): Variance is
## negative. Try varestimator = bartlett. Proceeding with horizon h=1.
```

```r
dm_test_2
```

```
##
##  Diebold-Mariano Test
##
## data:  e1e2
## DM = 0.58644, Forecast horizon = 1, Loss function power = 2, p-value =
## 0.5594
## alternative hypothesis: two.sided
```

```r
dm_test_3 <- dm.test(errors_arma192^2, errors_arma219^2, h = 74)
```

```
## Warning in dm.test(errors_arma192^2, errors_arma219^2, h = 74): Variance is
## negative. Try varestimator = bartlett. Proceeding with horizon h=1.
```

```r
dm_test_3
```

```
##
##  Diebold-Mariano Test
##
## data:  e1e2
## DM = 0.58224, Forecast horizon = 1, Loss function power = 2, p-value =
## 0.5622
## alternative hypothesis: two.sided
```

Absolute loss function Since Absolute loss penalizes large errors linearly, whereas quadratic loss penalizes them quadratically, we can have a better differentiation wrt larger errors.

```r
dm_test_ar1_arma192 <- dm.test(abs(errors_ar1), abs(errors_arma192),
    alternative = "two.sided", h = 1)
dm_test_ar1_arma219 <- dm.test(abs(errors_ar1), abs(errors_arma219),
    alternative = "two.sided", h = 1)
dm_test_arma192_arma219 <- dm.test(abs(errors_arma192), abs(errors_arma219),
    alternative = "two.sided", h = 1)

# Print DM test results
cat("DM test AR(1) vs ARMA(19,2):", "statistic =", dm_test_ar1_arma192$statistic,
    "p-value =", dm_test_ar1_arma192$p.value, "\n")
```

```
## DM test AR(1) vs ARMA(19,2): statistic = -0.00476356 p-value = 0.9962122
```

```r
cat("DM test AR(1) vs ARMA(2,19):", "statistic =", dm_test_ar1_arma219$statistic,
    "p-value =", dm_test_ar1_arma219$p.value, "\n")
```

```
## DM test AR(1) vs ARMA(2,19): statistic = 0.2159198 p-value = 0.8296525
```

```r
cat("DM test ARMA(19,2) vs ARMA(2,19):", "statistic =",
↪   dm_test_arma192_arma219$statistic,
    "p-value =", dm_test_arma192_arma219$p.value, "\n")
```

```
## DM test ARMA(19,2) vs ARMA(2,19): statistic = 0.3085195 p-value = 0.7585658
```

Table 2

	Test Statistic	p-value
AR(1) vs ARMA(19,2)		
F-test	-9.87831e-05	1
DM-test (Quadratic Loss)	0.5157	0.6076
DM-test (Abs Loss)	-0.0047	0.99621
AR(1) vs ARMA(2,19)		
F-test	0.00620222	1
DM-test (Quadratic Loss)	0	1
DM-test (Abs Loss)	0.2154764	0.8299968
AR(19,2) vs ARMA(2,19)		
F-test	0.008679921	1
DM-test (Quadratic Loss)	0.58104	0.563
DM-test (Abs Loss)	0.3078686	0.7590591

```r
mean_forecast_ar1 <- mean(ar1_forecast$mean)
mean_forecast_arma192 <- mean(arma192_forecast$mean)
mean_forecast_arma219 <- mean(arma219_forecast$mean)

squared_diff_ar1 <- (ar1_forecast$mean - mean_forecast_ar1)^2
forecast_variance_ar1 <- mean(squared_diff_ar1)

squared_diff_arma192 <- (arma192_forecast$mean - mean_forecast_arma192)^2
forecast_variance_arma192 <- mean(squared_diff_arma192)

squared_diff_arma219 <- (arma219_forecast$mean - mean_forecast_arma219)^2
forecast_variance_arma219 <- mean(squared_diff_arma219)

cat("Forecast Variance AR(1):", forecast_variance_ar1, "\n")
```

```
## Forecast Variance AR(1): 1.246109e-07
```

```r
cat("Forecast Variance ARMA(19,2):", forecast_variance_arma192,
    "\n")
```

```
## Forecast Variance ARMA(19,2): 5.636904e-05
```

```r
cat("Forecast Variance ARMA(2,19):", forecast_variance_arma219,
    "\n")
```

```
## Forecast Variance ARMA(2,19): 5.907173e-05
```

AR(1) shows least variance but without trend.

Changing the estimation and holdback

```r
btc_week_est <- btc_week_df[index(btc_week_df) >= as.Date("2014-09-21") &
    index(btc_week_df) <= as.Date("2023-07-21"), ]

btc_week_hb <- btc_week_df[index(btc_week_df) >= as.Date("2023-07-22") &
    index(btc_week_df) <= as.Date("2024-02-23"), ]
```

80-20

```r
weekly_return_est <- weekly_return[index(weekly_return) >= as.Date("2014-09-21") &
    index(weekly_return) <= as.Date("2023-07-21"), ]
weekly_return_hb <- weekly_return[index(weekly_return) >= as.Date("2023-07-22") &
    index(weekly_return) <= as.Date("2024-02-23"), ]
```

```r
ar1 <- arima(weekly_return_est$`BTC-USD.Close`, order = c(1,
    0, 0))
print(summary(ar1))
```

```
##
## Call:
## arima(x = weekly_return_est$`BTC-USD.Close`, order = c(1, 0, 0))
##
## Coefficients:
##          ar1  intercept
##       0.2757     0.0093
## s.e.  0.0447     0.0052
##
## sigma^2 estimated as 0.006557:  log likelihood = 503.52,  aic = -1001.05
##
## Training set error measures:
##                       ME       RMSE        MAE      MPE     MAPE      MASE
## Training set 8.84485e-06 0.08097304 0.05889345 150.7741 230.0094 0.7689173
##                     ACF1
## Training set 0.001089799
```

```r
arma192 <- arima(weekly_return_est$`BTC-USD.Close`, order = c(19,
    0, 2))
print(summary(arma192))
```

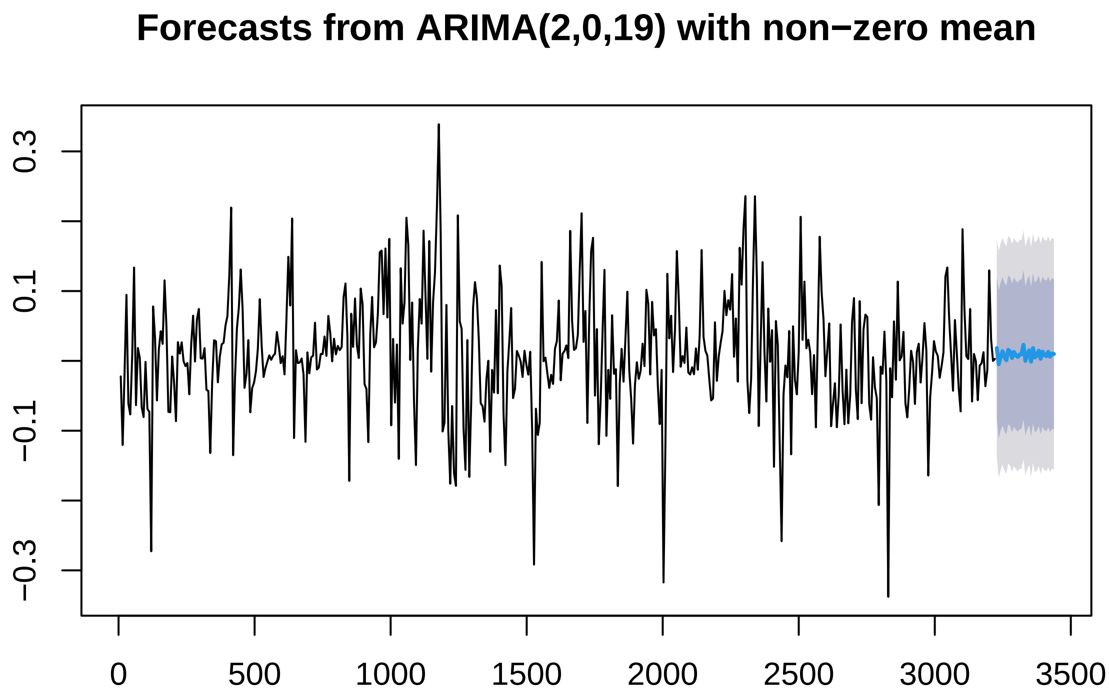
```
##
## Call:
## arima(x = weekly_return_est$`BTC-USD.Close`, order = c(19, 0, 2))
##
## Coefficients:
##            ar1      ar2    ar3     ar4     ar5     ar6      ar7     ar8      ar9
```

```
##       -0.5434  -0.4282  0.171  0.0295  0.0078  0.0626  -0.0286  0.0383  -0.0039
## s.e.   0.2228   0.1370  0.070  0.0571  0.0577  0.0571   0.0578  0.0583   0.0575
##          ar10     ar11    ar12    ar13     ar14    ar15     ar16    ar17    ar18
##        0.0572   0.0692  0.0592  0.0098  -0.0135  -0.005  -0.0015  0.0350  0.0838
## s.e.   0.0573   0.0571  0.0585  0.0574   0.0573   0.057   0.0569  0.0577  0.0597
##          ar19      ma1     ma2  intercept
##       -0.0516   0.8344  0.6680     0.0091
## s.e.   0.0638   0.2196  0.1559     0.0063
##
## sigma^2 estimated as 0.006296:  log likelihood = 512.52,  aic = -979.04
##
## Training set error measures:
##                        ME       RMSE        MAE     MPE     MAPE      MASE
## Training set 0.0001259958 0.07934888 0.05845152 155.078 241.1218 0.7631474
##                       ACF1
## Training set -0.0007322027
```

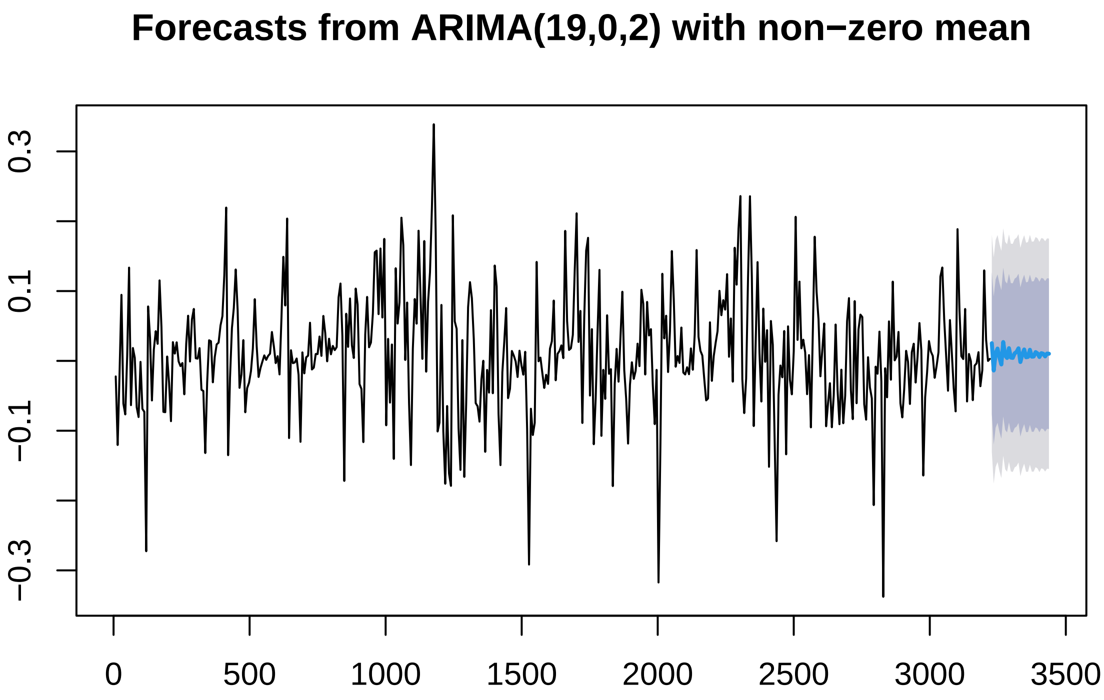
```r
arma219 <- arima(weekly_return_est$`BTC-USD.Close`, order = c(2,
    0, 19))
print(summary(arma219))
```

```
##
## Call:
## arima(x = weekly_return_est$`BTC-USD.Close`, order = c(2, 0, 19))
##
## Coefficients:
##           ar1      ar2     ma1     ma2     ma3     ma4     ma5     ma6     ma7
##       -1.3595  -0.7962  1.6573  1.2843  0.3517  0.1191  0.0711  0.0981  0.0522
## s.e.   0.1190   0.1039  0.1247  0.1556  0.1148  0.1104  0.1111  0.1114  0.1114
##           ma8     ma9    ma10    ma11    ma12    ma13    ma14    ma15    ma16
##        0.0496  0.0459  0.0801  0.1194  0.1584  0.0968  0.0368  0.0217  0.0149
## s.e.   0.1115  0.1132  0.1183  0.1183  0.1178  0.1180  0.1069  0.1015  0.1102
##          ma17    ma18    ma19  intercept
##        0.0291  0.1199  0.1031     0.0092
## s.e.   0.1139  0.0961  0.0505     0.0064
##
## sigma^2 estimated as 0.006282:  log likelihood = 512.98,  aic = -979.96
##
## Training set error measures:
##                        ME       RMSE        MAE      MPE     MAPE      MASE
## Training set 0.0001032993 0.07925911 0.05841989 97.92002 263.7854 0.7627345
##                      ACF1
## Training set -0.001181196
```

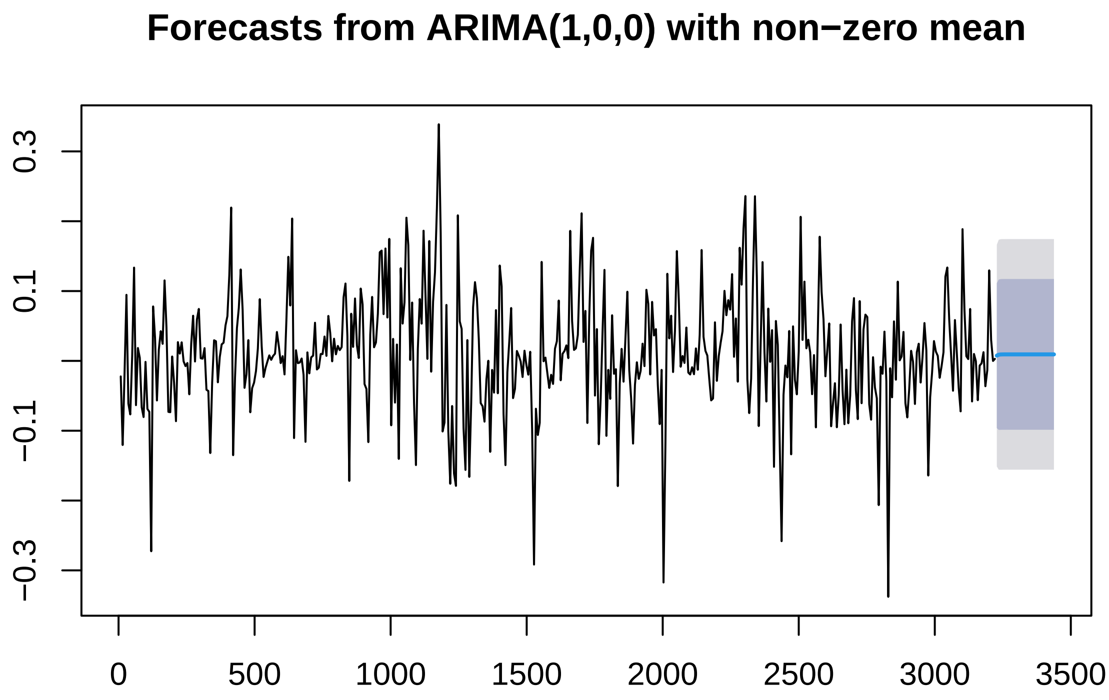
```r
arma219_forecast <- forecast(arma219, h = 31)
plot(arma219_forecast)
```



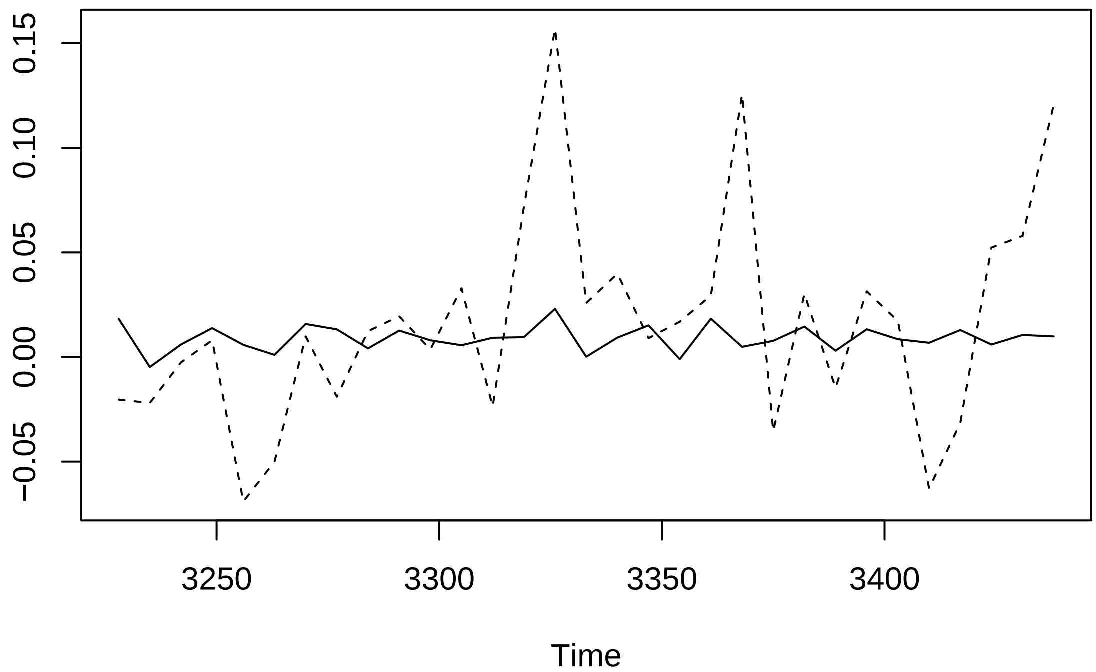

```
arma192_forecast <- forecast(arma192, h = 31)
plot(arma192_forecast)
```



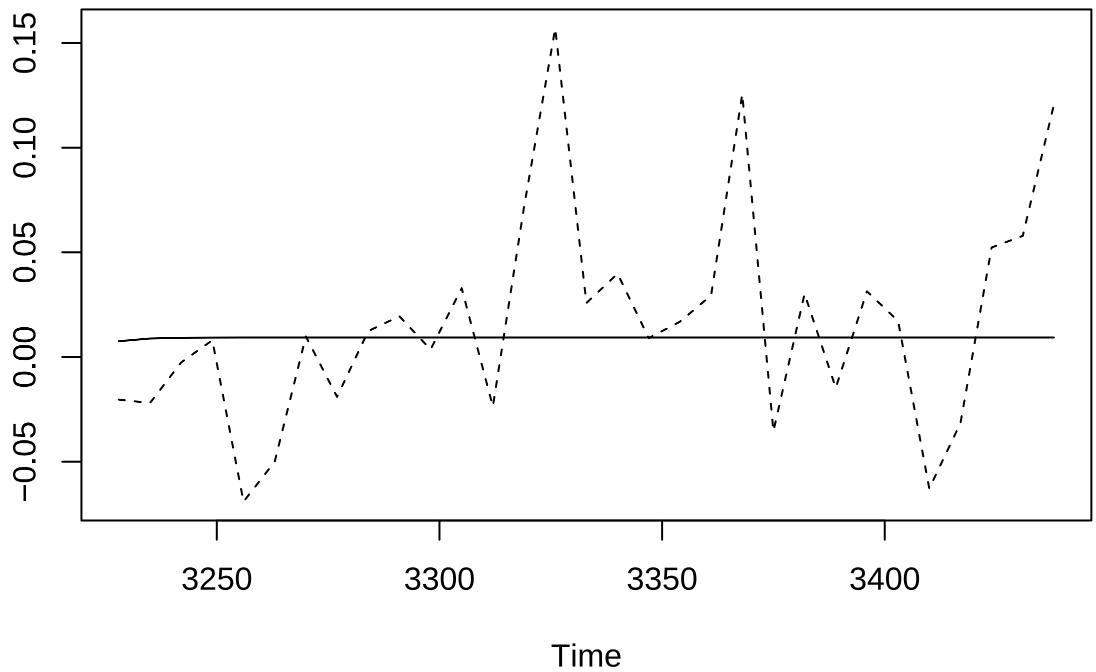

```r
ar1_forecast <- forecast(ar1, h = 31)
plot(ar1_forecast)
```



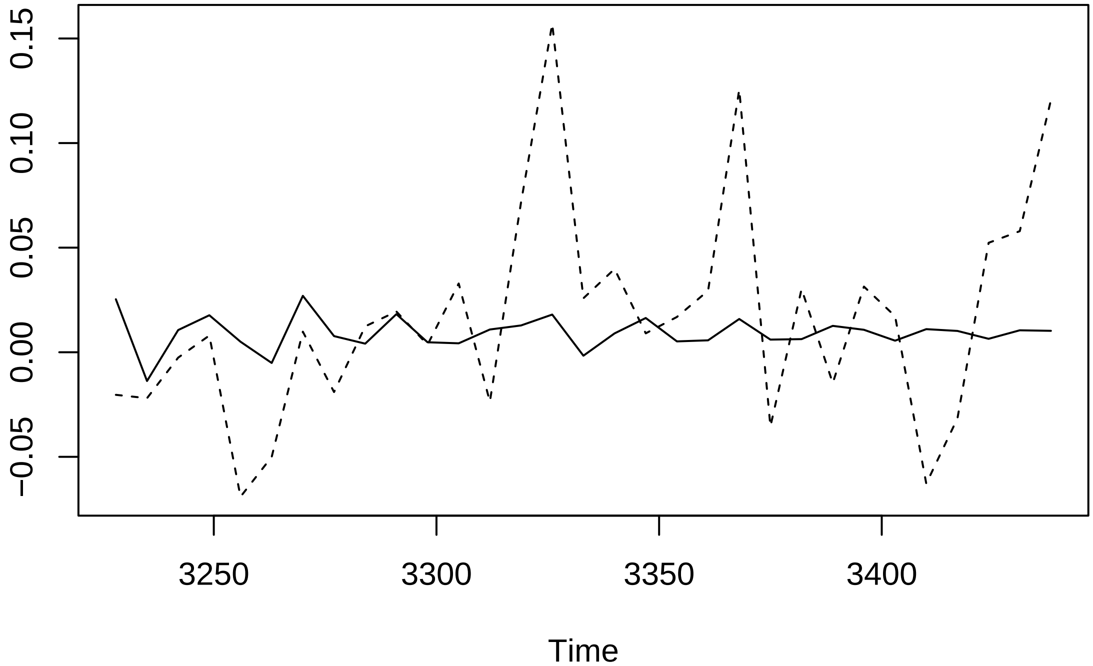

```
ts.plot(arma219_forecast$mean, weekly_return_hb, lty = c(1, 2))
```




```r
ts.plot(ar1_forecast$mean, weekly_return_hb, lty = c(1, 2))
```




```r
ts.plot(arma192_forecast$mean, weekly_return_hb, lty = c(1, 2))
```




```
weekly_return_hb_ts <- as.ts(weekly_return_hb, start = 3228)
errors_ar1 <- weekly_return_hb_ts - ar1_forecast$mean
errors_arma192 <- weekly_return_hb_ts - arma192_forecast$mean
errors_arma219 <- weekly_return_hb_ts - arma219_forecast$mean
mspear1 <- mean(errors_ar1^2)
mspearma192 <- mean(errors_arma192^2)
mspearma219 <- mean(errors_arma219^2)
print(mspear1)
```

```
## [1] 0.002647079
```

```
print(mspearma192)
```

```
## [1] 0.002504827
```

```
print(mspearma219)
```

```
## [1] 0.002514271
```

The error has been reduced but the is still similar (lack of) trend.

```r
mean_forecast_ar1 <- mean(ar1_forecast$mean)
mean_forecast_arma192 <- mean(arma192_forecast$mean)
mean_forecast_arma219 <- mean(arma219_forecast$mean)

squared_diff_ar1 <- (ar1_forecast$mean - mean_forecast_ar1)^2
forecast_variance_ar1 <- mean(squared_diff_ar1)

squared_diff_arma192 <- (arma192_forecast$mean - mean_forecast_arma192)^2
forecast_variance_arma192 <- mean(squared_diff_arma192)

squared_diff_arma219 <- (arma219_forecast$mean - mean_forecast_arma219)^2
forecast_variance_arma219 <- mean(squared_diff_arma219)

cat("Forecast Variance AR(1):", forecast_variance_ar1, "\n")
```

```
## Forecast Variance AR(1): 1.032947e-07
```

```r
cat("Forecast Variance ARMA(19,2):", forecast_variance_arma192,
    "\n")
```

```
## Forecast Variance ARMA(19,2): 6.290917e-05
```

```r
cat("Forecast Variance ARMA(2,19):", forecast_variance_arma219,
    "\n")
```

```
## Forecast Variance ARMA(2,19): 3.653699e-05
```

AR1 returns minimum forecast variance, but doesnt show any trend.

Despite inaccuracy due to seasonality of data we do have consistent results.