# GARCH model for the volatility of cryptocurrencies

Jimmy Ortega Bustamante[1]

[1] *MSC Financial Engineering, FIEECS, Universidad Nacional de Ingeniería, Lima, Perú*

**ABSTRACT**

*Virtual currencies (cryptocurrencies) have various fluctuations and changes over time like any currency. The present study is interested in investigating and analyzing the main conditions of productivity and variability in these currencies. Some of the main features of these cryptocurrencies are the high rate of fluctuation and variability, the uncertainty in the prices and a long memory in the variation.*

*An estimated model of random variability with several variants with non-continuous jumps to an intermediate productivity was formulated. This formulated model can give us as an answer some non-common values within the discontinuity of the raised uncertainty. To achieve optimal results, 10 of the most important cryptocurrencies were analyzed (with daily observations for 2 years). The results indicate that in 2 time periods (2021 and early 2022) there was a lot of variability. This component of variability seems to be stimulated by the growth and change in the market and by the attractiveness of digital currencies for investors. In the first period of 2021, the jumps in the variation transition were of greater magnitude and frequency, all of which would cause some changes in the price and profitability of the currencies. This subordination to the long memory of digital currencies has a greater tendency to increase when it comes to stable formulations but with jumps or transitions in between. Currency prices are mostly non-stationary series that agglomerate sets of volatilities; These characteristics make them candidates to model said volatilities through heteroscedastic autoregressive models.*

**Keyword:** - *Bitcoin, Cryptocurrencies, Risk, Variability, Jumps, Large memory, Digital asset, GARCH*

## 1. INTRODUCTION

Virtual currencies (cryptocurrencies) have various fluctuations and changes over time like any currency. The present study is interested in investigating and analyzing the main conditions of productivity and variability in these currencies. Some of the main features of these cryptocurrencies are the high rate of fluctuation and variability, the uncertainty in the prices and a long memory in the variation. The recent attention to the use of statistics to give greater depth to various variables of the digital asset market has resulted in essential aspects that should have greater scope. The suggested random model with jumps in the average and variability exemplifies the trends found in practice regarding the variability under condition of these assets and generates extensive information that remains current in the markets.

According to Coinmarketcap (2022) [3] the market capitalization of bitcoin at June 2022 reached 35 billion USD, and this is free of any intervention from regulatory authorities, such as central banks, the distributed block chain system in the which is based on Bitcoin and fulfilling the different levels of transaction settlement demand, and the series of Bitcoin exchange rates for the main currencies such as USD, EUR or GBP are known. Therefore, it is not uncommon for gains or losses in the tens of percent to occur within a week, if not during a single day.

There are several Bitcoin exchanges such as Binance, Coinbase, Kraken, FTX or Kucoin for the USD or EUR coins to name just a few of the currently active BTC to EUR exchanges. The highly volatile nature of the exchange rate represents an ideal environment for the study of extreme events in the field of financial time series. The prediction of extreme events is a key topic not only in economics, but also in climatology, geosciences, civil engineering, space technology, etc. Despite its importance, in our opinion, the subject is little studied.

In this paper, we will seek to explore the applicability of computational intelligence methods from financial analysis to Bitcoin exchange rate series. Since the Bitcoin price process is not stationary, but shows an appreciating trend, we

transform the time series data into log returns. If the absolute value of the logarithm return is large, it corresponds to an extreme event.

Much related scientific research focuses on the activity of variability in relation to the productivity of digital assets (cryptocurrencies). GARCH (generalized autoregressive conditional heteroscedasticity) samples are widely used in this type of study (Dyhrberg 2016) [5].

The demonstration of Dimpfl and Baur (2018) [1] results in the application of a sample based on the methodology mentioned above (GARCH) at a general level in any range, it can be coupled in a better statistical way for some specific digital currencies (4 to be exact) and an analysis of the valuation was additionally made in terms of expected and unexpected, dependent and independent risks. Risky specifications in the final stage or queue of the process are analyzed in the studies by Katsiampa and Gkillas (2018) [6]. The presence of jumps in the model and probable changes of direction were analyzed by Laurini, Fry, Ardía, Chaim, among other professionals in 2018 [4].

### 1.1 Value at Risk VaR

When we refer to the Value at Risk (VaR for its acronym in English) it is one of the development methods so that investors can visualize all the existing risks that have a positive relevance in the companies as a whole; these were developed by JP Morgan analysts. In this way, the JP Morgan VaR algorithm has a set of data necessary for the calculations, which were made freely available to the public in November 1994. This led to the adoption of the method within the market. In the same way, VaR drew attention in the literature and went on a path of its own. We can understand that this is not only focused on professionals within the securities markets, but also on banks, pension funds, financial institutions and non-financial companies that seek to adopt this method.

Therefore, according to Bolgun and Akcay (2009) [2] from an academic point of view, the VaR can be described as a methodology that could be able to estimate a greater loss than it could suggest throughout a certain period and in a certain confidence interval. in the financial markets by adopting a prospective perspective that can be understood by all in the same way (taking into account with monetary value).

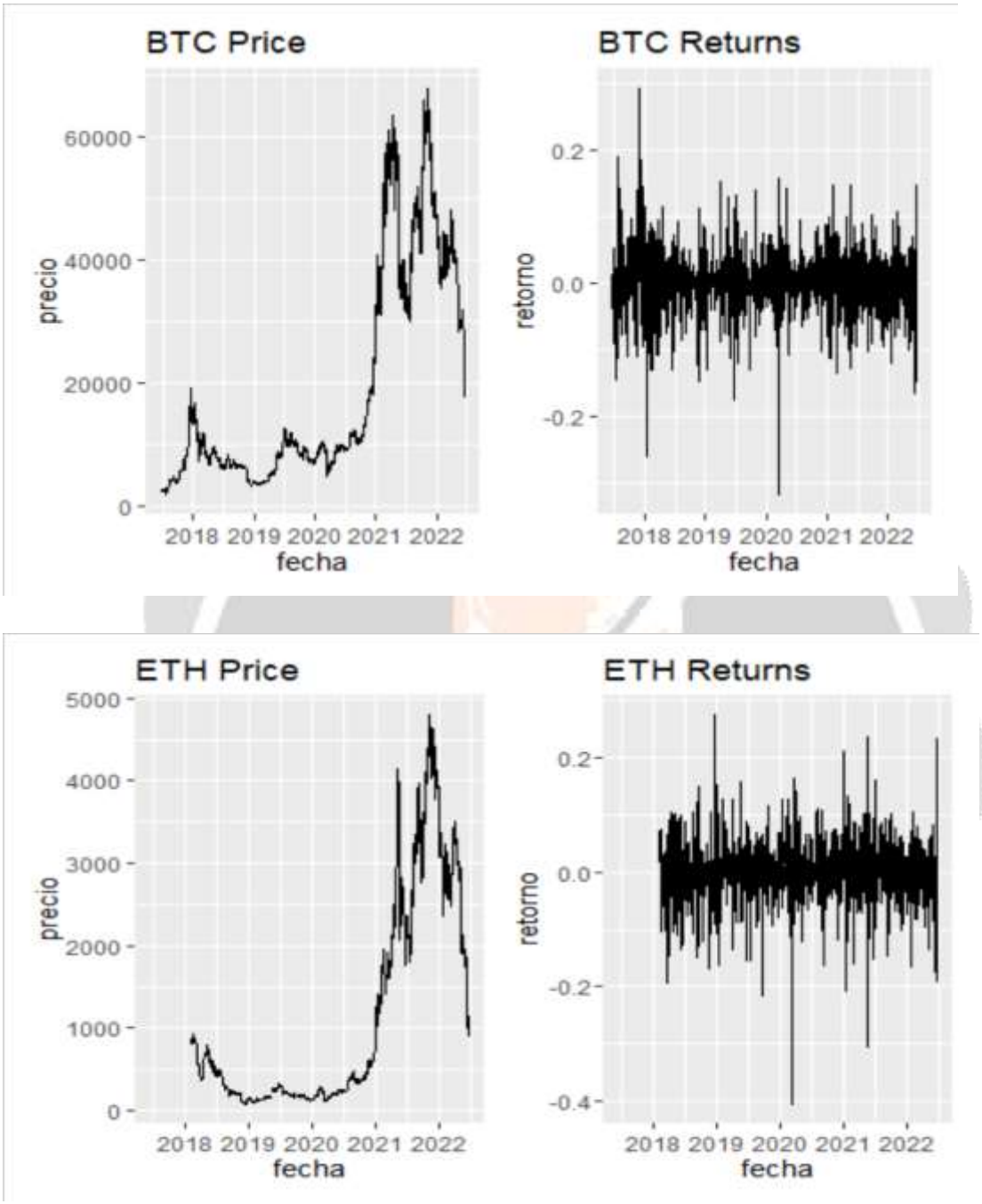
We can determine that the VaR seeks to provide a scale based on the estimation of the probabilities of the loss of a value of a portfolio or of a single asset in a time interval that can be determined. This being the method of consulting statistical calculations, which seeks to denominate the risk in monetary aspects. This analysis has been used in various fields, such as determining reporting risk limits, capital adequacy regulations, internal capital allocation, and performance measurement. Taking this into account, the method is widely used when it comes to measuring the risk of foreign exchange operations; This method seeks to allow estimating the maximum monetary loss subject to a certain type of foreign exchange investment of an investor within a certain time interval.

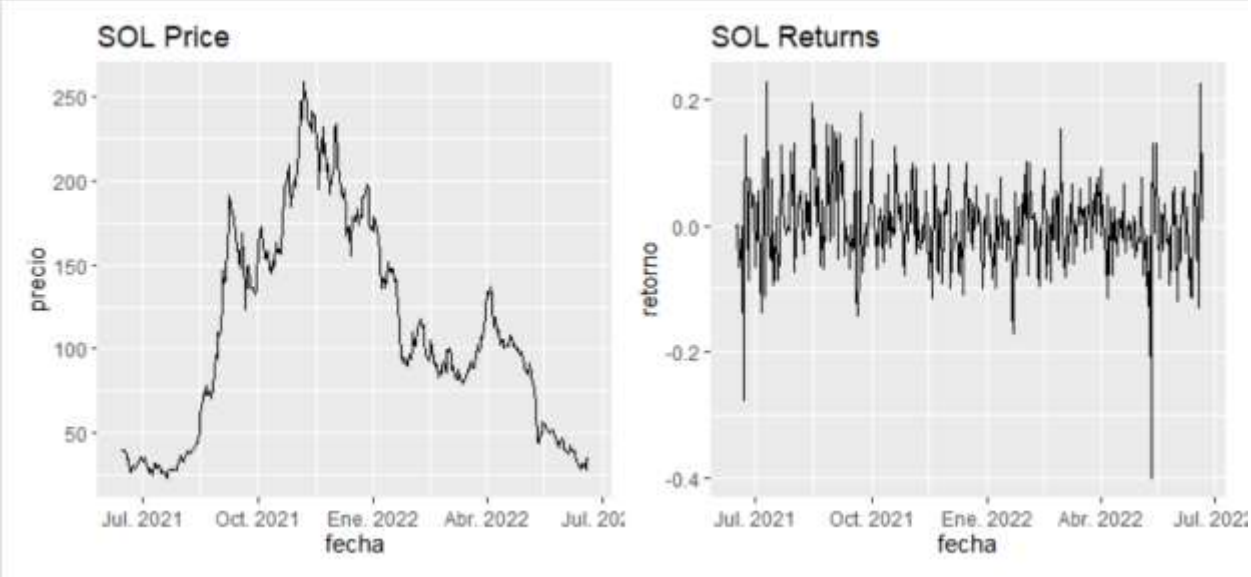
Taking this into account, the present analysis sought to use the daily market closing price of eight major currencies within the world (Swiss franc, euro, British pound, Japanese yen, Australian dollar, Canadian dollar and New Zealand dollar) and Bitcoin. These results can be obtained in different ways of evaluating terms in terms of risk levels to which they are exposed when building their portfolio, taking into account the main currencies or the digital money of Bitcoin.

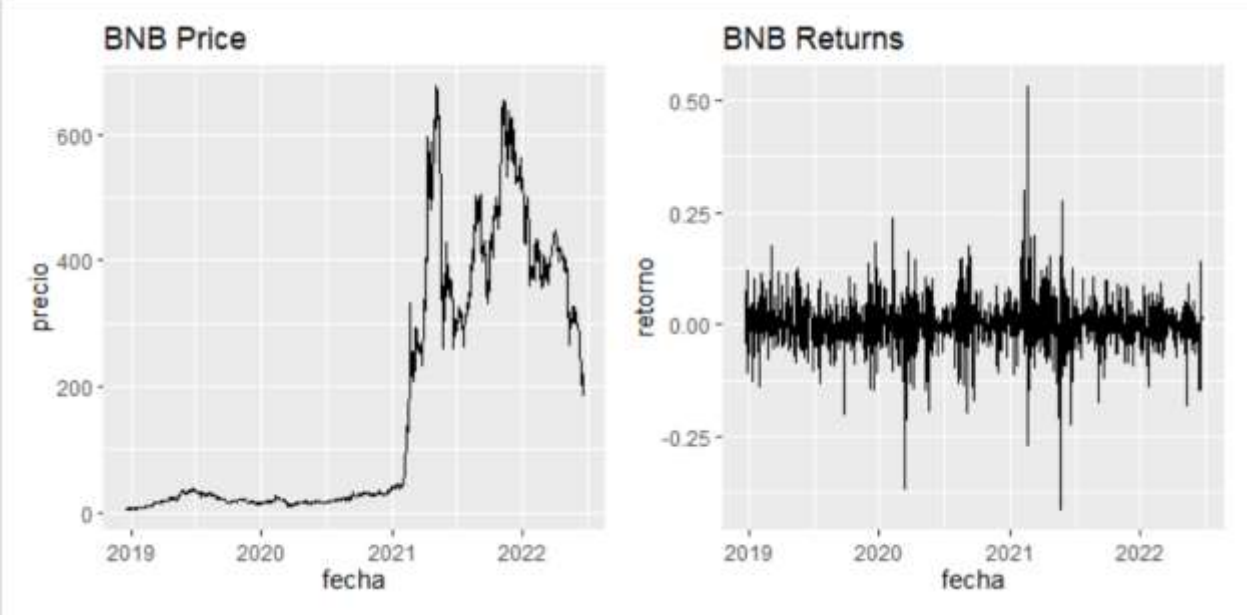
### 1.2 I. Analysis of results, descriptive statistics

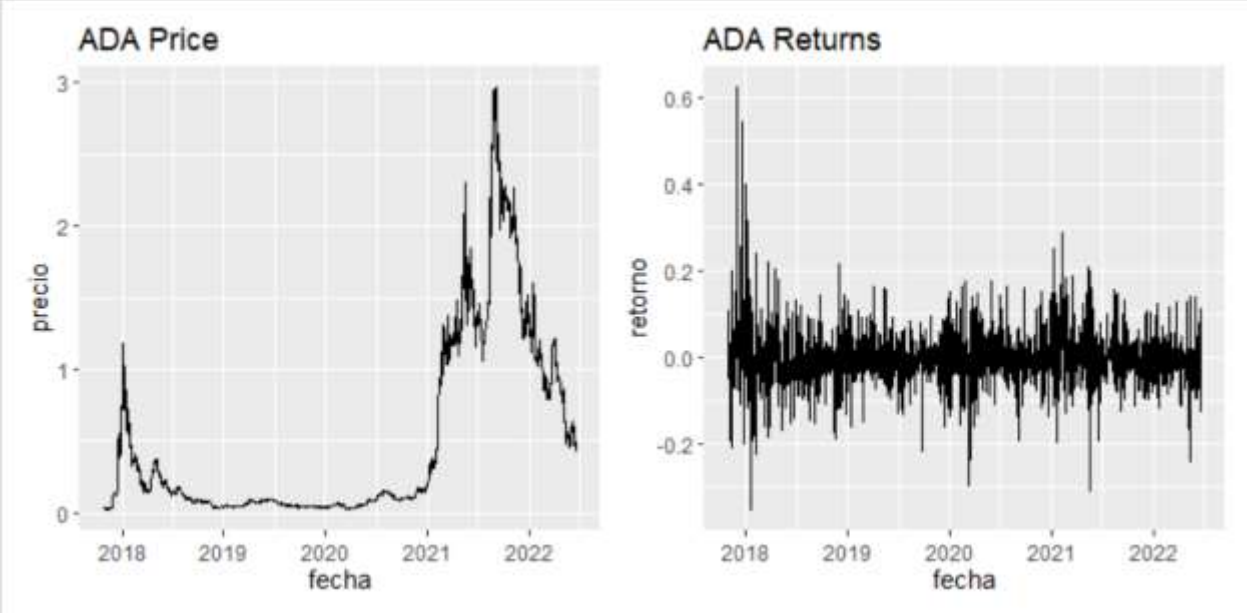
Graph 1 shows the evolution of the different currencies (expressed in MU), and the yields in logarithms of these currencies (hereinafter, yields). In this regard, an explosive evolution is observed in most coin prices, as they go from minimum monetary values to maximum values (occasionally in a period of 3 to 4 years). However, when observing the evolution of the yields, they present a reversion to the mean (to the zero value), which is interpreted that on average the currencies have yields equal to zero on average (see Graph 1).

**Graph 1. Evolution of Cryptocurrency prices and returns**

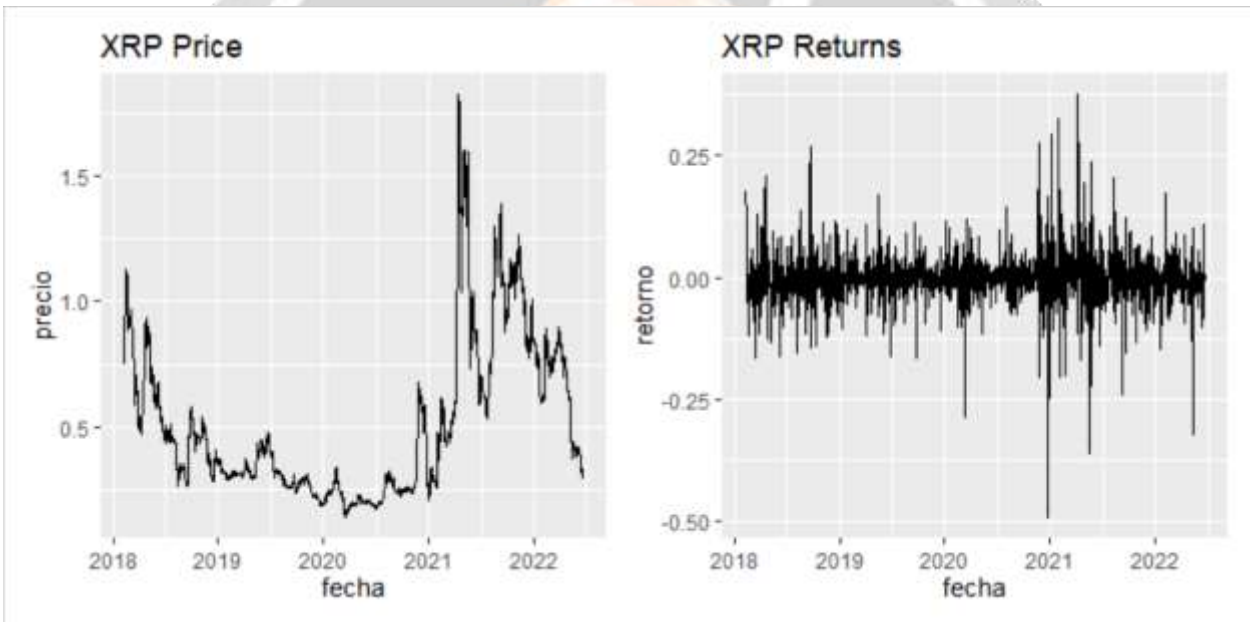



SOL Price
precio
250
200
150
100
50
Jul. 2021
Oct. 2021
Ene. 2022
Abr. 2022
fecha
SOL Returns
retorno
0.2
0.0
-0.2
-0.4
Jul. 2021
Oct. 2021
Ene. 2022
Abr. 2022
fecha

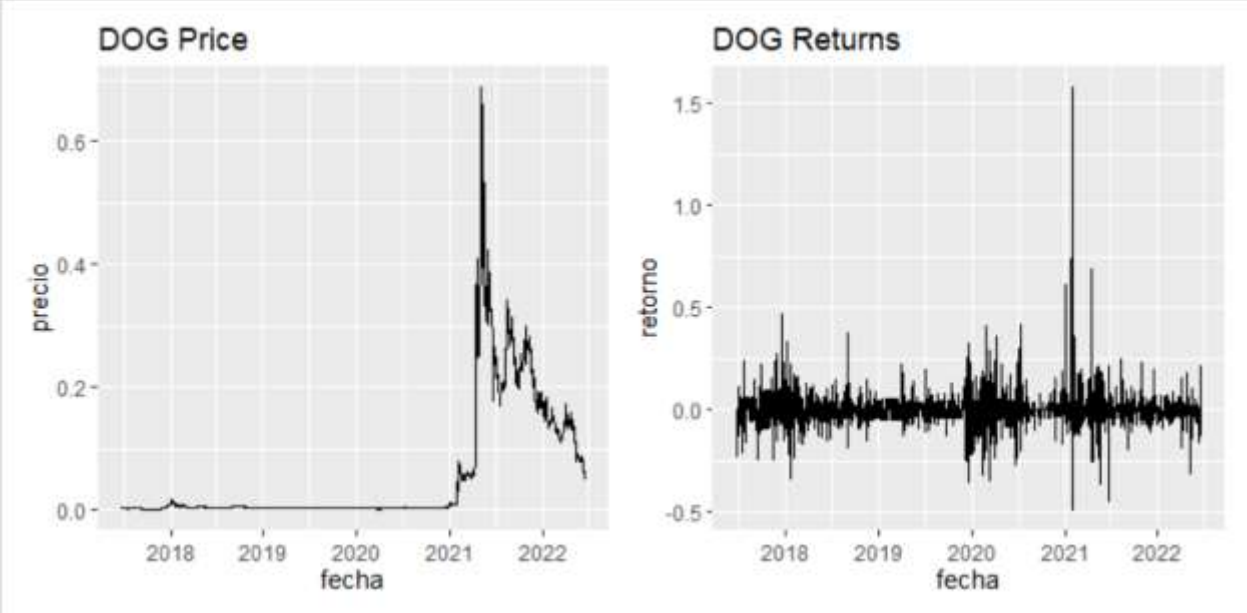

BNB Price
precio
600
400
200
0
2019
2020
2021
2022
fecha
BNB Returns
retorno
0.50
0.25
0.00
-0.25
2019
2020
2021
2022
fecha

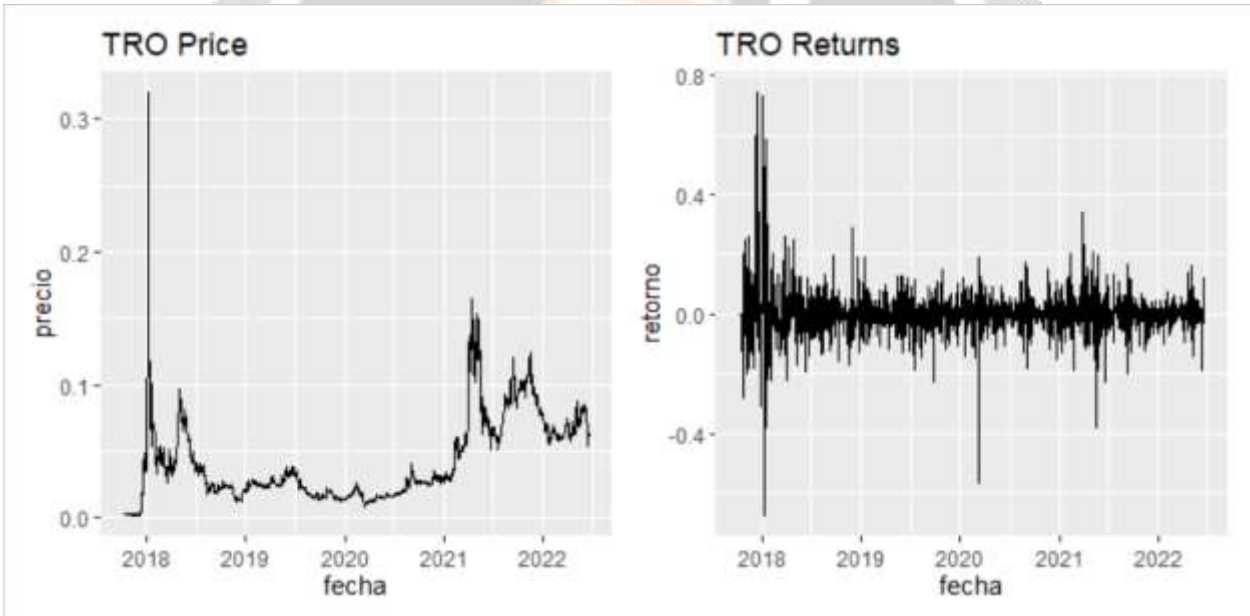

ADA Price
precio
fecha
2018
2019
2020
2021
2022
0
1
2
3

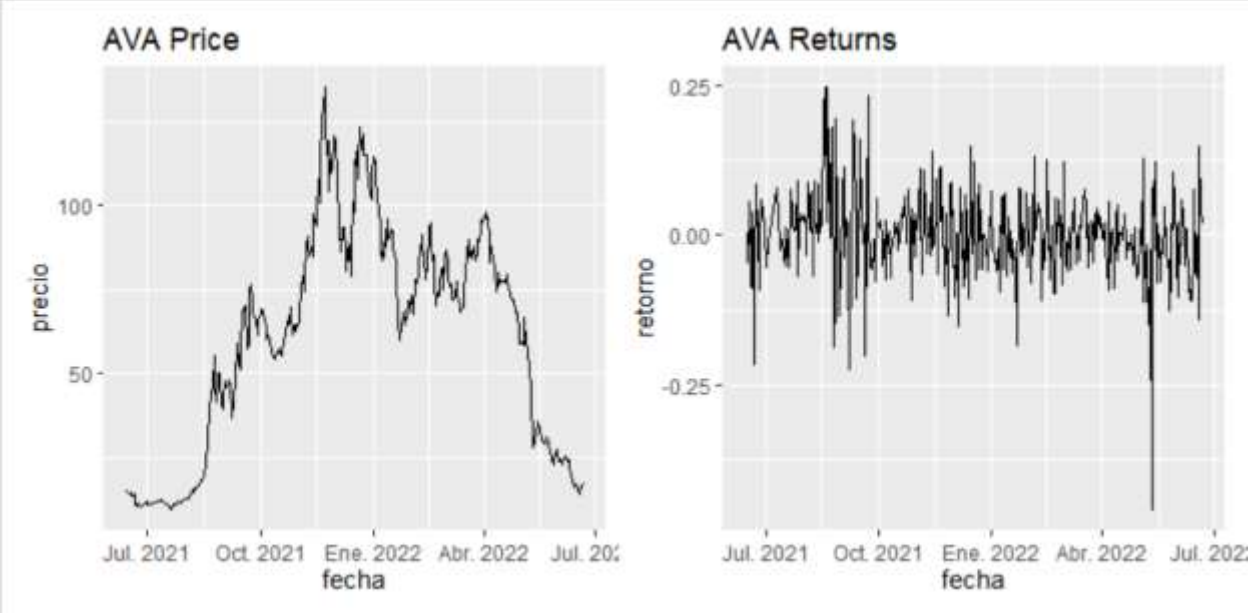

ADA Returns
retorno
fecha
2018
2019
2020
2021
2022
-0.2
0.0
0.2
0.4
0.6

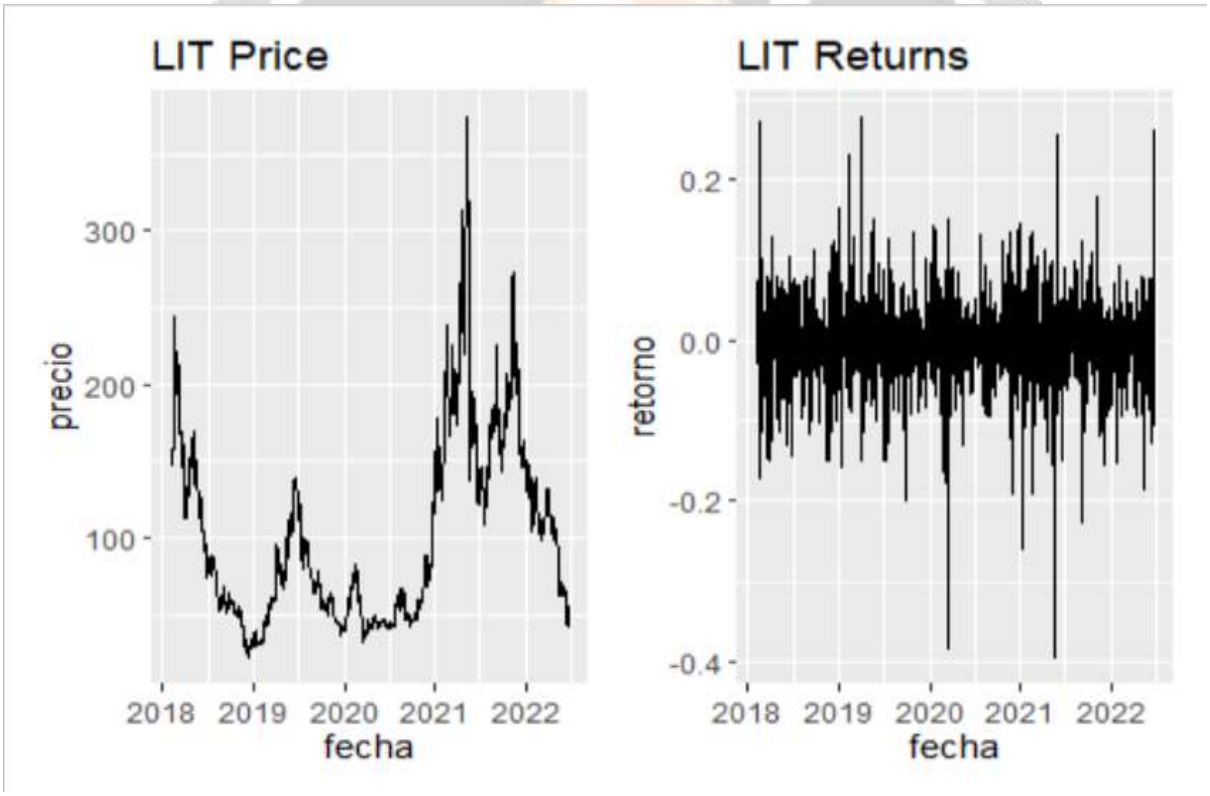

XRP Price
precio
fecha
2018
2019
2020
2021
2022
0.5
1.0
1.5


XRP Returns
retorno
fecha
2018
2019
2020
2021
2022
-0.50
-0.25
0.00
0.25


DOG Price
precio
0.6
0.4
0.2
0.0
2018
2019
2020
2021
2022
fecha
DOG Returns
retorno
1.5
1.0
0.5
0.0
-0.5
2018
2019
2020
2021
2022
fecha


TRO Price
precio
0.3
0.2
0.1
0.0
2018
2019
2020
2021
2022
fecha
TRO Returns
retorno
0.8
0.4
0.0
-0.4
2018
2019
2020
2021
2022
fecha




As can be seen in Table 1, on average the yields of the currencies have an average of zero; In addition, its median also coincides with said value; this could presume that the returns could behave like a normal distribution; however, some tests will be analyzed to evaluate this hypothesis.

**Chart -1 Yield Descriptive Statistics**

| Currency | Mean | Median | Max | Min | St. Dev. | Skewness | Kurtosis |
|---|---|---|---|---|---|---|---|
| Bitcoin | 0.001 | 0.001 | 0.293 | -0.317 | 0.043 | -0.156 | 5.423 |
| Ethereum | 0.000 | 0.000 | 0.275 | -0.405 | 0.051 | -0.560 | 5.760 |
| Solana | 0.000 | 0.000 | 0.227 | -0.399 | 0.067 | -0.295 | 4.086 |
| Binance | 0.003 | 0.002 | 0.531 | -0.411 | 0.059 | 0.242 | 9.759 |
| Cardano | 0.002 | 0.000 | 0.625 | -0.351 | 0.070 | 1.016 | 8.691 |
| XRP Ripple | -0.001 | -0.001 | 0.375 | -0.492 | 0.058 | -0.153 | 9.356 |
| Dogecoin | 0.002 | 0.000 | 1.576 | -0.489 | 0.090 | 3.472 | 55.510 |
| Tron | 0.002 | 0.000 | 0.743 | -0.675 | 0.078 | 1.356 | 22.104 |
| Avalance | 0.000 | 0.000 | 0.245 | -0.457 | 0.074 | -0.423 | 4.507 |
| Litecoin | -0.001 | -0.001 | 0.277 | -0.393 | 0.055 | -0.418 | 5.873 |

## 2. Normality tests

When analyzing the Jarque-Bera statistic, it is observed that these values are positive; In addition, when evaluating the null hypothesis of normality of the returns, it is observed that said hypothesis is rejected, so that the returns do not follow a normal distribution.

### 2.1 Unit Root Test

The unit root test allows evaluating whether or not a series is stationary, for which the Augmented Dickey-Fuller test is evaluated, which contrasts the null hypothesis of whether a series has a unit root (is not stationary). In addition, the KPSS test is evaluated, which, unlike Augmented Dickey-Fuller, tests the null hypothesis of whether a series is stationary.

When analyzing the Augmented Dickey-Fuller test, it is observed that the null hypothesis is rejected, concluding that the yields are stationary. However, when analyzing the KPSS test, it is observed that for the Ethereum, Solana and Avalance coins the null hypothesis is rejected; concluding that they are not stationary; while the returns of the other currencies are.

### 2.2 Autocorrelation test

Autocorrelation shows whether the current residuals are correlated with the same lagged residuals, bringing with it hypothesis testing problems due to non-stability of error variances. To evaluate the autocorrelation, the Ljung-Box and Ljung-Box^2 statistics are presented.

When evaluating the first test, it is determined that there is no presence of partial autocorrelation since the tests were rejected; however, when analyzing the second test, it is observed that there are signs of the presence of autocorrelation, unlike the returns of the Ethereum, Solana and Avalance coins.

Therefore, to model the presence of volatility groups, a GARCH model will be estimated, which includes residual errors within the model, as well as volatility.

**Chart -2. Contrasts of statistical tests**

| COIN | Jarque-Bera | KPSS | Augmented Dickey-Fuller | Ljung-Box (residuals) | Ljung-Box^2 (residuals) |
|---|---|---|---|---|---|
| Bitcoin | 2043*** | 0.172 | -11.2** | 0 | 12.14*** |
| Ethereum | 2144*** | 0.365* | -10.3** | 0.01 | 2.64 |
| Solana | 271*** | 0.617** | -7.1** | 0 | 0.05 |
| Binance | 6354*** | 0.23 | -9.3** | 0.13 | 46.0*** |
| Cardano | 5700*** | 0.145 | -11.3** | 0 | 36.36*** |
| XRP Ripple | 5083*** | 0.086 | -11.0** | 0 | 26.51*** |
| Dogecoin | 240872*** | 0.15 | -10.6** | 0.03 | 13.81*** |
| Tron | 35405*** | 0.117 | -12.6** | 0 | 52.23*** |
| Avalance | 317*** | 0.775** | -6.8** | 0.01 | 0.71 |
| Litecoin | 2232*** | 0.125 | -11.1** | 0 | 6.83*** |

**Note: The (*) denotes that the variables are significant at 10%, (**) at 5% and (***) at 1%.**

**Conditional heteroskedasticity autoregressive models**

**GARCH estimate (1,1)**

Next, the heteroscedasticity models for each of the currency yields are presented, following equation (1). In these models it can be seen that the coefficients are at least significant at 10%, although they are mostly so at 1%.

$$\sigma_t^2 = \omega + \alpha\epsilon_{t-1}^2 + \beta\sigma_{t-1}^2 \dots (1)$$

**Table 3. Estimation of GARCH (1,1) models (Part 1)**

| Model | Currency | Estimation | Coefficient | ee | Est. t | p-value | Significant at |
|---|---|---|---|---|---|---|---|
| 1 | Bitcoin | $\omega$ | 0.0002 | 0.0001 | 2.87 | 0.0041 | 1% |
| | | $\alpha$ | 0.1122 | 0.0279 | 4.02 | 0.0001 | 1% |
| | | $\beta$ | 0.8271 | 0.0389 | 21.27 | 0 | 1% |
| 2 | Ethereum | $\omega$ | 0.0002 | 0.0001 | 2.87 | 0.0041 | 1% |
| | | $\alpha$ | 0.1122 | 0.0279 | 4.02 | 0.0001 | 1% |
| | | $\beta$ | 0.8271 | 0.0389 | 21.27 | 0 | 1% |
| 3 | Solana | $\omega$ | 0 | 0 | 2.9 | 0.0037 | 1% |
| | | $\alpha$ | 0.0141 | 0.0075 | 1.89 | 0.0586 | 10% |
| | | $\beta$ | 0.9756 | 0.0082 | 119.15 | 0 | 1% |
| 4 | Binance | $\omega$ | 0.0003 | 0.0001 | 3.15 | 0.0017 | 1% |
| | | $\alpha$ | 0.1963 | 0.0464 | 4.23 | 0 | 1% |
| | | $\beta$ | 0.7545 | 0.0472 | 16 | 0 | 1% |
| 5 | Cardano | $\omega$ | 0.0002 | 0.0001 | 2.52 | 0.0118 | 5% |
| | | $\alpha$ | 0.1411 | 0.0356 | 3.97 | 0.0001 | 1% |
| | | $\beta$ | 0.8306 | 0.0408 | 20.38 | 0 | 1% |

**Table 4. Estimation of GARCH (1,1) models (Part 2)**

| Model | Currency | Estimation | Coefficient | ee | Est. t | p-value | Significant at |
|---|---|---|---|---|---|---|---|
| 6 | XRP Ripple | $\omega$ | 0.0002 | 0.0001 | 2.32 | 0.0202 | 5% |
| | | $\alpha$ | 0.2116 | 0.0572 | 3.7 | 0.0002 | 1% |
| | | $\beta$ | 0.7874 | 0.0534 | 14.75 | 0 | 1% |
| 7 | Dogecoin | $\omega$ | 0.0005 | 0.0001 | 3.76 | 0.0002 | 1% |
| | | $\alpha$ | 0.3897 | 0.062 | 6.28 | 0 | 1% |
| | | $\beta$ | 0.6093 | 0.0586 | 10.41 | 0 | 1% |
| 8 | Tron | $\omega$ | 0.0001 | 0 | 2.85 | 0.0043 | 1% |
| | | $\alpha$ | 0.1466 | 0.0293 | 5 | 0 | 1% |
| | | $\beta$ | 0.8524 | 0.0257 | 33.12 | 0 | 1% |
| 9 | Avalance | $\omega$ | 0.0004 | 0.0002 | 1.78 | 0.0744 | 10% |
| | | $\alpha$ | 0.1533 | 0.0607 | 2.53 | 0.0115 | 5% |
| | | $\beta$ | 0.7797 | 0.0739 | 10.56 | 0 | 1% |
| 10 | Litecoin | $\omega$ | 0.0001 | 0.0001 | 1.89 | 0.0588 | 10% |
| | | $\alpha$ | 0.0836 | 0.0241 | 3.47 | 0.0005 | 1% |
| | | $\beta$ | 0.8938 | 0.0318 | 28.15 | 0 | 1% |

**Stability of the coefficients**

To evaluate the stability of the coefficients of the models, the Nyblom test will be contrasted, which provides a means to test the structural change within a time series; that is, to verify the stability of the coefficients over time.

Table 4 presents the Nyblom stability tests for each estimated model. It is observed that not all the models present joint stability of the coefficients; Therefore, other types of models such as TGARCH could be evaluated to find stable coefficients; however, since most of the models are stable, the GARCH models will be considered to evaluate the volatility of the returns..

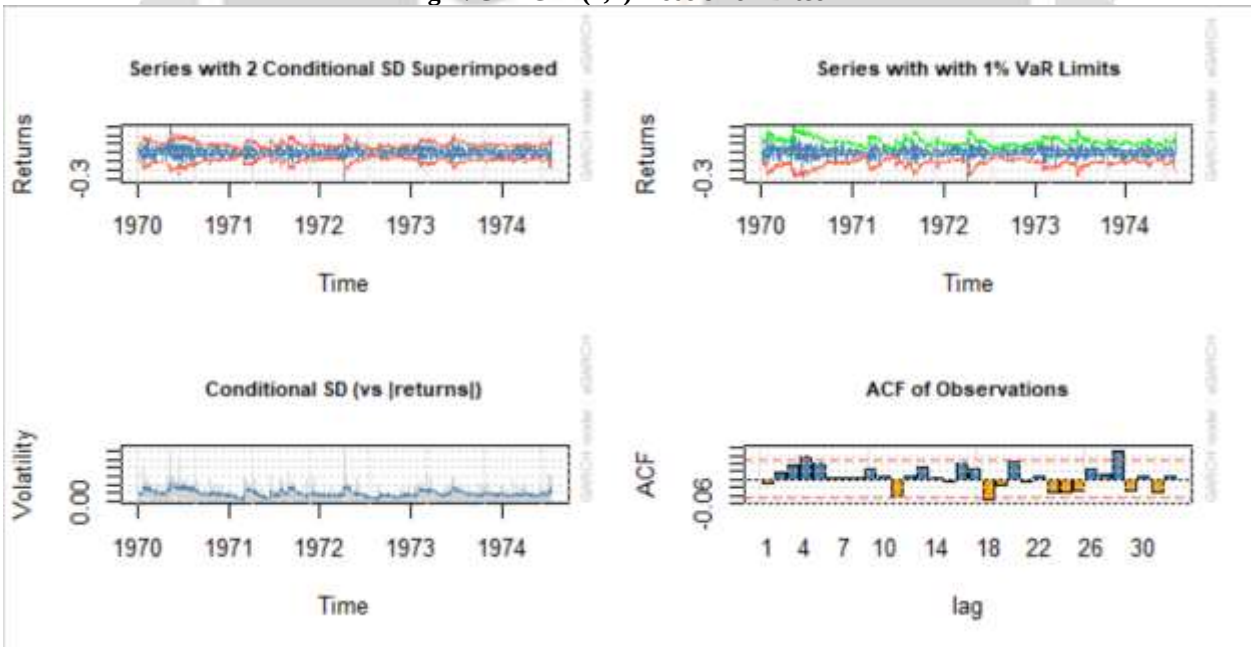
**Chart 4. Stability test of Nyblom coefficients**

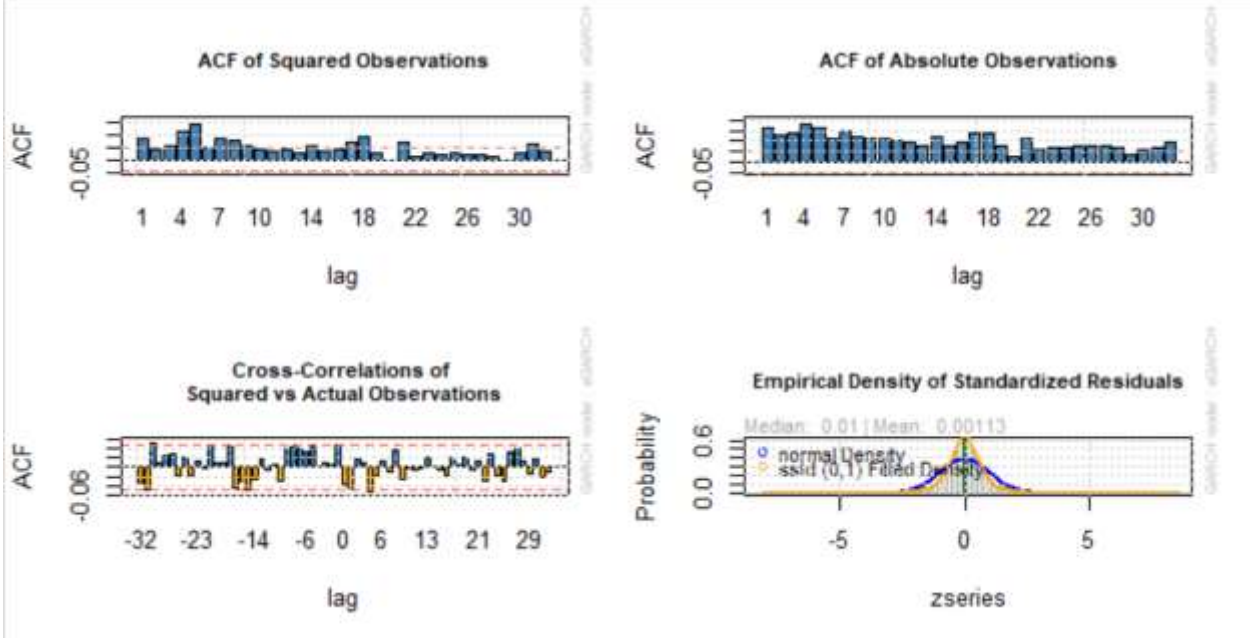
| Nyblom Test | Joint Statistic | 10% | 5% | 1% | Stability of coefficients |
|---|---|---|---|---|---|
| Bitcoin | 1.91 | 1.49 | 1.68 | 2.12 | No |
| Ethereum | 4.36 | 1.49 | 1.68 | 2.12 | Si |
| Solana | 1.91 | 1.49 | 1.68 | 2.12 | No |
| Binance | 1.01 | 1.49 | 1.68 | 2.12 | Si |
| Cardano | 1.00 | 1.49 | 1.68 | 2.12 | Si |
| XRP Ripple | 1.41 | 1.49 | 1.68 | 2.12 | Si |
| Dogecoin | 5.09 | 1.49 | 1.68 | 2.12 | No |
| Tron | 2.49 | 1.49 | 1.68 | 2.12 | No |
| Avalance | 1.09 | 1.49 | 1.68 | 2.12 | Si |
| Litecoin | 1.25 | 1.49 | 1.68 | 2.12 | Si |

**Model graphics**

Since the estimated models meet the individual significance test, they will be considered to model the volatilities of each currency. Next, through the graphs, the estimation of the volatilities is presented, as well as the autocorrelations of the errors and squared, observing that said values are found in the bands; so this would not be a problem in the estimated models.

**Fig 2. GARCH (1,1) models for Bitcoin**




ACF of Squared Observations
ACF
-0.05
1 4 7 10 14 18 22 26 30
lag
ACF of Absolute Observations
ACF
-0.05
1 4 7 10 14 18 22 26 30
lag
Cross-Correlations of
Squared vs Actual Observations
ACF
-0.06
-32 -23 -14 -6 0 6 13 21 29
lag
Empirical Density of Standardized Residuals
Probability
0.0 0.6
Median: 0.01 | Mean: 0.00113
normal Density
-5 0 5
zseries

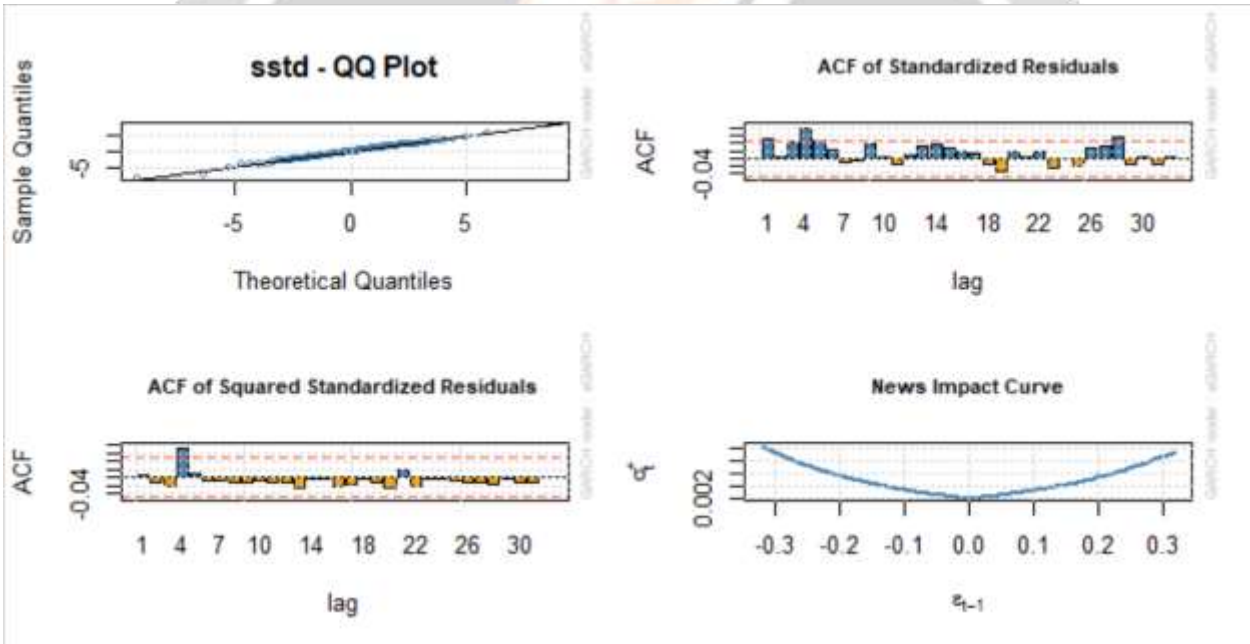

sstd - QQ Plot
Sample Quantiles
-5
-5 0 5
Theoretical Quantiles
ACF of Standardized Residuals
ACF
-0.04
1 4 7 10 14 18 22 26 30
lag
ACF of Squared Standardized Residuals
ACF
-0.04
1 4 7 10 14 18 22 26 30
lag
News Impact Curve
0.002
-0.3 -0.2 -0.1 0.0 0.1 0.2 0.3

**Fig 3. GARCH (1,1) models for ETHEREUM**

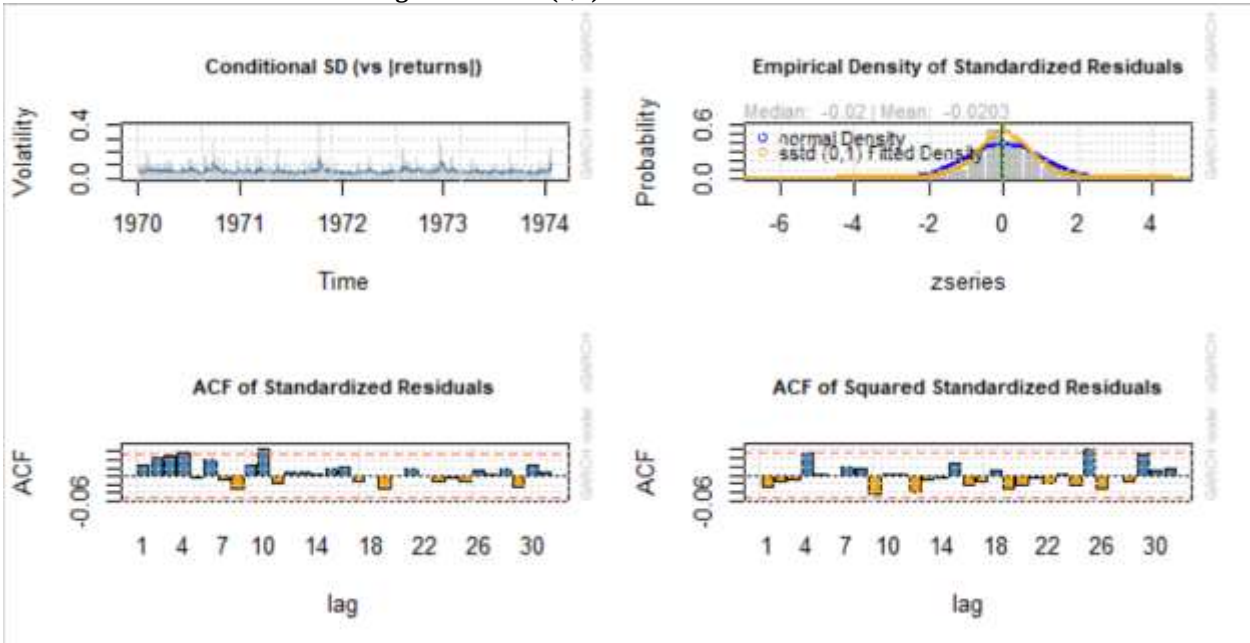



**Fig 4. GARCH (1,1) models for SOLANA**

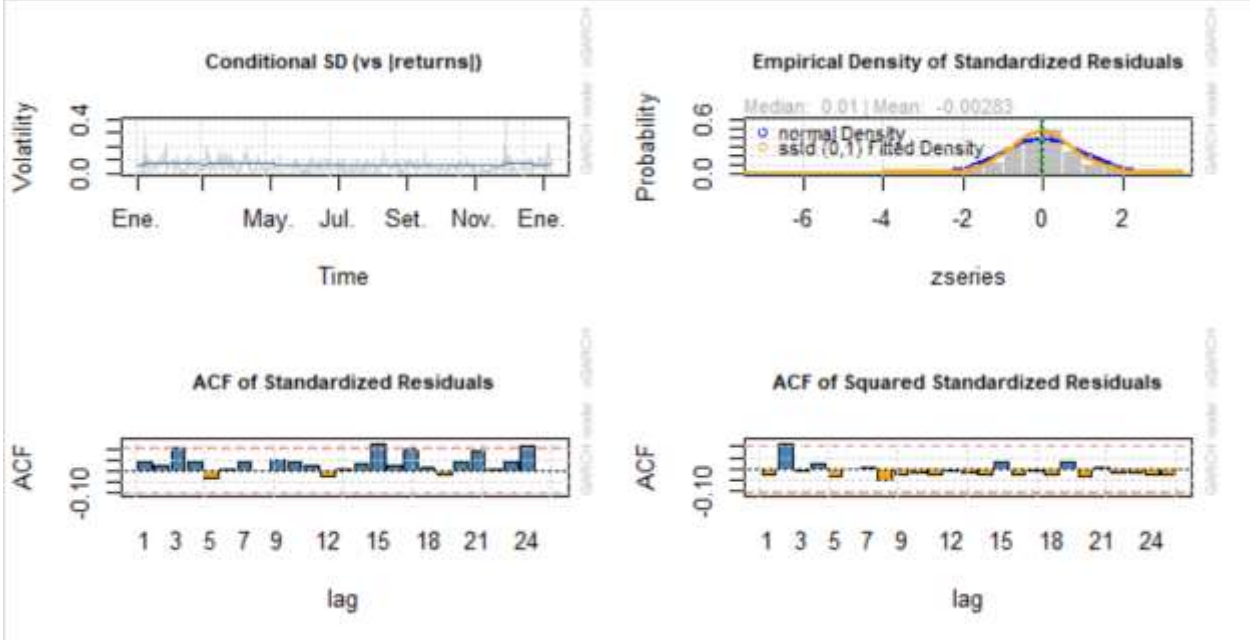


**Fig 4. GARCH (1,1) models for BINANCE**

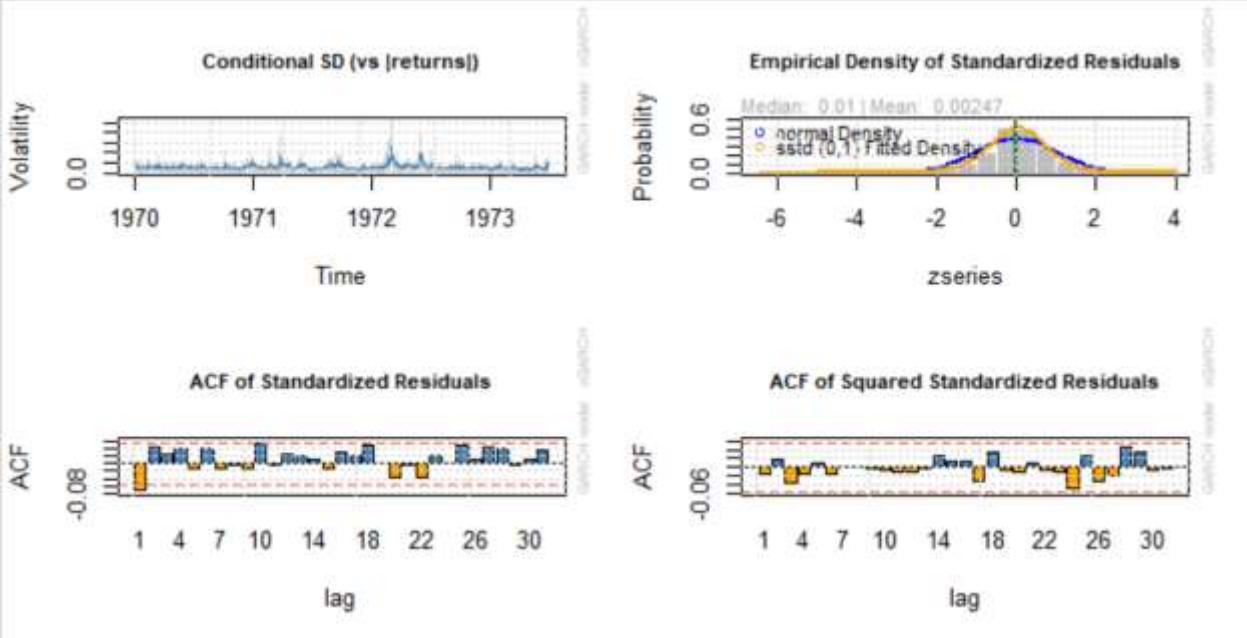



**Fig 5. GARCH (1,1) models for CARDANO**

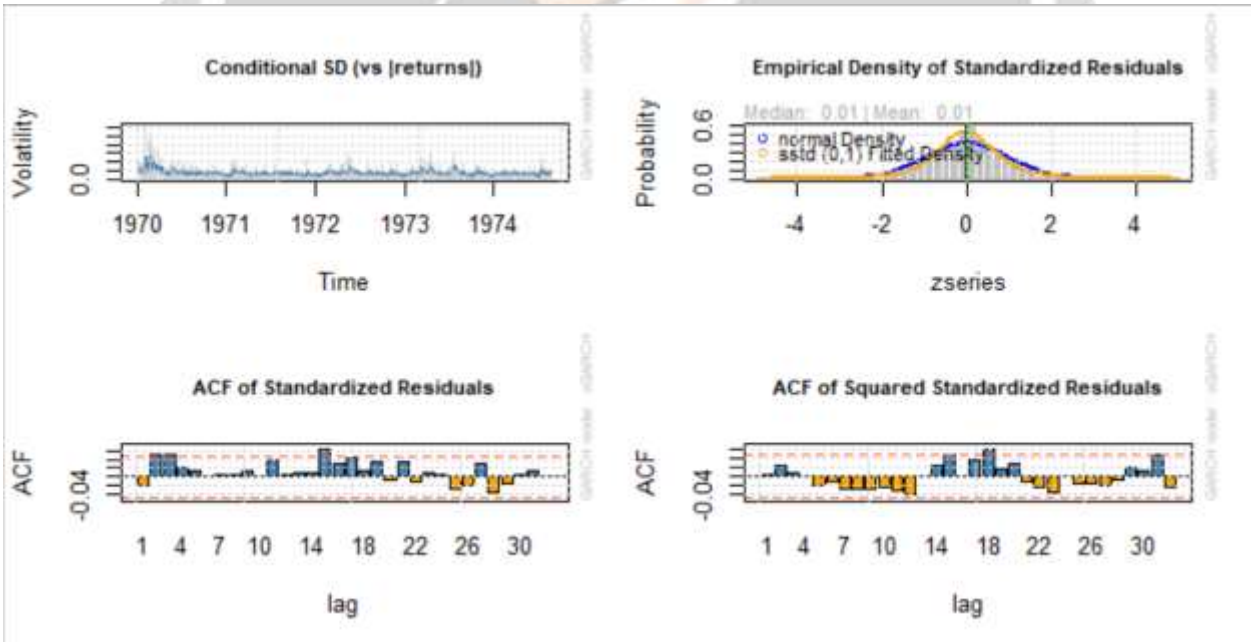


**Fig 6. GARCH (1,1) models for RIPPLE (XRP)**

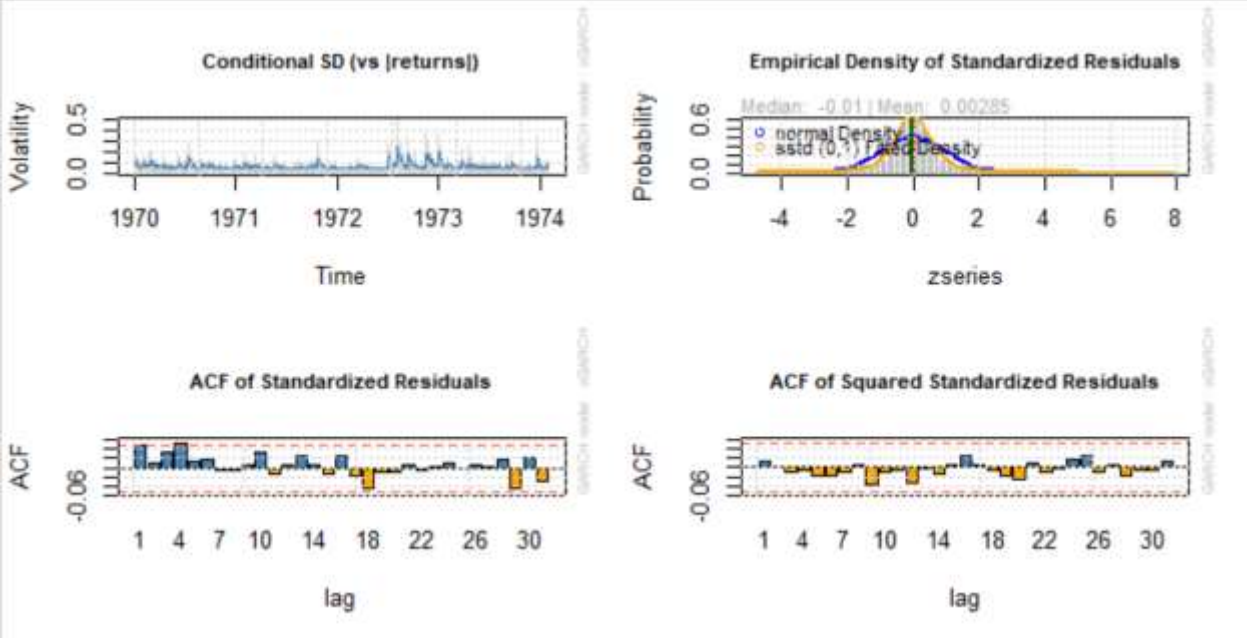



**Fig 7. GARCH (1,1) models for DOGECOIN**

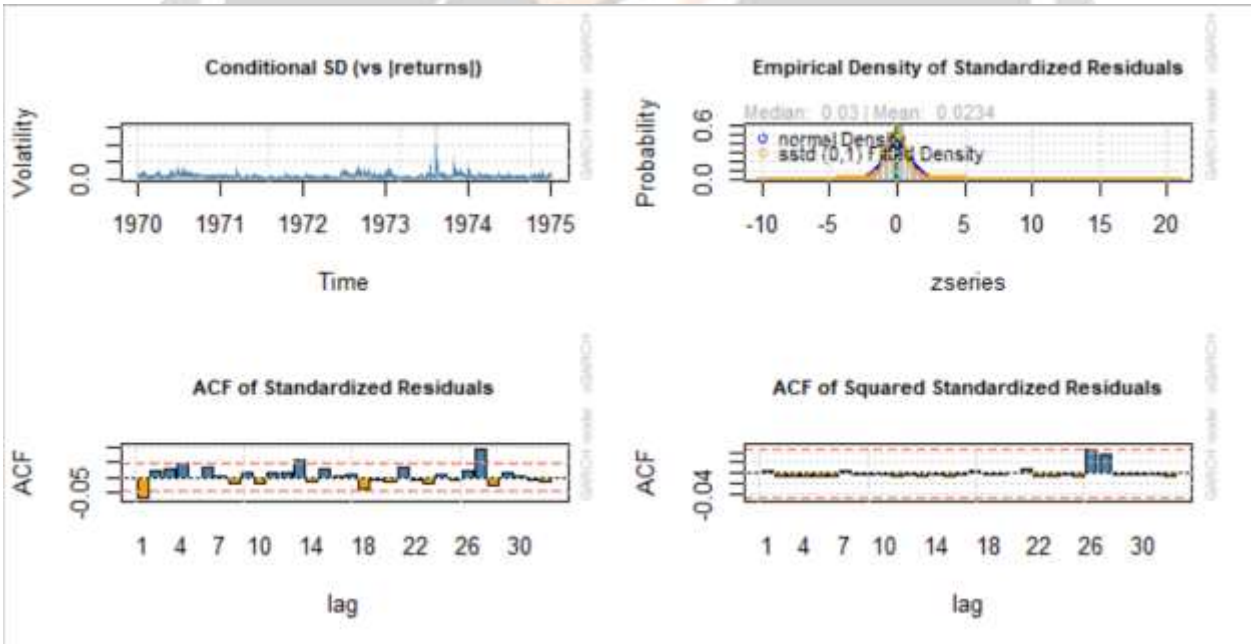


**Fig 8. GARCH (1,1) models for TRON**

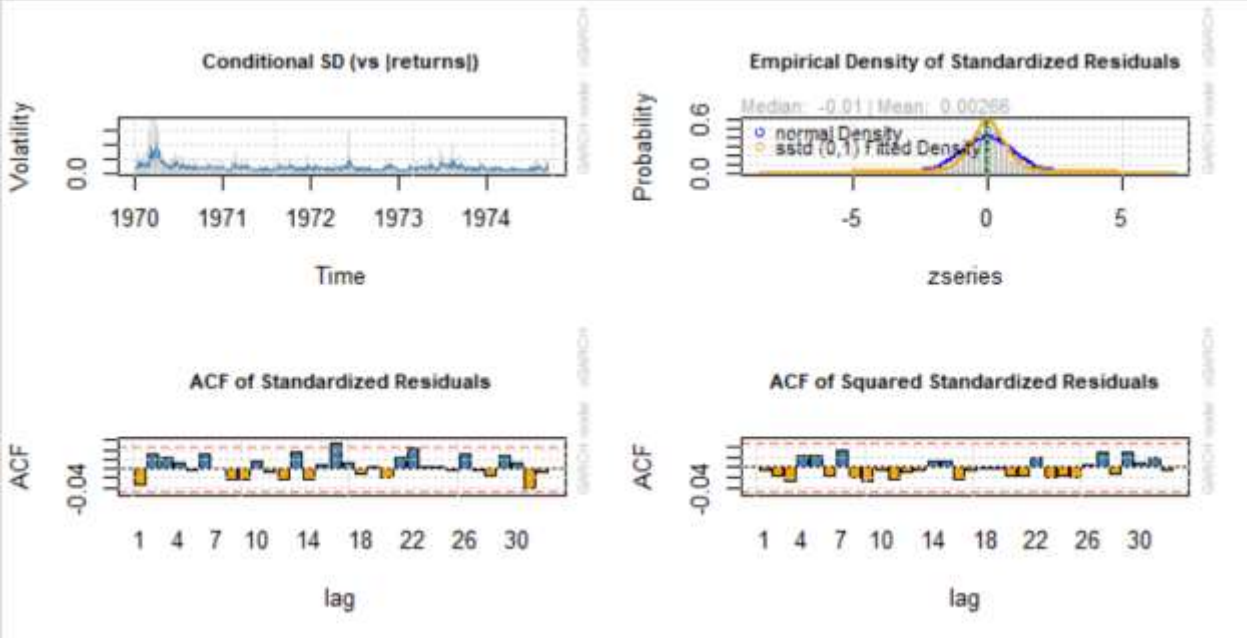



**Fig 9. GARCH (1,1) models for AVALANCE**

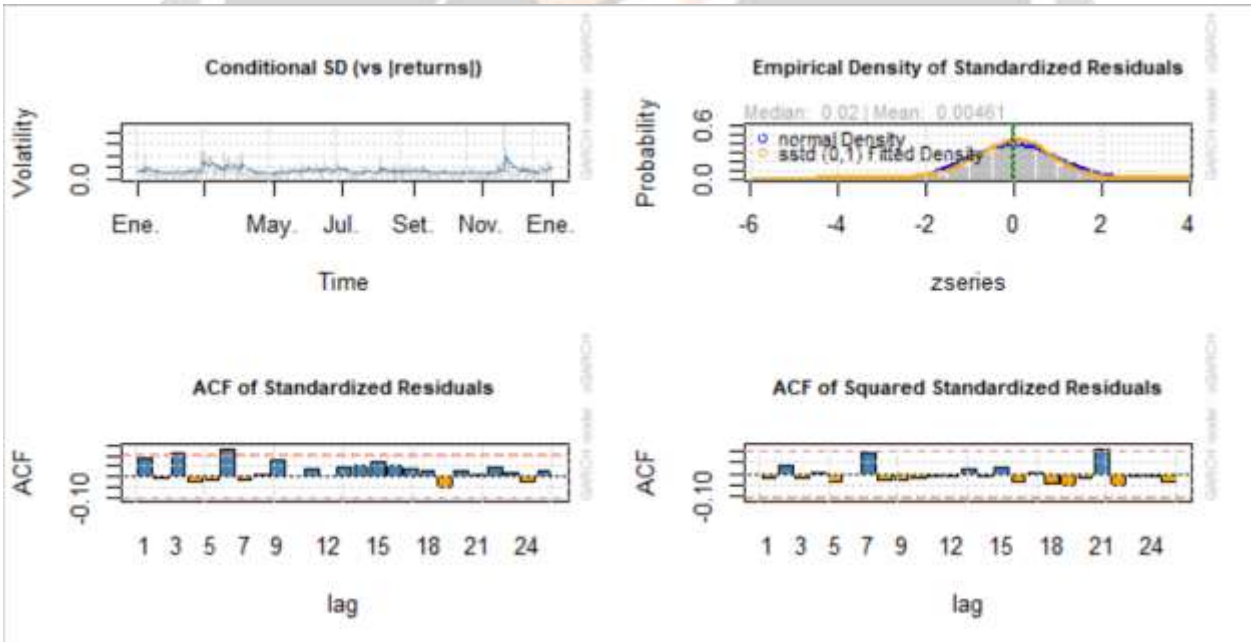
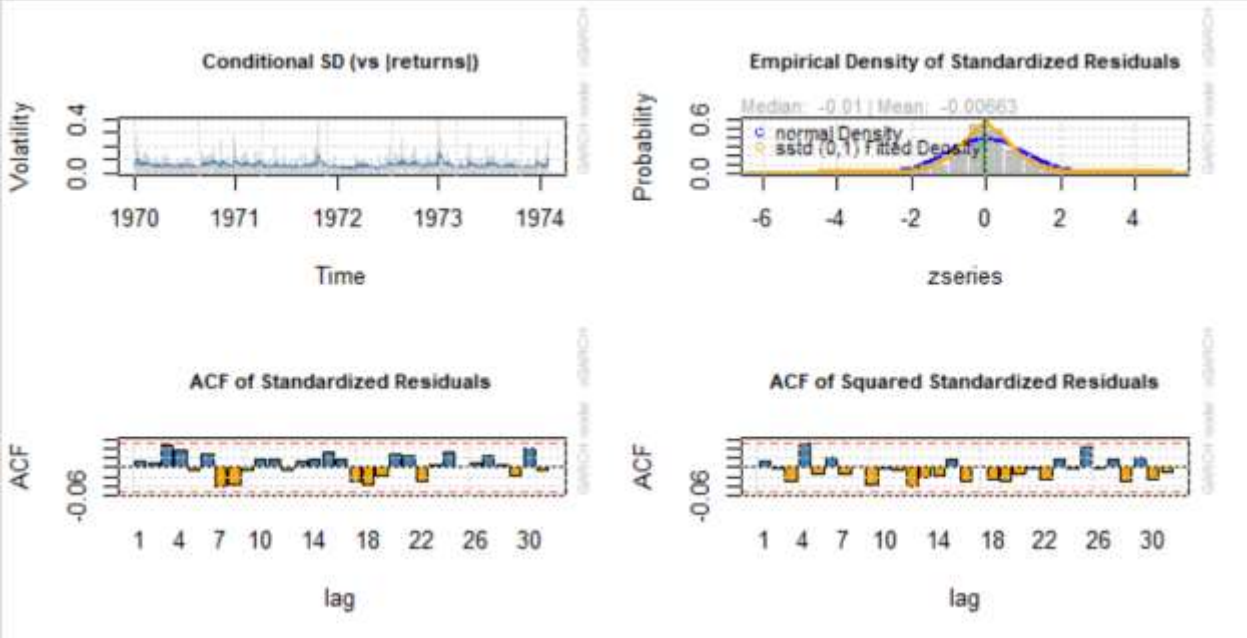


**Fig 10. GARCH (1,1) models for LITECOIN**




## 3. 5. CONCLUSION

Currency prices are mostly non-stationary series that agglomerate sets of volatilities; These characteristics make them candidates to model said volatilities through heteroscedastic autoregressive models.

Upon concluding that the yields do not have a unit root, which by default are understood to be stationary series, the autoregressive modeling of order one and heteroscedastics, also of order one, is taken as a strategy; since, although the series were stationary, they suffered from autocorrelation problems.

When estimating the heteroscedastic models, it was found that the coefficients were significant, which presupposes that the determinants are those that characterize the volatilities of the returns.

Some models do not meet the stability of coefficients according to Nyblom; however, most of the models do have coefficient stability, so these models would have a better effect when making forecasts.

## 5. ACKNOWLEDGEMENT

The authors can acknowledge to MSC. Rafael Caparó Coronado.

## 6. REFERENCES

[1]. Baur, D. G., & Dimpfl, T. (2018). Asymmetric volatility in cryptocurrencies. *Economics Letters*, *173*, 148-151.

[2]. Bolgün, K. E., & Akçay, M. B. Integrated Risk Measurement and Management Practices in the Emerging Turkish Financial Market

[3]. Coinmarketcap. (2022). Historical Data for Bitcoin. https://coinmarketcap.com/currencies/bitcoin/historical-data/

[4]. Chaim, P., & Laurini, M. P. (2019). Nonlinear dependence in cryptocurrency markets. *The North American Journal of Economics and Finance*, *48*, 32-47.

[5]. Dyhrberg, A. H. (2016). Bitcoin, gold and the dollar–A GARCH volatility analysis. *Finance Research Letters*, *16*, 85-92.

[6]. Gkillas, K., & Katsiampa, P. (2018). An application of extreme value theory to cryptocurrencies. Economics Letters, 164, 109-111.

## 7. APPENDICES

R code for GARCH analysis.

```
install.packages(c("readxl","quantmod","PerformanceAnalytics","rugarch","e1071","urca","tidyverse","xts","tseries
","patchwork"))

library(readxl)
library(quantmod)
library(PerformanceAnalytics)
library(rugarch)
library(e1071)
library(urca)
library(tidyverse)
library(xts)
library(tseries)
library(patchwork) # Aplicar graficos

cal_returns <- function(x) {
  x_t = x[1:(length(x)-1)]
  x_t_1 = x[2:length(x)]
  xt = log(x_t_1) - log(x_t)
  xt = append(c(NA), xt)
  return(xt)
}

stats_summary <- function(x) {
  mean_x = round(mean(x, na.rm = T),3)
  median_x = round(median(x, na.rm = T),3)
  max_x = round(max(x, na.rm = T),3)
  min_x = round(min(x, na.rm = T),3)
  sd_x = round(sd(x, na.rm = T),3)
  skewness_x = round(skewness(x, na.rm = T),3)
  kurtosis_x = round(kurtosis(x, na.rm = T),3)

  return(c(mean_x,median_x,max_x,min_x,
       sd_x,skewness_x,kurtosis_x))
}

test_arma <- function(resid_model){
  resid_model <- na.remove(resid_model)
  # Test
  jb_test <- jarque.bera.test(resid_model)
  adf_test <- adf.test(resid_model)
  kpss_test <- kpss.test(resid_model)
  LB_test <- Box.test(resid_model, lag = 1, type = "Ljung")
  LB2_test <- Box.test(resid_model^2, lag = 1, type = "Ljung")

  result <- data.frame("coeff" = c(round(jb_test$statistic,3), round(adf_test$statistic,3),
                                   round(kpss_test$statistic,3), round(LB_test$statistic,3),
                                   round(LB2_test$statistic,3)),
                       "p-value" = c(round(jb_test$p.value,3), round(adf_test$p.value,3),
                                     round(kpss_test$p.value,3), round(LB_test$p.value,3),
                                     round(LB2_test$p.value,3)))
  rownames(result) <- c("Jarque-Bera", "Augmented Dickey-Fuller", "KPSS",
                        "Ljung-Box (residuos)", "Ljung-Box (residuos^2)")
```

```
   return(result)
}
print (resid_model)
btc <- read_excel("c:\\cryptos.xlsx",sheet = "btc")
eth <- read_excel("c:\\cryptos.xlsx",sheet = "eth")
sol <- read_excel("c:\\cryptos.xlsx",sheet = "solana")
bnb <- read_excel("c:\\cryptos.xlsx",sheet = "binance")
ada <- read_excel("c:\\cryptos.xlsx",sheet = "cardano")
xrp  <- read_excel("c:\\cryptos.xlsx",sheet = "XRP")
dog <- read_excel("c:\\cryptos.xlsx",sheet = "DOGE")
tro <- read_excel("c:\\cryptos.xlsx",sheet  = "TRON")
ava <- read_excel("c:\\cryptos.xlsx",sheet = "AVAX")
lit <- read_excel("c:\\cryptos.xlsx",sheet = "LITECOIN")

names(btc) <- c("fecha", "precio", "1", "2", "3")
names(eth) <- c("fecha", "precio", "1", "2", "3", "4", "5")
names(sol) <- c("fecha", "precio", "1", "2", "3")
names(bnb) <- c("fecha", "precio", "1", "2", "3")
names(ada) <- c("fecha", "precio", "1", "2", "3")
names(xrp)  <- c("fecha", "precio", "1", "2", "3")
names(dog) <- c("fecha", "precio", "1", "2", "3")
names(tro) <- c("fecha", "precio", "1", "2", "3")
names(ava) <- c("fecha", "precio", "1", "2", "3")
names(lit) <- c("fecha", "precio", "1", "2", "3")

btc <- btc %>%
   select(fecha, precio) %>%
   mutate(fecha = as.character(fecha))

btc$fecha <- as.Date(btc$fecha, tryFormats = '%Y-%m-%d',tz  = "UTC")

btc <- btc %>%
   arrange(fecha) %>%
   mutate(retorno = cal_returns(precio))

eth <- eth %>%
   select(fecha, precio) %>%
   mutate(fecha = as.character(fecha))

eth$fecha <- as.Date(eth$fecha, tryFormats = '%Y-%m-%d',tz  = "UTC")

eth <- eth %>%
   arrange(fecha) %>%
   mutate(retorno = cal_returns(precio))

sol <- sol %>%
  select(fecha, precio) %>%
  mutate(fecha = as.character(fecha))

sol$fecha <- as.Date(sol$fecha, tryFormats = '%Y-%m-%d',tz  = "UTC")

sol <- sol %>%
  arrange(fecha) %>%
  mutate(retorno = cal_returns(precio))
```

```
bnb <- bnb %>%
  select(fecha, precio) %>%
  mutate(fecha = as.character(fecha))

bnb$fecha <- as.Date(bnb$fecha, tryFormats = '%Y-%m-%d',tz  = "UTC")

bnb <- bnb %>%
  arrange(fecha) %>%
  mutate(retorno = cal_returns(precio))

ada <- ada %>%
  select(fecha, precio) %>%
  mutate(fecha = as.character(fecha))

ada$fecha <- as.Date(ada$fecha, tryFormats = '%Y-%m-%d',tz  = "UTC")

ada <- ada %>%
  arrange(fecha) %>%
  mutate(retorno = cal_returns(precio))

xrp  <- xrp  %>%
  select(fecha, precio) %>%
  mutate(fecha = as.character(fecha))

xrp$fecha  <- as.Date(xrp$fecha,  tryFormats = '%Y-%m-%d',tz  = "UTC")

xrp  <- xrp  %>%
  arrange(fecha) %>%
  mutate(retorno = cal_returns(precio))

dog <- dog %>%
  select(fecha, precio) %>%
  mutate(fecha = as.character(fecha))

dog$fecha <- as.Date(dog$fecha, tryFormats = '%Y-%m-%d',tz  = "UTC")

dog <- dog %>%
  arrange(fecha) %>%
  mutate(retorno = cal_returns(precio))

tro <- tro %>%
  select(fecha, precio) %>%
  mutate(fecha = as.character(fecha))

tro$fecha <- as.Date(tro$fecha, tryFormats = '%Y-%m-%d',tz  = "UTC")

tro <- tro %>%
  arrange(fecha) %>%
  mutate(retorno = cal_returns(precio))

ava <- ava %>%
  select(fecha, precio) %>%
  mutate(fecha = as.character(fecha))

ava$fecha <- as.Date(ava$fecha, tryFormats = '%Y-%m-%d',tz  = "UTC")
```

```
ava <- ava %>%
  arrange(fecha) %>%
  mutate(retorno = cal_returns(precio))

lit <- lit %>%
  select(fecha, precio) %>%
  mutate(fecha = as.character(fecha))

lit$fecha <- as.Date(lit$fecha, tryFormats = '%Y-%m-%d',tz = "UTC")

lit <- lit %>%
  arrange(fecha) %>%
  mutate(retorno = cal_returns(precio))


# Graph ========================================================================
p_btc <- ggplot(btc, aes(x = fecha, y = precio)) +
    geom_line() +
    ggtitle("BTC Price")

p_btc_r <- ggplot(btc, aes(x = fecha, y = retorno)) +
    geom_line() +
    ggtitle("BTC Returns")

p_btc + p_btc_r

p_eth <- ggplot(eth, aes(x = fecha, y = precio)) +
    geom_line() +
    ggtitle("ETH Price")

p_eth_r <- ggplot(eth, aes(x = fecha, y = retorno)) +
    geom_line() +
    ggtitle("ETH Returns")

p_eth + p_eth_r

p_sol <- ggplot(sol, aes(x = fecha, y = precio)) +
  geom_line() +
  ggtitle("SOL Price")

p_sol_r <- ggplot(sol, aes(x = fecha, y = retorno)) +
  geom_line() +
  ggtitle("SOL Returns")

p_sol + p_sol_r

p_bnb <- ggplot(bnb, aes(x = fecha, y = precio)) +
  geom_line() +
  ggtitle("BNB Price")

p_bnb_r <- ggplot(bnb, aes(x = fecha, y = retorno)) +
  geom_line() +
  ggtitle("BNB Returns")

p_bnb + p_bnb_r
```

```
p_ada <- ggplot(ada, aes(x = fecha, y = precio)) +
  geom_line() +
  ggtitle("ADA Price")

p_ada_r <- ggplot(ada, aes(x = fecha, y = retorno)) +
  geom_line() +
  ggtitle("ADA Returns")

p_ada + p_ada_r

p_xrp <- ggplot(xrp, aes(x = fecha, y = precio)) +
  geom_line() +
  ggtitle("XRP Price")

p_xrp_r <- ggplot(xrp, aes(x = fecha, y = retorno)) +
  geom_line() +
  ggtitle("XRP Returns")

p_xrp + p_xrp_r

p_dog <- ggplot(dog, aes(x = fecha, y = precio)) +
  geom_line() +
  ggtitle("DOG Price")

p_dog_r <- ggplot(dog, aes(x = fecha, y = retorno)) +
  geom_line() +
  ggtitle("DOG Returns")

p_dog + p_dog_r

p_tro <- ggplot(tro, aes(x = fecha, y = precio)) +
  geom_line() +
  ggtitle("TRO Price")

p_tro_r <- ggplot(tro, aes(x = fecha, y = retorno)) +
  geom_line() +
  ggtitle("TRO Returns")

p_tro + p_tro_r

p_ava <- ggplot(ava, aes(x = fecha, y = precio)) +
  geom_line() +
  ggtitle("AVA Price")

p_ava_r <- ggplot(ava, aes(x = fecha, y = retorno)) +
  geom_line() +
  ggtitle("AVA Returns")

p_ava + p_ava_r

p_lit <- ggplot(lit, aes(x = fecha, y = precio)) +
  geom_line() +
  ggtitle("LIT Price")

p_lit_r <- ggplot(lit, aes(x = fecha, y = retorno)) +
  geom_line() +
```

```
  ggtitle("LIT Returns")

p_lit + p_lit_r

# Descriptive statistics ============================================================
btc_sum = stats_summary(btc$retorno)
eth_sum = stats_summary(eth$retorno)
sol_sum = stats_summary(sol$retorno)
bnb_sum = stats_summary(bnb$retorno)
ada_sum = stats_summary(ada$retorno)
xrp_sum = stats_summary(xrp$retorno)
dog_sum = stats_summary(dog$retorno)
tro_sum = stats_summary(tro$retorno)
ava_sum = stats_summary(ava$retorno)
lit_sum = stats_summary(lit$retorno)

table_1 = rbind(btc_sum,eth_sum,sol_sum,bnb_sum,ada_sum,xrp_sum,dog_sum,tro_sum,ava_sum,lit_sum)
colnames(table_1) = c("Mean", "Median", "Max", "Min", "SDev", "Skewsness", "Kurtosis")

print(table_1)

# AR model
ar_btc <- arima(na.omit(btc$retorno),  order = c(1,0,0))
ar_btc

ar_btc_summary <- test_arma(ar_btc$residuals)
names(ar_btc_summary) <- c("BTC_Coeff",  "BTC_pvalue")

ar_eth <- arima(na.omit(eth$retorno),  order = c(1,0,0))
ar_eth

ar_eth_summary <- test_arma(ar_eth$residuals)
names(ar_eth_summary) <- c("ETH_Coeff",  "ETH_pvalue")

ar_sol <- arima(na.omit(sol$retorno),  order = c(1,0,0))
ar_sol

ar_sol_summary <- test_arma(ar_sol$residuals)
names(ar_sol_summary) <- c("SOL_Coeff",  "SOL_pvalue")

ar_bnb <- arima(na.omit(bnb$retorno),  order = c(1,0,0))
ar_bnb

ar_bnb_summary <- test_arma(ar_bnb$residuals)
names(ar_bnb_summary) <- c("BNB_Coeff",  "BNB_pvalue")

ar_ada <- arima(na.omit(ada$retorno),   order = c(1,0,0))
ar_ada

ar_ada_summary <- test_arma(ar_ada$residuals)
names(ar_ada_summary) <- c("ADA_Coeff", "ADA_pvalue")

ar_xrp <- arima(na.omit(xrp$retorno),  order = c(1,0,0))
ar_xrp

ar_xrp_summary <- test_arma(ar_xrp$residuals)
```

```
names(ar_xrp_summary) <- c("XRP_Coeff", "XRP_pvalue")

ar_dog <- arima(na.omit(dog$retorno), order = c(1,0,0))
ar_dog

ar_dog_summary <- test_arma(ar_dog$residuals)
names(ar_dog_summary) <- c("DOG_Coeff", "DOG_pvalue")

ar_tro <- arima(na.omit(tro$retorno), order = c(1,0,0))
ar_tro

ar_tro_summary <- test_arma(ar_tro$residuals)
names(ar_tro_summary) <- c("TRO_Coeff", "TRO_pvalue")

ar_ava <- arima(na.omit(ava$retorno), order = c(1,0,0))
ar_ava

ar_ava_summary <- test_arma(ar_ava$residuals)
names(ar_ava_summary) <- c("AVA_Coeff", "AVA_pvalue")

ar_lit <- arima(na.omit(lit$retorno), order = c(1,0,0))
ar_lit

ar_lit_summary <- test_arma(ar_lit$residuals)
names(ar_lit_summary) <- c("LIT_Coeff", "LIT_pvalue")


# Table ============================================================
ar_test = cbind(ar_btc_summary, ar_eth_summary, ar_sol_summary, ar_bnb_summary, ar_ada_summary,
ar_xrp_summary, ar_dog_summary, ar_tro_summary, ar_ava_summary, ar_lit_summary)

print(ar_test)

# GARCH Models
garch_spec = ugarchspec(mean.model = list(armaOrder = c(0, 0)),
                        variance.model = list(model = "sGARCH",garchOrder = c(1,1)),
                        distribution.model = "sstd")
garch_btc = ugarchfit(data = na.remove(btc$retorno), spec = garch_spec, out.sample = 0)

garch_eth = ugarchfit(data = na.remove(eth$retorno), spec = garch_spec, out.sample = 0)

garch_sol = ugarchfit(data = na.remove(sol$retorno), spec = garch_spec, out.sample = 0)

garch_bnb = ugarchfit(data = na.remove(bnb$retorno), spec = garch_spec, out.sample = 0)

garch_ada = ugarchfit(data = na.remove(ada$retorno), spec = garch_spec, out.sample = 0)

garch_xrp = ugarchfit(data = na.remove(xrp$retorno), spec = garch_spec, out.sample = 0)

garch_dog = ugarchfit(data = na.remove(dog$retorno), spec = garch_spec, out.sample = 0)

garch_tro = ugarchfit(data = na.remove(tro$retorno), spec = garch_spec, out.sample = 0)

garch_ava = ugarchfit(data = na.remove(ava$retorno), spec = garch_spec, out.sample = 0)

garch_lit = ugarchfit(data = na.remove(lit$retorno), spec = garch_spec, out.sample = 0)
```

```
print(garch_btc)

print(garch_eth)

print(garch_sol)

print(garch_bnb)

print(garch_ada)

print(garch_xrp)

print(garch_dog)

print(garch_tro)

print(garch_ava)

print(garch_lit)

egarch_spec = ugarchspec(mean.model = list(armaOrder  = c(0, 0)),
                         variance.model  = list(model  = "eGARCH", garchOrder = c(1,1)),
                         distribution.model = "sstd")
egarch_btc = ugarchfit(data = na.remove(btc$retorno), spec = egarch_spec, out.sample = 0)

egarch_eth = ugarchfit(data = na.remove(eth$retorno), spec = egarch_spec, out.sample = 0)

egarch_sol = ugarchfit(data = na.remove(sol$retorno), spec = egarch_spec, out.sample = 0)

egarch_bnb = ugarchfit(data = na.remove(bnb$retorno), spec = egarch_spec, out.sample = 0)

egarch_ada = ugarchfit(data = na.remove(ada$retorno), spec = egarch_spec, out.sample = 0)

egarch_xrp = ugarchfit(data = na.remove(xrp$retorno),  spec = egarch_spec, out.sample = 0)

egarch_dog = ugarchfit(data = na.remove(dog$retorno), spec = egarch_spec, out.sample = 0)

egarch_tro = ugarchfit(data = na.remove(tro$retorno), spec = egarch_spec, out.sample = 0)

egarch_ava = ugarchfit(data = na.remove(ava$retorno), spec = egarch_spec, out.sample = 0)

egarch_lit = ugarchfit(data = na.remove(lit$retorno), spec = egarch_spec, out.sample = 0)


# Criterios de información
table_ic = data.frame("BTC.GARCH"  = infocriteria(garch_btc), "BTC.EGARCH"  = infocriteria(egarch_btc),
                      "ETH.GARCH"  = infocriteria(garch_eth), "ETH.EGARCH"  = infocriteria(egarch_eth),
                      "SOL.GARCH"  = infocriteria(garch_eth), "SOL.EGARCH"  = infocriteria(egarch_eth),
                      "BNB.GARCH"  = infocriteria(garch_eth), "BNB.EGARCH"  = infocriteria(egarch_eth),
                      "ADA.GARCH"  = infocriteria(garch_eth), "ADA.EGARCH"  = infocriteria(egarch_eth),
                      "XRP.GARCH"  = infocriteria(garch_eth), "XRP.EGARCH"  = infocriteria(egarch_eth),
                      "DOG.GARCH"  = infocriteria(garch_eth), "DOG.EGARCH"  = infocriteria(egarch_eth),
                      "TRO.GARCH"  = infocriteria(garch_eth), "TRO.EGARCH"  = infocriteria(egarch_eth),
                      "AVA.GARCH"  = infocriteria(garch_eth), "AVA.EGARCH"  = infocriteria(egarch_eth),
                      "LIT.GARCH"  = infocriteria(garch_eth), "LIT.EGARCH"  = infocriteria(egarch_eth))
```

```
names(table_ic) = c("BTC.GARCH", "BTC.EGARCH", "ETH.GARCH", "ETH.EGARCH","SOL.GARCH",
"SOL.EGARCH","BNB.GARCH", "BNB.EGARCH","ADA.GARCH", "ADA.EGARCH","XRP.GARCH",
"XRP.EGARCH","DOG.GARCH", "DOG.EGARCH","TRO.GARCH", "TRO.EGARCH","AVA.GARCH",
"AVA.EGARCH","LIT.GARCH", "LIT.EGARCH")

# Coef GARCH(1,1) Ether
garch_eth@fit$matcoef[c("omega", "alpha1", "beta1"),]

# Coef EGARCH(1,1) Bitcoin
egarch_btc@fit$matcoef[c("omega", "alpha1", "beta1", "gamma1"),]

# Coef GARCH(1,1) Solana
garch_sol@fit$matcoef[c("omega", "alpha1", "beta1"),]

# Coef GARCH(1,1) Binance
garch_bnb@fit$matcoef[c("omega", "alpha1", "beta1"),]

# Coef GARCH(1,1) Cardano
garch_ada@fit$matcoef[c("omega", "alpha1", "beta1"),]

# Coef GARCH(1,1) XRP
garch_xrp@fit$matcoef[c("omega", "alpha1", "beta1"),]

# Coef GARCH(1,1) Dogecoin
garch_dog@fit$matcoef[c("omega", "alpha1", "beta1"),]

# Coef GARCH(1,1) Tron
garch_tro@fit$matcoef[c("omega", "alpha1", "beta1"),]

# Coef GARCH(1,1) Avalanche
garch_ava@fit$matcoef[c("omega", "alpha1", "beta1"),]

# Coef GARCH(1,1) Litecoin
garch_lit@fit$matcoef[c("omega", "alpha1", "beta1"),]

# Graph EGARCH
par(mfrow=c(2,2))
plot(egarch_btc, which = 1)
plot(egarch_btc, which = 2)
plot(egarch_btc, which = 3)
plot(egarch_btc, which = 4)
plot(egarch_btc, which = 5)
plot(egarch_btc, which = 6)
plot(egarch_btc, which = 7)
plot(egarch_btc, which = 8)
plot(egarch_btc, which = 9)
plot(egarch_btc, which = 10)
plot(egarch_btc, which = 11)
plot(egarch_btc, which = 12)

par(mfrow=c(2,2))
plot(garch_eth, which = 3)
plot(garch_eth, which = 8)
plot(garch_eth, which = 10)
plot(garch_eth, which = 11)
```

```
par(mfrow=c(2,2))
plot(garch_sol, which = 3)
plot(garch_sol, which = 8)
plot(garch_sol, which = 10)
plot(garch_sol, which = 11)

par(mfrow=c(2,2))
plot(garch_bnb, which = 3)
plot(garch_bnb, which = 8)
plot(garch_bnb, which = 10)
plot(garch_bnb, which = 11)

par(mfrow=c(2,2))
plot(garch_ada, which = 3)
plot(garch_ada, which = 8)
plot(garch_ada, which = 10)
plot(garch_ada, which = 11)

par(mfrow=c(2,2))
plot(garch_xrp, which = 3)
plot(garch_xrp, which = 8)
plot(garch_xrp, which = 10)
plot(garch_xrp, which = 11)

par(mfrow=c(2,2))
plot(garch_dog, which = 3)
plot(garch_dog, which = 8)
plot(garch_dog, which = 10)
plot(garch_dog, which = 11)

par(mfrow=c(2,2))
plot(garch_tro, which = 3)
plot(garch_tro, which = 8)
plot(garch_tro, which = 10)
plot(garch_tro, which = 11)

par(mfrow=c(2,2))
plot(garch_ava, which = 3)
plot(garch_ava, which = 8)
plot(garch_ava, which = 10)
plot(garch_ava, which = 11)

par(mfrow=c(2,2))
plot(garch_lit, which = 3)
plot(garch_lit, which = 8)
plot(garch_lit, which = 10)
plot(garch_lit, which = 11)
par(mfrow=c(1,1))
```