COURSEWORK

BRUNEL UNIVERSITY LONDON

DEPARTMENT OF MATHEMATICS

Time Series Modelling

MA5629

Author:
Jean Charles Kouame' (CID: 2140558)

Date: April 23, 2022

1 Introduction

Using the function **get.hist.quote** in the **tseries** library, extract a subset with the size of around 4000, from daily adjusted prices on the FTSE250 index (instrument FTMC) ranging from 1st January 2000 to 15th February 2021. In your report, you will discuss the modelling of the log-returns calculated from these data.

1.1 Import data

```
# Selecting a time interval that guarantees 4000 observations
start <- "2005-03-18"
end <- "2021-01-16"

# Read data in ftmc
ftmc <- get.hist.quote(instrument = "^FTMC", start, end,
                quote = "Adjusted",
                provider = c("yahoo"), method = NULL,
                origin = "1899-12-30", compression = "d",
                retclass = c("zoo", "ts"))
ftmc.c <- na.omit(ftmc)

# Plot time series
plot(ftmc.c, ylab="Value", xlab="Time", main="FTSE250")
grid(nx = 5, ny = 10, lty = 2, col = "gray", lwd = 1)
```

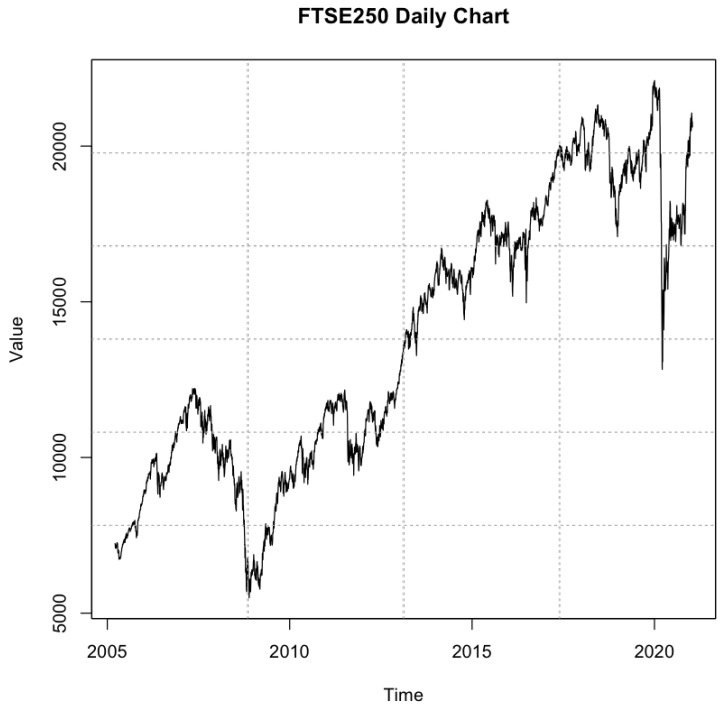


Figure 1: Time series plot

2 Discuss Stationarity and the transformations to achieve it

2.1 Stationarity in the data

Stationarity is defined as the probability distribution of a sequence of n observations does not depend on their time origin.

A time series Xt is (weakly) stationary if the mean and covariance functions do not change over time:

- the mean value function $\mu_x(t)$ is independent of t
- the covariance function $\gamma_x(t+h,t)$ is independent of t for each h

Strict stationarity is when the joint distribution of any set of $X_{t1}, X_{t2}, ...$ is unchanged if the times $t_1, t_2, ...$ are shifted by an amount s. This one is a very strong assumption because it requires that "all aspects" of behavior be constant in time.

Generally, we can assume weak stationarity.

Many financial time series, for example, stock prices, do not appear stationary, but the changes in these time series do appear stationary and can be modeled as stationary processes. For this reason, stationary time series models are far more applicable than they might appear.

2.2 Log transformation

Finance data tends to incorporate many potential multiplicative effects (e.g. inflation or interest), and distributions tend to be skewed, in some cases close to lognormal.
The log transformation is one of the most useful transformations finance. It is used as a transformation to normality and as a variance stabilizing transformation. Transformations such as logarithms have the following properties:

- Symmetry: positive and negative returns of equal magnitude cancel each other out
- Time additivity: the compound return over **n** periods is merely the difference in log between initial and final period $r = \log(p_f) - \log(p_0)$

$$R_t = \frac{P_t - P_{t-1}}{P_{t-1}} \tag{1}$$

$$R_t = \frac{P_t}{P_{t-1}} - 1 \tag{2}$$

$$R_t + 1 = \frac{P_t}{P_{t-1}} \tag{3}$$

$$\log(R_t + 1) = \log(\frac{P_t}{P_{t-1}}) \tag{4}$$

$$r_t = \log(P_t) - \log(P_{t-1}) \tag{5}$$

- Suitable for economic data: on logaritmic scale a change between two values is perceived on the basis of the ratio of the two values. Because of this reason plotting data on logaritmic scale will show different data points on the same level if the had the same percentage change.

2.3 Differencing

Differencing can help stabilise the mean of a time series by removing changes in the level of a time series. If we notice the plot in **fig.1** of the imported **raw data** we can see a change in the mean and variance of the time series, meaning that the series is non-stationary.

On the contrary a stationary time series is one whose properties do not depend on the time at which the series is observed. As we cited in the previous paragraph a time series has the following properties:

- mean does not depend on time
- variance is constant

In general, a stationary time series will have no predictable patterns in the long-term. Time plots will show the series to be roughly horizontal (although some cyclic behaviour is possible), with constant variance.

2.3.1 First-order differencing

A way to make a time series stationary it to compute the differences between two consecutive observations $y\prime_t = y_t - y_{t-1}$. This is known as **differencing**. It is possible to implement differencing in R like showed below:

```
# Create log returns
ftmc.log <- log(as.numeric(ftmc.c[,1]))

# First order differencing
ftmc.diff <- diff(ftmc.log)

# Plot first order differencing
plot(ftmc.diff, type="l", main="First-order differencing", ylab="return",
    xlab="period", ylim=c(-0.10,0.10))
grid(nx = 0, ny = 6, lty = 2, col = "black", lwd = 0.5)
```

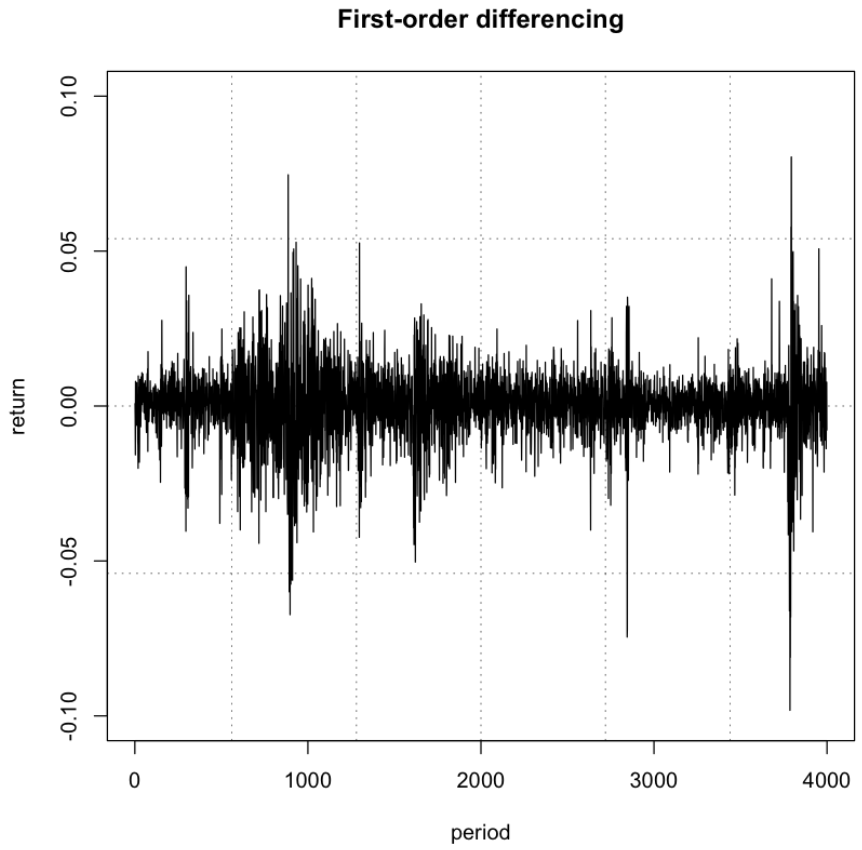


Figure 2: First-order differencing

2.3.2 Second-order differencing

Occasionally the differenced data will not appear to be stationary and it may be necessary to difference the data a second time to obtain a stationary series[2]:

```
# Second order differencing
ftmc.diff2 <- diff(ftmc.log, differences = 2)

# Plot second order differencing
plot(ftmc.diff2, type="l", main="Second-order differencing", ylab="return",
    xlab="period", ylim=c(-0.10,0.10))
grid(nx = 0, ny = 6, lty = 2, col = "black", lwd = 0.5)
```

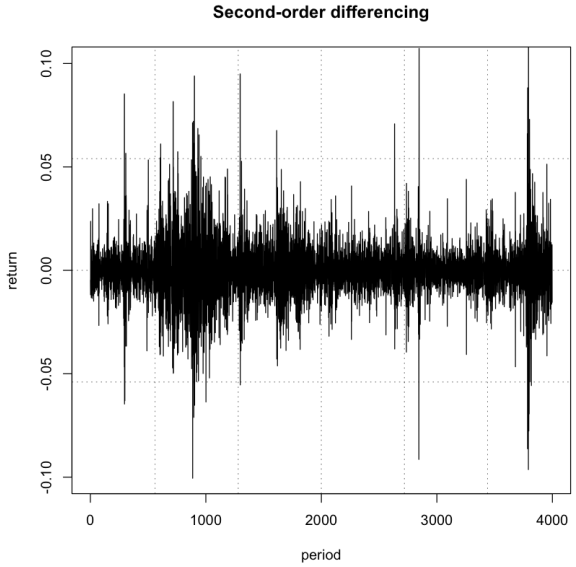


Figure 3: Second-order differencing

There is a degree of subjectivity in selecting which differences to apply. The seasonally differenced data in **fig.3** do not show substantially different behaviour from differenced data in **fig.2**.

2.4 Stationarity check

To determine is a time series is stationary we can make use of ACF and DACF plots.

ACF is the correlation between time series with a lagged version of itself. The correlation between the observation at the current time spot and the observations at previous time spots.The autocorrelation function starts a lag 0, which is the correlation of the time series with itself and therefore results in a correlation of 1. **Autocorrelation function (ACF)** identify if correlation at different time lags goes to 0 and therefore answering the following questions:

- Is the observed time series white noise/random?
- Is an observation related to an adjacent observation, an observation twice-removed, and so on?
- Can the observed time series be modeled with an MA model? If yes, what is the order?

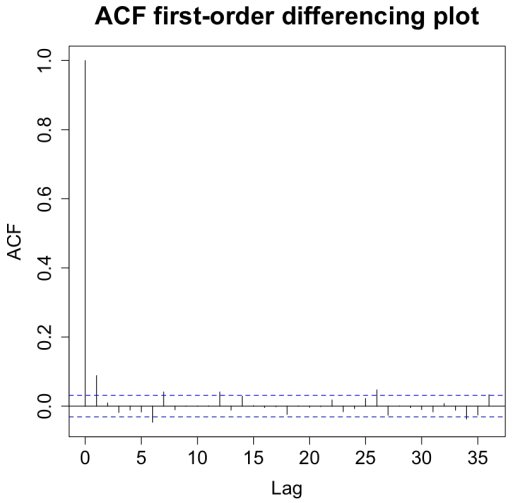
The **PACF** gives the partial correlation of a stationary time series with its own lagged values, regressed the values of the time series at all shorter lags. It contrasts with the autocorrelation function, which does not control for other lags. PACF plot can provide answers to the following questions:

- Can the observed time series be modeled with an AR model? If yes, what is the order?

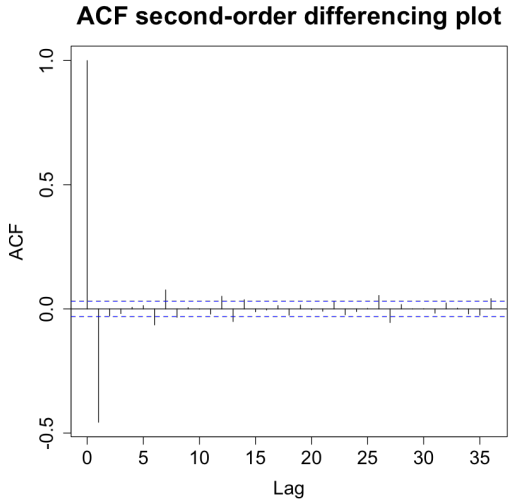
2.4.1 ACF and PACF plots

To help determining if ACF and PACF plots are tailing off to zero, many softwares will automatically plot two blue lines, in which depict the 95% confidence interval and is in indicator for the significance threshold. That means, anything within the blue area is statistically close to zero and anything outside the blue area is statistically non-zero.

```
# Plot ACF
acf(ftmc.diff,cex.axis=1.5,cex.lab=1.5,main="")
title("ACF first-order differencing plot",cex.main=2)
acf(ftmc.diff,cex.axis=1.5,cex.lab=1.5,main="", plot = FALSE)

acf(ftmc.diff2,cex.axis=1.5,cex.lab=1.5,main="")
title("ACF second-order differencing plot",cex.main=2)
acf(ftmc.diff2,cex.axis=1.5,cex.lab=1.5,main="", plot=FALSE)
```




```
Autocorrelations of series 'ftmc.diff', by lag

    0     1     2     3     4     5     6     7     8     9    10
1.000 0.088 0.009 -0.018 -0.012 -0.017 -0.047 0.041 -0.011 -0.001 -0.001
   11    12    13    14    15    16    17    18    19    20    21
0.001 0.041 -0.012 0.029 0.002 -0.004 -0.002 -0.024 0.001 -0.003 -0.001
   22    23    24    25    26    27    28    29    30    31    32
0.018 -0.016 -0.007 0.022 0.047 -0.026 0.001 -0.005 -0.010 -0.017 0.008
   33    34    35    36
-0.012 -0.037 -0.026 0.033
```

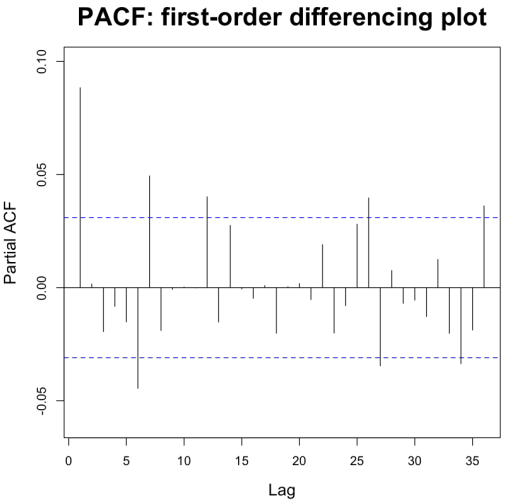


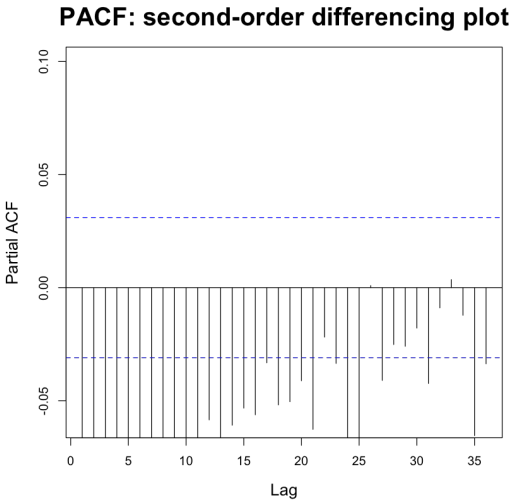

```
Autocorrelations of series 'ftmc.diff2', by lag

    0     1     2     3     4     5     6     7     8     9    10
1.000 -0.457 -0.028 -0.019 0.007 0.013 -0.065 0.077 -0.034 0.005 -0.001
   11    12    13    14    15    16    17    18    19    20    21
-0.021 0.051 -0.051 0.037 -0.012 -0.005 0.013 -0.025 0.016 -0.003 -0.010
   22    23    24    25    26    27    28    29    30    31    32
0.029 -0.024 -0.011 0.002 0.054 -0.055 0.018 0.000 0.001 -0.017 0.024
   33    34    35    36
0.003 -0.020 -0.026 0.042
```

Figure 4: ACF of First and Second differencing

The ACF plots are showing that there are several autocorrelations that are significantly non-zero. Therefore, the time series is non-random. Both ACF of first and second order differencing cut off after lag n.1, even if the ACF of the firs differences falls to zero slightly more rapidly.

In the DACF plots we can observe that the first-order differencing drops off after the lag **n.1**, while the DACF plot for the second-order differences tails off to zero much slower. This is an evidence that the first-order differences is more stationary than the second-order.





Partial autocorrelations of series 'as.numeric(ftmc.diff)', by lag

1	2	3	4	5	6	7	8	9	10	11
0.088	0.002	−0.019	−0.008	−0.015	−0.045	0.049	−0.019	−0.001	0.000	0.000
12	13	14	15	16	17	18	19	20	21	22
0.040	−0.015	0.028	−0.001	−0.005	0.001	−0.020	0.000	0.002	−0.005	0.019
23	24	25	26	27	28	29	30	31	32	33
−0.020	−0.008	0.028	0.040	−0.035	0.008	−0.007	−0.006	−0.013	0.012	−0.020
34	35	36								
−0.034	−0.019	0.036								

Partial autocorrelations of series 'as.numeric(ftmc.diff2)', by lag

1	2	3	4	5	6	7	8	9	10	11
−0.457	−0.299	−0.239	−0.187	−0.134	−0.199	−0.108	−0.114	−0.103	−0.093	−0.122
12	13	14	15	16	17	18	19	20	21	22
−0.058	−0.095	−0.061	−0.053	−0.056	−0.033	−0.052	−0.050	−0.041	−0.063	−0.022
23	24	25	26	27	28	29	30	31	32	33
−0.034	−0.067	−0.075	0.001	−0.041	−0.025	−0.026	−0.018	−0.042	−0.009	0.004
34	35	36								
−0.012	−0.065	−0.034								

Figure 5: ACF of First and Second differencing

Qualitatively, we can see and conclude from the ACFs and PACFs that the signal of the **first-order differencing** is stationary, and it is the best one bewtween the two, so we can now stop considering the second-order differencing time series.

2.4.2 Ljung/Pierce Box tests

Quantitatively, we can also use built-in test for testing stationariy.

First, the Ljung-Box test examines whether there is significant evidence for non-zero correlations at given lags (1-35 shown below), with the null hypothesis of independence in a given time series (a non-stationary signal will have a low p-value). This would tell us if the time series is a white noise

The Pierce-Box checks if the residuals are random (which is the case for residuals from a well specified model), therefore we could tell if the time series is non-stationary.

The tests show small p-values, suggesting that we should reject the null hypothesis of both tests, therefore the series is not white noise and it is stationary.

```
# Perform Ljung-Box and Box-Pierce tests
lag.length = 35
Box.test(ftmc.diff, lag=lag.length, type="Ljung-Box")
Box.test(ftmc.diff,lag=lag.length,type="Box-Pierce")
```

```
        Box-Ljung test

data:  ftmc.diff
X-squared = 90.799, df = 35, p-value = 7.591e-07


        Box-Pierce test

data:  ftmc.diff
X-squared = 90.478, df = 35, p-value = 8.423e-07
```

Figure 6: Ljung/Pierce Box Test for first-order differencing

3 Select a model and fit a classic ARMA-GARCH process

3.1 Model selection: correlogram analysis

Once the time series has been transformed by differencing it n-times, the next step is to select the appropriate ARIMA model, which means finding the values of most appropriate values of p and q for an ARIMA(p,d,q) model [1].

To do this, we need to examine the correlogram (ACF) and partial correlogram (PACF) of the stationary time series.
We see from the first-order correlogram that the autocorrelation at lag 1 (0.088) exceeds the significance bounds, but all other autocorrelations between lags 1-36 do not exceed the significance bounds.
The partial correlogram inf **fig.5a** shows that the partial autocorrelation at lags 1 exceeds the significance bounds, and is slowly decreasing in magnitude. We can consider the partial autocorrelations tailing off to zero after lag 1.
Since the correlogram is zero after lag 1, and the partial correlogram tails off to zero after lag 1, this means that the following ARMA (autoregressive moving average) models are possible for the time series of first differences:

- an **ARMA(1,0)** model, that is, an autoregressive model of order p=1, since the partial autocorrelogram is zero after lag 1, and the autocorrelogram tails off to zero.
- an **ARMA(0,1)** model, that is, a moving average model of order q=1, since the autocorrelogram cuts off after lag 1 and the partial autocorrelogram tails off to zero with a geometric decay.
- an **ARMA(1,1)** model, that is, a mixed model with p and q greater than 0.

3.2 Selecting 'armaOrder' for the 'mean.model' in ARMA-GARCH

The auto.arima() function in R uses a variation of the Hyndman-Khandakar algorithm, which combines unit root tests, minimisation of the AICc and MLE to obtain an ARIMA model.
By feeding the first-order time series into the function we obtain the following:

```
# Load auto.arima()
library(forecast)

# Run function auto.arima() using BIC criteria
auto.arima(ftmc.diff, max.p=9, max.q=9, ic="bic",approximation = FALSE)
```

```
Series: ftmc.diff
ARIMA(1,0,0) with zero mean

Coefficients:
         ar1
      0.0889
s.e.  0.0157

sigma^2 = 0.0001333:  log likelihood = 12166.83
AIC=-24329.66   AICc=-24329.66   BIC=-24317.08
```

Figure 7: auto.arima() output

The function output as the best model the ARIMA (1,0,0), and because we fed the first differences time series ($d = 1$).

This mean that the function is considering an ARMA(1,0) model is an autoregressive model of order 1, or AR(1) model. This model can be written as:

$$X_t - \mu = \phi(X_{t-1} - \mu) + \epsilon_t$$

where X_t is the stationary time series we are studying, μ is the mean of time series X_t, ϕ is the parameter to be estimated, and ϵ_t is white noise with mean zero and constant variance.

Since an ARMA(1,0) model (with p=1, q=0) is taken to be the best candidate model for the time series of first differences of the FTSE250 daily returns, then the original time series can be modelled using an ARIMA(1,1,0) model (with p=1, d=1, q=0, where d is the order of differencing required).

Therefore, the **ARMA(1,0) model will be the mean model for the ARMA-GARCH model**.

3.3 Selecting with Bayesian Criterion the ARMA-GARCH model

Now that we have found the mean model for the ARMA-GARCH model we will choose what parameters for GARCH variance model will give use the best ARMA-GARCH.

To do so following we can find the code that compare four ARMA-GARCH models, all with mean model ARMA(1,0): GARCH(1,1), GARCH(1,2), GARCH(2,1), GARCH(2,2). **We will select the model with the lower BIC value**:

```
# ARMA(1,0) + GARCH(1,1)
garch.spec.norm10 = ugarchspec(variance.model=list(garchOrder=c(1,1)),
                               mean.model = list(armaOrder=c(1,0)))
ftmc.garch.norm10 = ugarchfit(spec=garch.spec.norm10, data=ftmc.diff)
coef(ftmc.garch.norm10)
infocriteria(ftmc.garch.norm10)

# ARMA(1,0) + GARCH(1,2)
garch.spec.norm12 = ugarchspec(variance.model=list(garchOrder=c(1,2)),
                               mean.model = list(armaOrder=c(1,0)))
ftmc.garch.norm12 = ugarchfit(spec=garch.spec.norm12, data=ftmc.diff)
coef(ftmc.garch.norm12)
infocriteria(ftmc.garch.norm12)

# ARMA(1,0) + GARCH(2,1)
```

```
garch.spec.norm21 = ugarchspec(variance.model=list(garchOrder=c(2,1)),
                               mean.model = list(armaOrder=c(1,0)))
ftmc.garch.norm21 = ugarchfit(spec=garch.spec.norm21, data=ftmc.diff)
coef(ftmc.garch.norm21)
infocriteria(ftmc.garch.norm21)

# ARMA(1,0) + GARCH(2,2)
garch.spec.norm22 = ugarchspec(variance.model=list(garchOrder=c(2,2)),
                               mean.model = list(armaOrder=c(1,0)))
ftmc.garch.norm22 = ugarchfit(spec=garch.spec.norm22, data=ftmc.diff)
coef(ftmc.garch.norm22)
infocriteria(ftmc.garch.norm22)
```

Akaike	-6.505238
Bayes	-6.497369
Shibata	-6.505241
Hannan-Quinn	-6.502448

Akaike	-6.504656
Bayes	-6.495213
Shibata	-6.504660
Hannan-Quinn	-6.501309

Akaike	-6.505534
Bayes	-6.496091
Shibata	-6.505538
Hannan-Quinn	-6.502186

Akaike	-6.505034
Bayes	-6.494017
Shibata	-6.505040
Hannan-Quinn	-6.501128

Figure 8: Comparison of the ARMA-GARCH models

The final model selected is **ARMA(1,0)-GARCH(1,1)**, with the **lowest BIC** value(-6.497369).

4 Fit a GARCH model and compare it to the ARMA-GARCH model

4.1 Selecting an extension of the GARCH

Traditional GARCH models fail to explain the asymmetry of the distribution of errors and the leverage effect. The leverage effect is caused by the fact that negative returns have a greater influence on future volatility than do positive returns.[3]

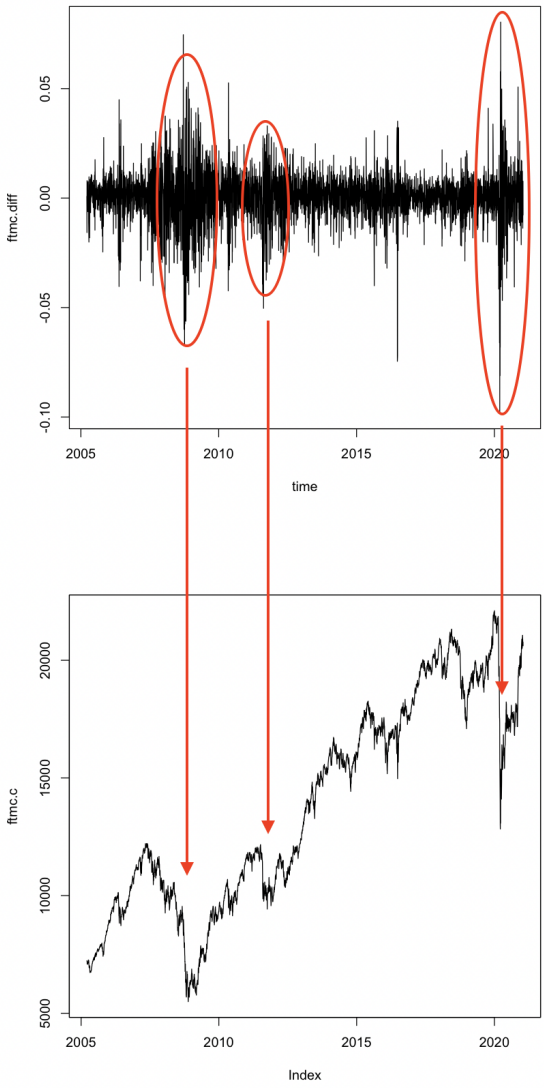


Figure 9: Leverage effect or Asymmetric volatility

For the standard GARCH process described before, the conditional variance is given by:

$$\sigma_t^2 = \omega + \sum_{i=1}^{p} \alpha_i Z_{t-i}^2 + \sum_{j=1}^{q} \beta_j \sigma_{t-j}^2$$

This process cannot account for the leverage effect, since σ_t^2 is a function of past innovations squared, Z_{t-i}^2, so **whether past values of the residuals or shocks are positive or negative has no effect on the future volatility**.

Various extensions of the classic GARCH model have been developed. Valid models currently implemented in the **ugarchspec()** function are: sGARCH (standard), fGARCH (Hentschel's fGARCH model, also known as Family GARCH), eGARCH (exponential GARCH), gjrGARCH (Glosten- Jagannathan- Runkle GARCH), apARCH, iGAR (Integrated GARCH). In some cases, these extensions lead to little improvement of the standard model.

Two commonly used models to estimate the **asymmetric volatility** are the **eGARCH**, and **gjrGARCH** models.

4.2 EGARCH model

We will choose EGARCH model for volatilty modelling, as it is one of the most diffused model to model volatility in finance.

The time series X_t s an Exponential GARCH process of order p and q, EGARCH(p,q), if [4]:

$$\begin{cases} Z_t = \sigma_t \epsilon_t \\ \log(\sigma_t^2) = \omega + \sum_{i=1}^{p} \alpha_i \frac{|Z_{t-i}| - \gamma_i Z_{t-i}}{\sigma_{t-i}} + \sum_{j=1}^{q} \beta_j \log(\sigma_{t-j}^2) \end{cases} \tag{6}$$

We can notice that the **variance is always positive, without any restrictions on parameters**. This is an advantage over standard GARCH.
The implementation in R it is the following:

```
# Define model specifications
garch.spec.norm2 = ugarchspec(variance.model=list(model="eGARCH", garchOrder=
    c(1,1)),
mean.model = list(armaOrder=c(0,0)))

# Fit model to data
ftmc.garch.norm2 = ugarchfit(spec=garch.spec.norm2, data=ftmc.diff)

# Show coefficients and infocriteria
coef(ftmc.garch.norm2)
infocriteria(ftmc.garch.norm2)
```

mu: 0.000284694451717468 **omega:** -0.273146206762614 **alpha1:** -0.119168399493056 **beta1:** 0.970256136833466 **gamma1:** 0.199294951646383

A matrix: 4 × 1 of type dbl

Akaike	-6.530693
Bayes	-6.522823
Shibata	-6.530696
Hannan-Quinn	-6.527903

Figure 10: EGARCH model output in R

Because the presence of the leverage effect, as expected $\gamma_1 > 0$

4.3 Compare EGARCH with ARMA-GARCH model

We can decide which model fits best on the basis of the AIC and BIC values. The model with the lowest AIC and lowest BIC is considered the best, and as showed in the table below, both scores are in favour of the EGARCH model, even if the difference between the two values is very limited.

Model	AIC	BIC
ARMA(1,0)-GARCH(1,1)	-6.505238	-6.497369
EGARCH(1,1)	-6.530693	-6.522823

Another way to compare the models is interpreting the qqplots. In fig.11 o the left we have the qqplot from the ARMA-GARCH model, while on the right the one from the EGARCH. We can observe that the EGARCH qq plot has an improvement on the tails, meaning that is better suited to model the time series, as very low and very big returns are better modeled.

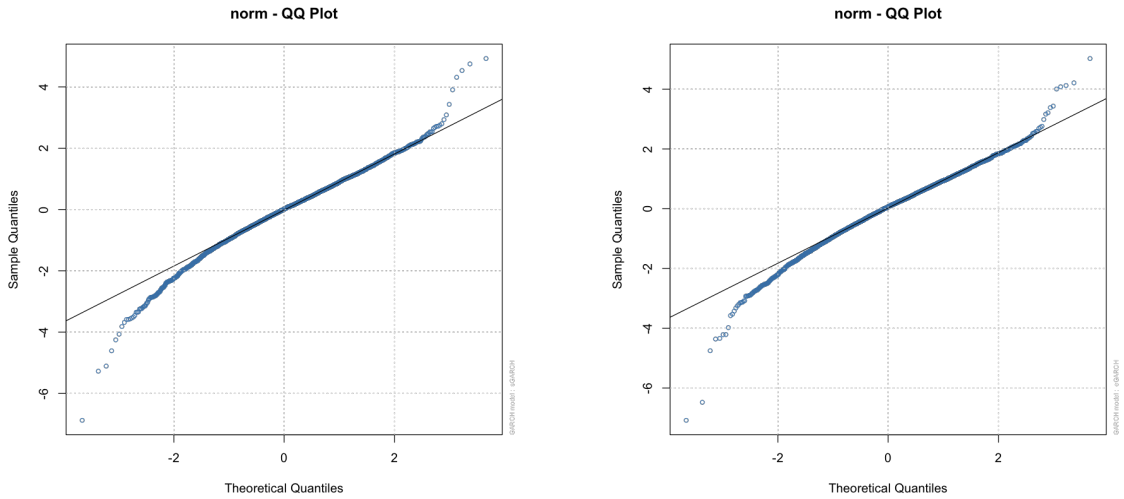


Figure 11: left: ARMA-GARCH qqplot, right: EGARCH qqplot

The plots in fig.12 shows the impact of the news on the volatility of the model. As we said at the beginning of par.4, the EGARCH model is able to capture the asymmetry of volatility: the ARMA-GARCH model has a simmetric volatility, while the EGARCH model has high volatility for negative returns, and a much lower volatility for positive ones.

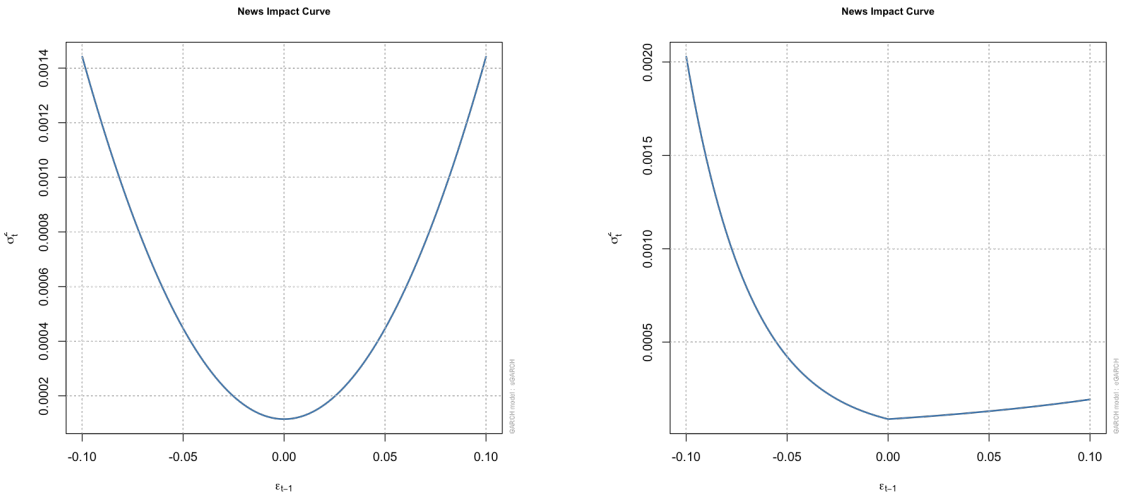


Figure 12: Asymmetry of volatility, left ARMA-GARCH, right EGARCH

We can conclude that the EGARCH model is a better model to fit the time series analyzed.

5 Plot the time-varying conditional volatility based on your selected GARCH model

In the GARCH models the conditional volatility is conditioned on past values of itself and of model errors. In fig.13 we can observe the graph of the EGARCH conditional volatility.

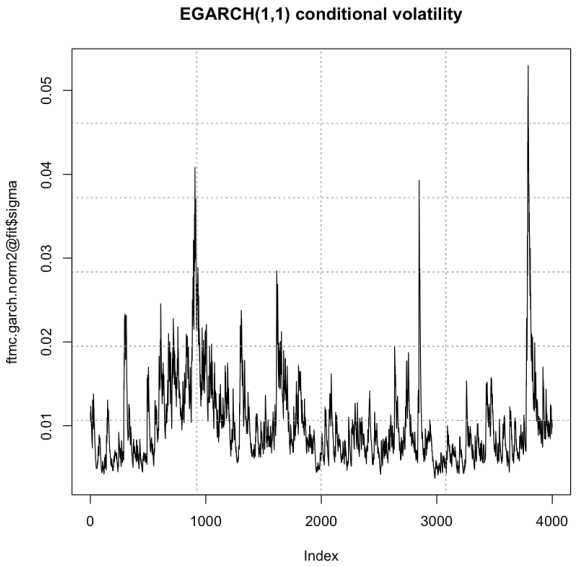


Figure 13: EGARCH(1,1) Conditional Volatility

6 APPENDIX: reference code

```
library(tseries)
library(zoo)
# This time interval guaratees 4000 observations
start <- "2005-03-18"
end <- "2021-01-16"
# Load data
ftmc <- get.hist.quote(instrument = "^FTMC", start, end,
                quote = "Adjusted",
                provider = c("yahoo"), method = NULL,
                origin = "1899-12-30", compression = "d",
                retclass = c("zoo", "ts"))
# Remove NA values
ftmc.c <- na.omit(ftmc)
# Plot chart
plot(ftmc.c, ylab="Value", xlab="Time", main="FTSE250 Daily Chart")
grid(nx = 4, ny = 6, lty = 3, col = "gray", lwd = 1)
#create log returns
ftmc.log <- log(as.numeric(ftmc.c[,1]))
ftmc.diff <- diff(ftmc.log)
ftmc.diff2 <- diff(ftmc.log, differences = 2)
# Plot first and second differencing
```

```
plot(ftmc.diff, type="l", main="First-order differencing", ylab="return",
    xlab="period", ylim=c(-0.10,0.10))
grid(nx = 6, ny = 4, lty = 3, col = "black", lwd = 0.5)
plot(ftmc.diff2, type="l", main="Second-order differencing", ylab="return",
    xlab="period", ylim=c(-0.10,0.10))
grid(nx = 6, ny = 4, lty = 3, col = "black", lwd = 0.5)
# Plot ACF and show values
acf(ftmc.diff,cex.axis=1.5,cex.lab=1.5,main="")
title("ACF first-order differencing plot",cex.main=2)
acf(ftmc.diff,cex.axis=1.5,cex.lab=1.5,main="", plot = FALSE)
acf(ftmc.diff2,cex.axis=1.5,cex.lab=1.5,main="")
title("ACF second-order differencing plot",cex.main=2)
acf(ftmc.diff2,cex.axis=1.5,cex.lab=1.5,main="", plot=FALSE)
# Plot PACF and show values
pacf(as.numeric(ftmc.diff),main="",cex.lab=1.3, ylim= c(-0.06,0.1))
title("PACF: first-order differencing plot",cex.main=2)
pacf(as.numeric(ftmc.diff),main="",cex.lab=1.3, ylim= c(-0.06,0.1), plot=
    FALSE)
pacf(as.numeric(ftmc.diff2),main="",cex.lab=1.3, ylim= c(-0.06,0.1))
title("PACF: second-order differencing plot",cex.main=2)
pacf(as.numeric(ftmc.diff2),main="",cex.lab=1.3, ylim= c(-0.06,0.1), plot=
    FALSE)
# Box test
lag.length = 35
Box.test(ftmc.diff, lag=lag.length, type="Ljung-Box") # test stationary
    signal
Box.test(ftmc.diff,lag=lag.length,type="Box-Pierce")
# Load auto.arima()
library(forecast)
# Choose best ARIMA fit
auto.arima(ftmc.diff, max.p=9, max.q=9, ic="bic",approximation = FALSE)
# Fit of best-fitting ARMA process
ftmc.ARMA10 <- arima(ftmc.c, c(1,1,0))
# sum(is.na(ftmc.ARMA10$resid)) = 1783
ftmc.ARMA10$resid <- na.remove(ftmc.ARMA10$resid)
# Plot ACF for of the fitted model
acf(as.numeric(ftmc.ARMA10$resid), cex.axis=1.5, cex.lab=1.5, main="")
title("ACF plot of ARMA(1,0) residuals", cex.main=2)
acf(as.numeric(ftmc.ARMA10$resid), cex.axis=1.5, cex.lab=1.5, main="", plot=
    FALSE)
# Box test
test.ARMA10 <- Box.test(ftmc.ARMA10$resid, lag=10, type="Ljung-Box", fitdf=2)
test.ARMA10
# Load library rugarch
library(rugarch)
# ARMA(1,0) + GARCH(1,1)
garch.spec.norm10 = ugarchspec(variance.model=list(garchOrder=c(1,1)),
                               mean.model = list(armaOrder=c(1,0)))
ftmc.garch.norm10 = ugarchfit(spec=garch.spec.norm10, data=ftmc.diff)
coef(ftmc.garch.norm10)
infocriteria(ftmc.garch.norm10)
# ARMA(1,0) + GARCH(1,2)
garch.spec.norm12 = ugarchspec(variance.model=list(garchOrder=c(1,2)),
                               mean.model = list(armaOrder=c(1,0)))
ftmc.garch.norm12 = ugarchfit(spec=garch.spec.norm12, data=ftmc.diff)
coef(ftmc.garch.norm12)
infocriteria(ftmc.garch.norm12)
# ARMA(1,0) + GARCH(2,1)
garch.spec.norm21 = ugarchspec(variance.model=list(garchOrder=c(2,1)),
                               mean.model = list(armaOrder=c(1,0)))
```

```
ftmc.garch.norm21 = ugarchfit(spec=garch.spec.norm21, data=ftmc.diff)
coef(ftmc.garch.norm21)
infocriteria(ftmc.garch.norm21)
# ARMA(1,0) + GARCH(2,2)
garch.spec.norm22 = ugarchspec(variance.model=list(garchOrder=c(2,2)),
                               mean.model = list(armaOrder=c(1,0)))
ftmc.garch.norm22 = ugarchfit(spec=garch.spec.norm22, data=ftmc.diff)
coef(ftmc.garch.norm22)
infocriteria(ftmc.garch.norm22)
# Plot
plot((ftmc.garch.norm11@fit$residuals)^2, type="l")
lines((ftmc.garch.norm11@fit$var), col = "green")
# Show result
ftmc.garch.norm10
# Load library ggplot2
library(ggplot2)
# Create model specifications
garch.spec.norm2 = ugarchspec(variance.model=list(model="eGARCH", garchOrder=
    c(1,1)),
                               mean.model = list(armaOrder=c(0,0)))
# Fit model to the data
ftmc.garch.norm2 = ugarchfit(spec=garch.spec.norm2, data=ftmc.diff)
# Show results
coef(ftmc.garch.norm2)
infocriteria(ftmc.garch.norm2)
ftmc.garch.norm2
names(ftmc.garch.norm2@fit)
infocriteria(ftmc.garch.norm10)
infocriteria(ftmc.garch.norm2)
# qqplot
plot(ftmc.garch.norm10, which = 9)
plot(ftmc.garch.norm2, which = 9)
# ACF
acf(as.numeric(ftmc.garch.norm@fit$residuals), cex.axis=1.5, cex.lab=1.5,
    main="")
acf(as.numeric(ftmc.garch.norm2@fit$residuals), cex.axis=1.5, cex.lab=1.5,
    main="")
# Volatility asymmetry
plot(ftmc.garch.norm2, which = 12)
plot(ftmc.garch.norm10, which = 12)
# Plot conditional volatility
plot(ftmc.garch.norm2@fit$sigma, type="l", main="EGARCH(1,1) conditional
    volatility")
grid(nx = 4, ny = 6, lty = 3, col = "gray", lwd = 1)
```

References

[1] Avril Coghlan. **Using R for Time Series Analysis**. https://a-little-book-of-r-for-time-series.readthedocs.io/en/latest/src/timeseries.html. 2018.

[2] Rob J Hyndman and Monash University George Athanasopoulos. **Stationarity and differencing, Forecasting: Principles and Practice**. https://otexts.com/fpp2/stationarity.html. 2018.

[3] Dr. Shailesh Rastogi. **Volatility Estimation using GARCH Family of Models: Comparison with Option Pricing**. http://www.pbr.co.in/2018/2018_month/Feb/7.pdf. 2018.

[4] Keming Yu. **Notes: Time Series Modelling**. MA5629 2022 Course Slides. 2021.