Project Time Series

2023-02-15

Stori Matteo: [matteo.stori01@icatt.it](mailto:matteo.stori01@icatt.it) | Esposito Francesco: [francesco.esposito06@icatt.it](mailto:francesco.esposito06@icatt.it) Lanubile Fabrizio: [fabrizio.lanubile01@icatt.it](mailto:fabrizio.lanubile01@icatt.it) | Marchesi Serena: [serena.marchesi02@icatt.it](mailto:serena.marchesi02@icatt.it)

The dataset contains the weekly fuel prices in Italy from March 2005 to August 2021, but we are looking only for the prices from 2013 till now, in order to carry on an appropriate analysis of this time series. Prices are ordered according to time and are multiplied by a factor of 1000 to have a more precise accuracy on the decimal numbers. All the indicated prices are referring to self service prices, and for each week, we have also the references to the IVA tax. All the prices are available on the Italian government site “Ministero dell’ambiente e della sicurezza energetica”.

library(readxl)

## Warning: il pacchetto 'readxl' è stato creato con R versione 4.1.3

# Loading the raw dataset  
#df <- read\_excel("C:/Users/kecco/Documents/CATTOLICA/TIME SERIES/weekly\_fuel\_prices\_all\_data\_from\_2005\_to\_20210823.xlsx")  
df <- read\_excel("C:/Users/matte/OneDrive/Desktop/TSA and Spatial/project/weekly\_fuel\_prices\_all\_data\_from\_2005\_to\_20210823.xlsx")  
  
# Checking the correctness of the dataframe structure  
str(df)

## tibble [4,980 x 8] (S3: tbl\_df/tbl/data.frame)  
## $ SURVEY\_DATE : POSIXct[1:4980], format: "2005-01-03" "2005-01-03" ...  
## $ PRODUCT\_ID : num [1:4980] 1 2 3 5 6 8 1 2 3 5 ...  
## $ PRODUCT\_NAME: chr [1:4980] "Euro-Super 95" "Automotive gas oil" "Heating gas oil" "LPG" ...  
## $ PRICE : chr [1:4980] "1115.75" "1018.28" "948.5" "552.5" ...  
## $ VAT : chr [1:4980] "185.96" "169.71" "158.08" "92.08" ...  
## $ EXCISE : chr [1:4980] "558.64" "403.21" "403.21" "156.62" ...  
## $ NET : chr [1:4980] "371.15" "445.36" "387.21" "303.8" ...  
## $ CHANGE : chr [1:4980] "-1.57" "-0.33" "-22.55" "0.22" ...

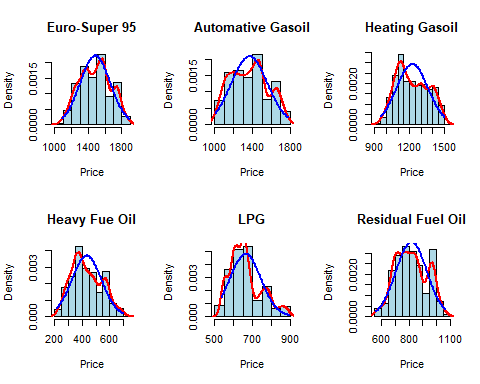
# Converting the "PRICE" column as numeric   
df$PRICE <- as.numeric(as.character(df$PRICE), na.rm = TRUE)  
  
# Subsetting the whole dataframe by "PRODUCT\_NAME" column  
df\_EuroSuper95 <- df[df$PRODUCT\_NAME == "Euro-Super 95", ]  
df\_Automotive\_gasoil <- df[df$PRODUCT\_NAME == "Automotive gas oil", ]  
df\_Heating\_gasoil <- df[df$PRODUCT\_NAME == "Heating gas oil", ]  
df\_LPG <- df[df$PRODUCT\_NAME == "LPG", ]  
df\_Residual\_fueloil <- df[df$PRODUCT\_NAME == "Residual fuel oil", ]  
df\_Heavy\_fueloil <- df[df$PRODUCT\_NAME == "Heavy fuel oil", ]

Now we have 6 different dataframes.

EXPLORATORY ANALYSIS

Comparing all the different types of fuel by their distribution

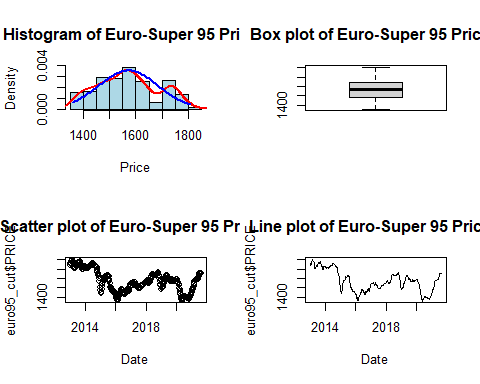
#x11();  
par(mfrow = c(2,3))  
  
# EURO95 --- Histogram of the "PRICE" column  
hist(df\_EuroSuper95$PRICE, main = "Euro-Super 95",  
 xlab = "Price", col = "lightblue", border = "black", probability = TRUE)  
d <- density(df\_EuroSuper95$PRICE)  
lines(d, col = "red", lwd = 2)  
mu <- mean(df\_EuroSuper95$PRICE)  
sigma <- sd(df\_EuroSuper95$PRICE)  
x <- seq(min(df\_EuroSuper95$PRICE), max(df\_EuroSuper95$PRICE), length.out = 100)  
y <- dnorm(x, mean = mu, sd = sigma)  
lines(x, y, col = "blue", lwd = 2)  
  
# AUTOMOTIVE\_GASOIL -- Histogram of the "PRICE" column  
hist(df\_Automotive\_gasoil$PRICE, main = "Automative Gasoil",  
 xlab = "Price", col = "lightblue", border = "black", probability = TRUE)  
d <- density(df\_Automotive\_gasoil$PRICE)  
lines(d, col = "red", lwd = 2)  
mu <- mean(df\_Automotive\_gasoil$PRICE)  
sigma <- sd(df\_Automotive\_gasoil$PRICE)  
x <- seq(min(df\_Automotive\_gasoil$PRICE), max(df\_Automotive\_gasoil$PRICE), length.out = 100)  
y <- dnorm(x, mean = mu, sd = sigma)  
lines(x, y, col = "blue", lwd = 2)  
  
# HEATING GASOIL --- Histogram of the "PRICE" column  
hist(df\_Heating\_gasoil$PRICE, main = "Heating Gasoil",  
 xlab = "Price", col = "lightblue", border = "black", probability = TRUE)  
d <- density(df\_Heating\_gasoil$PRICE)  
lines(d, col = "red", lwd = 2)  
mu <- mean(df\_Heating\_gasoil$PRICE)  
sigma <- sd(df\_Heating\_gasoil$PRICE)  
x <- seq(min(df\_Heating\_gasoil$PRICE), max(df\_Heating\_gasoil$PRICE), length.out = 100)  
y <- dnorm(x, mean = mu, sd = sigma)  
lines(x, y, col = "blue", lwd = 2)  
  
# HEAVY FUEL OIL --- Histogram of the "PRICE" column  
hist(df\_Heavy\_fueloil$PRICE, main = "Heavy Fue Oil",  
 xlab = "Price", col = "lightblue", border = "black", probability = TRUE)  
d <- density(df\_Heavy\_fueloil$PRICE)  
lines(d, col = "red", lwd = 2)  
mu <- mean(df\_Heavy\_fueloil$PRICE)  
sigma <- sd(df\_Heavy\_fueloil$PRICE)  
x <- seq(min(df\_Heavy\_fueloil$PRICE), max(df\_Heavy\_fueloil$PRICE), length.out = 100)  
y <- dnorm(x, mean = mu, sd = sigma)  
lines(x, y, col = "blue", lwd = 2)  
  
# LPG --- Histogram of the "PRICE" column  
hist(df\_LPG$PRICE, main = "LPG",  
 xlab = "Price", col = "lightblue", border = "black", probability = TRUE)  
d <- density(df\_LPG$PRICE)  
lines(d, col = "red", lwd = 2)  
mu <- mean(df\_LPG$PRICE)  
sigma <- sd(df\_LPG$PRICE)  
x <- seq(min(df\_LPG$PRICE), max(df\_LPG$PRICE), length.out = 100)  
y <- dnorm(x, mean = mu, sd = sigma)  
lines(x, y, col = "blue", lwd = 2)  
  
# RESIDUAL FUEL OIL --- Histogram of the "PRICE" column  
hist(df\_Residual\_fueloil$PRICE, main = "Residual Fuel Oil",  
 xlab = "Price", col = "lightblue", border = "black", probability = TRUE)  
d <- density(df\_Residual\_fueloil$PRICE)  
lines(d, col = "red", lwd = 2)  
mu <- mean(df\_Residual\_fueloil$PRICE)  
sigma <- sd(df\_Residual\_fueloil$PRICE)  
x <- seq(min(df\_Residual\_fueloil$PRICE), max(df\_Residual\_fueloil$PRICE), length.out = 100)  
y <- dnorm(x, mean = mu, sd = sigma)  
lines(x, y, col = "blue", lwd = 2)



Since EURO-SUPER 95 is the most similar distribution to the Gaussian one, we selected it for our further analysis We want to look only at data from 2013 since in the 2008-2012 there was a big recession that could potentially make more complicate our analysis.

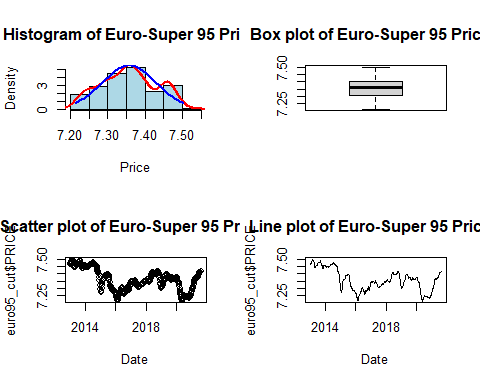
CREATION OF THE DATASET WITH date >= 2013 and product\_ID = 1

df$SURVEY\_DATE <- as.Date(df$SURVEY\_DATE, format = "%d/%m/%Y")  
  
euro95\_cut <- subset(df, format(df$SURVEY\_DATE, "%Y") >= "2013")  
euro95\_cut <- subset(euro95\_cut, euro95\_cut$PRODUCT\_ID == '1')  
  
#View(euro95\_cut)  
  
  
# EURO95 --- Histogram of the "PRICE" column  
#x11();   
par(mfrow = c(2,2))  
hist(euro95\_cut$PRICE, main = "Histogram of Euro-Super 95 Prices",  
 xlab = "Price", col = "lightblue", border = "black", probability = TRUE)  
d <- density(euro95\_cut$PRICE)  
lines(d, col = "red", lwd = 2)  
mu <- mean(euro95\_cut$PRICE)  
sigma <- sd(euro95\_cut$PRICE)  
x <- seq(min(euro95\_cut$PRICE), max(euro95\_cut$PRICE), length.out = 100)  
y <- dnorm(x, mean = mu, sd = sigma)  
lines(x, y, col = "blue", lwd = 2)  
  
  
boxplot(euro95\_cut$PRICE, main = "Box plot of Euro-Super 95 Prices")  
  
plot(euro95\_cut$SURVEY\_DATE, euro95\_cut$PRICE, main = "Scatter plot of Euro-Super 95 Prices",  
 xlab = "Date")  
  
plot(euro95\_cut$SURVEY\_DATE, euro95\_cut$PRICE, type = "l", main = "Line plot of Euro-Super 95 Prices",  
 xlab = "Date")



Take the logs of the y’s trying to make data more normal.

# EURO95 --- Histogram of the "PRICE" column  
euro95\_cut$PRICE = log(euro95\_cut$PRICE)  
  
#x11();   
par(mfrow = c(2,2))  
hist(euro95\_cut$PRICE, main = "Histogram of Euro-Super 95 Prices",  
 xlab = "Price", col = "lightblue", border = "black", probability = TRUE)  
d <- density(euro95\_cut$PRICE)  
lines(d, col = "red", lwd = 2)  
mu <- mean(euro95\_cut$PRICE)  
sigma <- sd(euro95\_cut$PRICE)  
x <- seq(min(euro95\_cut$PRICE), max(euro95\_cut$PRICE), length.out = 100)  
y <- dnorm(x, mean = mu, sd = sigma)  
lines(x, y, col = "blue", lwd = 2)  
  
  
boxplot(euro95\_cut$PRICE, main = "Box plot of Euro-Super 95 Prices")  
  
plot(euro95\_cut$SURVEY\_DATE, euro95\_cut$PRICE, main = "Scatter plot of Euro-Super 95 Prices",  
 xlab = "Date")  
  
plot(euro95\_cut$SURVEY\_DATE, euro95\_cut$PRICE, type = "l", main = "Line plot of Euro-Super 95 Prices",  
 xlab = "Date")

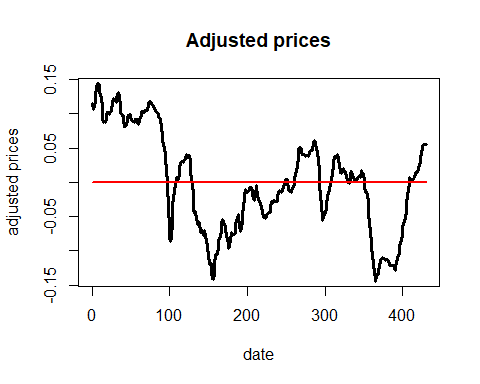


Centralization of the distribution

adj\_price=euro95\_cut$PRICE  
mean = mean(euro95\_cut$PRICE)  
adj\_price = (euro95\_cut$PRICE) - mean  
range(adj\_price)

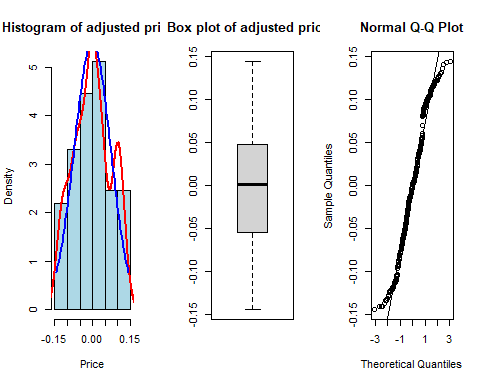
## [1] -0.1442151 0.1442516

val\_x=c()  
val\_y=c()  
plot(adj\_price,main="Adjusted prices",xlab="date",ylab="adjusted prices",xlim=c(0,430),ylim=c(-0.14,0.14),  
 xaxt="n",yaxt="n",type="l",lwd=3)  
axis(1,val\_x,val\_x)  
axis(2,val\_y,val\_y)  
lines(rep(mean(adj\_price,na.rm=TRUE),times=430),col="red",lwd=2)



Let’s check the Normality of the distribution.

#HISTOGRAM  
#x11();   
par(mfrow = c(1, 3))  
hist(adj\_price, main = "Histogram of adjusted prices",  
 xlab = "Price", col = "lightblue", border = "black", probability = TRUE)  
d <- density(adj\_price)  
lines(d, col = "red", lwd = 2)  
mu <- mean(adj\_price)  
sigma <- sd(adj\_price)  
x <- seq(min(adj\_price), max(adj\_price), length.out = 100)  
y <- dnorm(x, mean = mu, sd = sigma)  
lines(x, y, col = "blue", lwd = 2)  
  
#BOXPLOT  
boxplot(adj\_price,main = "Box plot of adjusted prices")   
  
#QQ-PLOT  
qqnorm(adj\_price)  
qqline(adj\_price)



In the histogram is evident that the red distribution (representing the trend of histograms and thus of our distribution) is quite far from being approximated to a normal one.

In the boxplot there is a good symmetry centered in zero, due to the centralization operation done above.

The QQ-plot is useful to determine if a set of data come from some theoretical distribution. Seeing the tails, it is quite clear that the data are not normal.

Let’s assume it is normal and analyze by normal model and see if the results are good or not.

shapiro.test(adj\_price) #Shapiro-Wilk Normality - independent data!

##   
## Shapiro-Wilk normality test  
##   
## data: adj\_price  
## W = 0.97268, p-value = 3.32e-07

#Jarque-Bera Normality - independent data!  
library (tseries)

## Warning: il pacchetto 'tseries' è stato creato con R versione 4.1.3

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

jarque.bera.test(adj\_price)

##   
## Jarque Bera Test  
##   
## data: adj\_price  
## X-squared = 13.376, df = 2, p-value = 0.001246

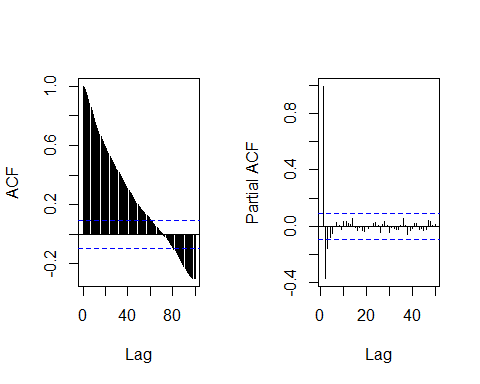
In the Shapiro Test the data are Normal if the p-value is above 0.05, in this case it is smaller than the threshold, so this test suggests that the data are not Normal. In the Jacques Brera test the null hp is that the data are normal. Due to the fact that the p-value<0.05, the test provides results that our data are not normal.

So, as we see from the graphs and tests, it would be rash to say that our data are normally distributed, despite the application of the ‘log’ function and centralization. Graphically there is some (faint) clue that may lead us to this assumption, but considering the test results we cannot conclude that our data are normal. This means that we will probably not have normally distributed residuals in our analysis.

To develop an analysis on this data, we will assume Normality.

A first step to begin to understand which type of models our data best fit is to use the Autocorrelation function and Partial Autocorrelation function. Respectively we will use: - ACF to identify the fit of the Moving Average model, since the latter is a model with exponentially decreasing autocorrelation as lag increases, ACF is fit (not adjusted for the influence of intermediate lags) - PACF to identify the fit of the Autoregressive Model. In this case, the autocorrelation of the model does not decay exponentially, so distant lags may have influential statistical significance, so PACF is used (adjusts the result for the influence of intermediate values).

#ACF vs PACF  
#x11();   
par(mfrow = c(1,2))  
acf(adj\_price,lag.max=100,main="")  
pacf(adj\_price,lag.max=50,main="")



From the results suggested by the previous functions, we can then begin to eliminate the MA model from the possible models applicable to our time series. What PACF suggests instead is quite different. It seems that the AR model fits. It is also possible to point out that there are 3 possible significant lagged values, identifiable by the three spikes beyond the horizontal limit, and so the maximum order that can be considered in our models that we are going to test should be 3.  
Recall that the horizontal limit is an approximation of 0, usually computed as 2/sqrt(n).

We can implement now the step of the Model Specification. Assuming that our data are stationary, then we can start comparing different models. It is reasonable to exclude comparisons with MA models of different order based on what has been pointed out by the ACF. Instead, it will be useful to compare different orders of AR models with a more complex model such as the ARMA model to see whether it is worth using higher complexity or not.

Moreover, for each model we will compute AIC and BIC. They are two fundamental tools used to choose the best model. Both of them are based on the LogLikelihood function, which will be also useful for the next step, when we will estimate the parameters of the chosen model using the MLE method. We will choose the model that provides us the lower AIC/BIC.

n = length(adj\_price)  
AIC=rep(0,times=8)  
BIC=rep(0,times=8)  
AIC.tsa=rep(0,times=8)  
  
## AR(1) ##  
  
# Z\_{t} = phi\_{1} \* Z\_{t-1} + a\_{t} #  
  
dim=2  
AR1=arima(adj\_price,order=c(1,0,0),method='ML')  
  
est.phi1=AR1$coef[1]  
est.sigma2=AR1$sigma2  
  
LogLik=-0.5\*(n-1)\*log(2\*pi\*est.sigma2)  
for (t in 2:n) {  
 LogLik=LogLik-0.5\*((adj\_price[t]-est.phi1\*adj\_price[t-1])^2)/est.sigma2  
}  
  
Deviance=-2\*LogLik  
  
AIC[1]=Deviance+2\*dim  
BIC[1]=Deviance+dim\*log(n)  
AIC.tsa[1]=AR1$aic  
  
## AR(2) ##  
  
# Z\_{t} = phi\_{1} \* Z\_{t-1} + phi\_{2} \* Z\_{t-2} + a\_{t} #  
  
dim=3  
AR2=arima(adj\_price,order=c(2,0,0),method='ML')  
  
est.phi1=AR2$coef[1]  
est.phi2=AR2$coef[2]  
est.sigma2=AR2$sigma2  
  
LogLik=-0.5\*(n-2)\*log(2\*pi\*est.sigma2)  
for (t in 3:n) {  
 LogLik=LogLik-0.5\*((adj\_price[t]-est.phi1\*adj\_price[t-1]-est.phi2\*adj\_price[t-2])^2)/est.sigma2  
}  
  
Deviance=-2\*LogLik  
  
AIC[2]=Deviance+2\*dim  
BIC[2]=Deviance+dim\*log(n)  
AIC.tsa[2]=AR2$aic  
  
  
## AR(3) ##  
  
# Z\_{t} = phi\_{1} \* Z\_{t-1} + phi\_{2} \* Z\_{t-2} + phi\_{3} \* Z\_{t-3} + a\_{t} #  
  
dim=4  
  
AR3=arima(adj\_price,order=c(3,0,0),method='ML')  
est.phi1=AR3$coef[1]  
est.phi2=AR3$coef[2]  
est.phi3=AR3$coef[3]  
est.sigma2=AR2$sigma2  
  
LogLik=-0.5\*(n-3)\*log(2\*pi\*est.sigma2)  
for (t in 4:n) {  
 LogLik=LogLik-0.5\*((adj\_price[t]-est.phi1\*adj\_price[t-1]-est.phi2\*adj\_price[t-2]  
 -est.phi3\*adj\_price[t-3])^2)/est.sigma2  
}  
  
Deviance=-2\*LogLik  
  
AIC[3]=Deviance+2\*dim  
BIC[3]=Deviance+dim\*log(n)  
AIC.tsa[3]=AR3$aic  
  
## AR(4) ##  
  
# Z\_{t} = phi\_{1} \* Z\_{t-1} + phi\_{2} \* Z\_{t-2}   
  
# + phi\_{3} \* Z\_{t-3} + phi\_{4} \* Z\_{t-4} + a\_{t} #  
  
dim=5  
  
AR4=arima(adj\_price,order=c(4,0,0),method='ML')  
  
est.phi1=AR4$coef[1]  
est.phi2=AR4$coef[2]  
est.phi3=AR4$coef[3]  
est.phi4=AR4$coef[4]  
est.sigma2=AR4$sigma2  
  
LogLik=-0.5\*(n-4)\*log(2\*pi\*est.sigma2)  
for (t in 5:n) {  
 LogLik=LogLik-0.5\*((adj\_price[t]-est.phi1\*adj\_price[t-1]-est.phi2\*adj\_price[t-2]  
 -est.phi3\*adj\_price[t-3]-est.phi4\*adj\_price[t-4])^2)/est.sigma2  
}  
  
Deviance=-2\*LogLik  
  
AIC[4]=Deviance+2\*dim  
BIC[4]=Deviance+dim\*log(n)  
AIC.tsa[4]=AR4$aic  
  
  
  
# ARMA(1,1) #  
  
# Z\_{t} = phi1 \* Z\_{t-1} + a\_{t} + theta1 \* a\_{t-1} #  
  
dim=3  
ARMA11=arima(adj\_price,order=c(1,0,1),method='ML')

## Warning in arima(adj\_price, order = c(1, 0, 1), method = "ML"): possible  
## convergence problem: optim gave code = 1

est.phi1=ARMA11$coef[1]  
est.theta1=ARMA11$coef[2]  
est.sigma2=ARMA11$sigma2  
  
a.obs=rep(0,times=n)  
  
LogLik=-0.5\*(n-1)\*log(2\*pi\*est.sigma2)  
for (t in 2:n) {  
 a.obs[t]=adj\_price[t]-est.phi1\*adj\_price[t-1]-est.theta1\*a.obs[t-1]  
 LogLik=LogLik-0.5\*(a.obs[t]^2)/est.sigma2  
}  
  
Deviance=-2\*LogLik  
  
AIC[5]=Deviance+2\*dim  
BIC[5]=Deviance+dim\*log(n)  
AIC.tsa[5]=ARMA11$aic  
  
# ARMA(2,1) #  
  
# Z\_{t} = phi1 \* Z\_{t-1} + phi2 \* Z\_{t-2} + a\_{t} + theta1 \* a\_{t-1} #  
  
dim=4  
ARMA21=arima(adj\_price,order=c(2,0,1),method='ML')

## Warning in arima(adj\_price, order = c(2, 0, 1), method = "ML"): possible  
## convergence problem: optim gave code = 1

est.phi1=ARMA21$coef[1]  
est.phi2=ARMA21$coef[2]  
est.theta1=ARMA21$coef[3]  
est.sigma2=ARMA21$sigma2  
  
a.obs=rep(0,times=n)  
  
LogLik=-0.5\*(n-2)\*log(2\*pi\*est.sigma2)  
for (t in 3:n) {  
 a.obs[t]=adj\_price[t]-est.phi1\*adj\_price[t-1]-est.phi2\*adj\_price[t-2]-est.theta1\*a.obs[t-1]  
 LogLik=LogLik-0.5\*(a.obs[t]^2)/est.sigma2  
}  
Deviance=-2\*LogLik  
  
AIC[6]=Deviance+2\*dim  
BIC[6]=Deviance+dim\*log(n)  
AIC.tsa[6]=ARMA21$aic  
  
# ARMA(1,2) #  
  
# Z\_{t} = phi1 \* Z\_{t-1} + a\_{t} + theta1 \* a\_{t-1} + theta2 \* a\_{t-2} #  
  
dim=4  
ARMA12=arima(adj\_price,order=c(1,0,2),method='ML')

## Warning in arima(adj\_price, order = c(1, 0, 2), method = "ML"): possible  
## convergence problem: optim gave code = 1

est.phi1=ARMA12$coef[1]  
est.theta1=ARMA12$coef[2]  
est.theta2=ARMA12$coef[3]  
est.sigma2=ARMA12$sigma2  
  
a.obs=rep(0,times=n)  
  
LogLik=-0.5\*(n-2)\*log(2\*pi\*est.sigma2)  
for (t in 3:n) {  
 a.obs[t]=adj\_price[t]-est.phi1\*adj\_price[t-1]-est.theta1\*a.obs[t-1]-est.theta2\*a.obs[t-2]  
 LogLik=LogLik-0.5\*(a.obs[t]^2)/est.sigma2  
}  
Deviance=-2\*LogLik  
  
AIC[7]=Deviance+2\*dim  
BIC[7]=Deviance+dim\*log(n)  
AIC.tsa[7]=ARMA12$aic  
  
# ARMA(2,2) #  
  
# Z\_{t} = phi1 \* Z\_{t-1} + phi1 \* Z\_{t-1} + a\_{t}   
  
#+ theta1 \* a\_{t-1} + theta2 \* a\_{t-2} #  
  
dim=5  
ARMA22=arima(adj\_price,order=c(2,0,2),method='ML')  
  
est.phi1=ARMA22$coef[1]  
est.phi2=ARMA22$coef[2]  
est.theta1=ARMA22$coef[3]  
est.theta2=ARMA22$coef[4]  
est.sigma2=ARMA22$sigma2  
  
a.obs=rep(0,times=n)  
  
LogLik=-0.5\*(n-2)\*log(2\*pi\*est.sigma2)  
for (t in 4:n) {  
 a.obs[t]=adj\_price[t]-est.phi1\*adj\_price[t-1]-est.phi2\*adj\_price[t-2]  
 -est.theta1\*a.obs[t-1]-est.theta2\*a.obs[t-2]  
 LogLik=LogLik-0.5\*(a.obs[t]^2)/est.sigma2  
}  
  
Deviance=-2\*LogLik  
AIC[8]=Deviance+2\*dim  
BIC[8]=Deviance+dim\*log(n)  
AIC.tsa[8]=ARMA22$aic  
  
AIC

## [1] -3037.119 -3320.009 -3319.878 -3312.813 -3251.159 -3323.873 -3280.229  
## [8] -1407.112

BIC

## [1] -3028.992 -3307.818 -3303.623 -3292.494 -3238.967 -3307.618 -3263.973  
## [8] -1386.793

AIC.tsa

## [1] -3036.728 -3325.583 -3327.728 -3328.853 -3252.320 -3328.765 -3283.318  
## [8] -3249.195

min(AIC) # -3323.873 ARMA(2,1)

## [1] -3323.873

min(BIC) # -3307.818 AR(2)

## [1] -3307.818

min(AIC.tsa) # -3328 ARMA(2,1)

## [1] -3328.853

The model comparison gives us the following result: the model that minimize the BIC is AR(2). Now we should use it to go further in the analysis and proceeding in the estimation parameters step, but…

After performing the Model Specification, having assumed that our data are stationary, we can try to improve our analysis by going deeper and performing a test that allows us to understand whether the data are effectively stationary or nonstationary. Recall that stationarity means that the data maintain the same characteristics in the temporal distribution (the same distribution if we are talking about strictly stationary or the same covariances if we are talking about weakly stationary).

TEST STATIONARITY: Augmented Dickey Fuller Test will be adopted a statistical test used to check the stationarity.

library(tseries)   
x <- ts(euro95\_cut$PRICE)  
adf.test(x)

##   
## Augmented Dickey-Fuller Test  
##   
## data: x  
## Dickey-Fuller = -2.4912, Lag order = 7, p-value = 0.37  
## alternative hypothesis: stationary

When the test statistic is lower than the critical value shown, you reject the null hypothesis and infer that the time series is stationary. This is NOT the case. As we can see, the p-value is greater than 0.05, so the data are not stationary.

In this case we can use the ARIMA model, with the integration operation which is used to remove the non stationarity of the data. It is based on the differentiation between a value of the series and the previous one. The order ‘d’ which will be put in the function ARIMA(p, d, q) means how many times the integration operation will be done.  
Before applying the function, we develop the analytical code in order to execute the integration operation and then plot the data. This is good to understand the efficiency of the differentiation and also to replicate the Augmented Dickey Fuller Test checking the stationarity.

n=length(adj\_price)  
adj\_price.diff=rep(NA,times=n)   
# 2:n not 1:n-1  
for (t in 2:n) { # z[t]=z[t+1]-z[t]  
 adj\_price.diff[t]=adj\_price[t]-adj\_price[t-1]  
}  
  
mean=mean(adj\_price.diff,na.rm=TRUE)  
  
mean

## [1] -0.00014072

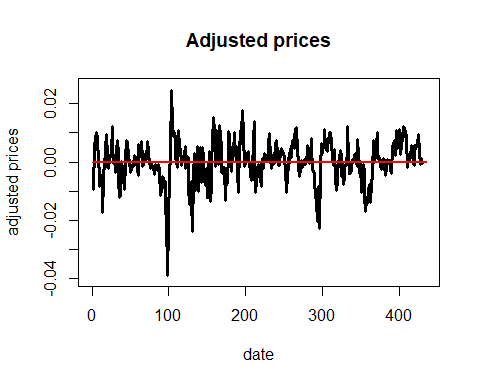
adj\_price.diff=adj\_price.diff-mean  
mean(adj\_price.diff,na.rm=TRUE)

## [1] -1.022099e-19

range(adj\_price.diff,na.rm=TRUE)

## [1] -0.03900336 0.02450208

val\_x=c()  
val\_y=c()  
plot(adj\_price.diff,main="Adjusted prices",xlab="date",ylab="adjusted prices",xlim=c(0,435),ylim=c(-0.040,0.026),  
 xaxt="n",yaxt="n",type="l",lwd=3)  
axis(1,val\_x,val\_x)  
axis(2,val\_y,val\_y)  
lines(rep(mean(adj\_price.diff,na.rm=TRUE),times=435),col="red",lwd=2)



It is clear that the distribution is quite better than the initial distribution. It extends more neatly through the mean horizontal line.

adj\_price.diff[1] = -1.022099e-19 #replace the first element (NA) with the mean of the new distribution in order to compute the test  
  
library(tseries)   
x <- ts(adj\_price.diff)  
adf.test(x)

## Warning in adf.test(x): p-value smaller than printed p-value

##   
## Augmented Dickey-Fuller Test  
##   
## data: x  
## Dickey-Fuller = -5.8491, Lag order = 7, p-value = 0.01  
## alternative hypothesis: stationary

The data are stationary since the null hp of the test is that the data are non stationary and the p-value <0.05.

Let’s now go deeply into the choice of the best ARIMA model applying directly the R function to the original data. After some implicit analysis, in addition to what the PACF and ACF function suggest, we thought that it will be useful to compare ARIMA models with non zero order respectively for the AR part and the integration part, because, as we have seen, the MA model is not so suitable for our data.

We compare the AIC and BIC between: ARIMA(1, 1, 0) ARIMA(2, 1, 0) ARIMA(3, 1, 0)

n=length(adj\_price)  
AIC\_arima=rep(0,times=3)  
BIC\_arima=rep(0,times=3)  
AIC.tsa\_arima=rep(0,times=3)  
  
#ARIMA(1, 1, 0)  
ARI11=arima(adj\_price,order=c(1,1,0),method='ML')   
dim=2 #k  
  
est.phi1=ARI11$coef[1]  
est.sigma2=ARI11$sigma2  
  
  
resMod = ARI11$residuals  
loglik <- sum(dnorm(resMod, mean = 0, sd = sd(resMod), log = TRUE))  
AIC\_arima[1] <- -2 \* loglik + 2 \* dim  
BIC\_arima[1] <- -2 \* loglik + dim \* log(n)  
AIC.tsa\_arima[1] <- AIC(ARI11)  
  
#ARIMA(2, 1, 0)  
ARI21=arima(adj\_price,order=c(2,1,0),method='ML')   
dim=3   
  
est.phi1=ARI21$coef[1]  
est.phi2=ARI21$coef[2]  
est.sigma2=ARI21$sigma2  
  
resMod = ARI21$residuals  
loglik <- sum(dnorm(resMod, mean = 0, sd = sd(resMod), log = TRUE))  
AIC\_arima[2] <- -2 \* loglik + 2 \* dim  
BIC\_arima[2] <- -2 \* loglik + dim \* log(n)  
AIC.tsa\_arima[2] <- AIC(ARI21)  
  
  
#ARIMA(3, 1, 0)  
ARI31=arima(adj\_price,order=c(3,1,0),method='ML')   
dim=4 #k  
  
est.phi1=ARI31$coef[1]  
est.phi2=ARI31$coef[2]  
est.phi2=ARI31$coef[3]  
est.sigma2=ARI31$sigma2  
  
resMod = ARI31$residuals  
loglik <- sum(dnorm(resMod, mean = 0, sd = sd(resMod), log = TRUE))  
AIC\_arima[3] <- -2 \* loglik + 2 \* dim  
BIC\_arima[3] <- -2 \* loglik + dim \* log(n)  
AIC.tsa\_arima[3] <- AIC(ARI31)  
  
AIC\_arima

## [1] -3331.594 -3334.782 -3335.161

BIC\_arima

## [1] -3323.466 -3322.591 -3318.906

AIC.tsa\_arima

## [1] -3322.169 -3325.319 -3325.677

# Which is the best model?  
min(AIC\_arima)

## [1] -3335.161

min(BIC\_arima)

## [1] -3323.466

min(AIC.tsa\_arima)

## [1] -3325.677

choosing BIC as main reference, we’ll chose ARIMA(1, 1, 0) as the best model.

We also find confirmation in the BIC minor model comparison between models with assumed stationarity. We see that the BIC of ARIMA(1, 1, 0) is less than the BIC of AR(2) which is the least among models without Integration (‘d’ order).

min(BIC\_arima) # ARIMA

## [1] -3323.466

min(BIC) # AR

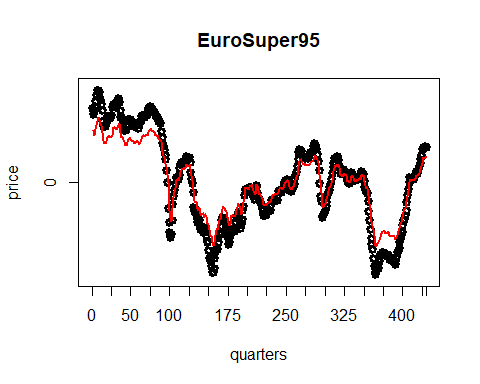
## [1] -3307.818

#ARIMA(1, 1, 0)  
#recall the parameters of the chosen model  
est.phi1=ARI11$coef[1]  
est.sigma2=ARI11$sigma2

we will choose the ARIMA(1, 1, 0) as model.

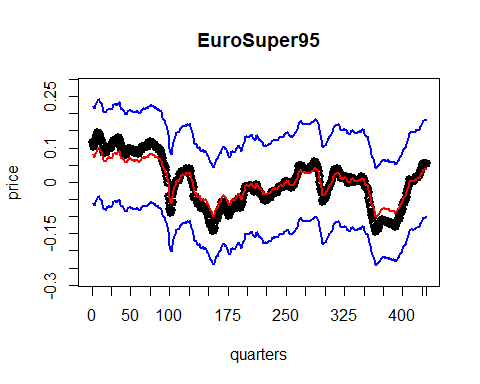
ESTIMATION PARAMETER STEP

adj\_price.fit=rep(0,times=n)  
  
adj\_price.fit[1]=NA  
  
for (t in 2:n) {  
 adj\_price.fit[t]=est.phi1\*adj\_price[t-1]  
}  
  
  
val\_x=c(0,(1:20)\*25,430)  
val\_y=(c(-2:2)\*2.5)  
plot(adj\_price,main="EuroSuper95",xlab="quarters",ylab="price",xlim=c(0,430),ylim=c(-0.15,0.15),  
 xaxt="n",yaxt="n",type="p",lwd=3)  
axis(1,val\_x,val\_x)  
axis(2,val\_y,val\_y)  
lines(adj\_price.fit,col="red",lwd=2)

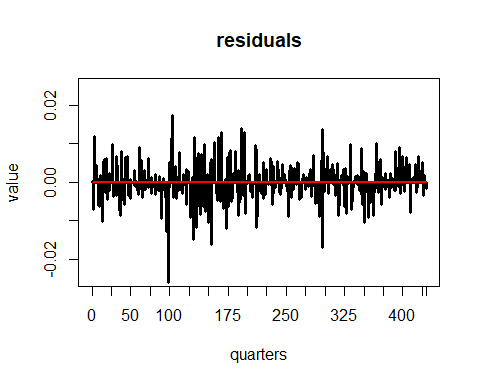


Confidence intervals

# Left (lower bound)  
left=rep(0,times=n)  
left[1]=NA  
  
for (t in 2:n) {  
 left[t]=adj\_price.fit[t]-1.960\*sd(adj\_price) #P(|Z|<1.960)=0.95 Z=Standard Normal  
} #P(|Z|<2.576)=0.99  
  
# Right (upper bound)  
right=rep(0,times=n)  
right[1]=NA  
  
for (t in 2:n) {  
 right[t]=adj\_price.fit[t]+1.960\*sd(adj\_price) #P(|Z|<1.960)=0.95 Z=Standard Normal  
}  
  
val\_x=c(0,(1:20)\*25,430)  
val\_y=c(0,(-15:15)\*0.05,0.30)  
#win.graph(width=12,height=6)  
plot(adj\_price,main="EuroSuper95",xlab="quarters",ylab="price",xlim=c(0,430),ylim=c(-0.28,0.28),  
 xaxt="n",yaxt="n",type="p",lwd=3)  
axis(1,val\_x,val\_x)  
axis(2,val\_y,val\_y)  
lines(adj\_price.fit,col="red",lwd=2)  
lines(left,col="blue",lwd=2)  
lines(right,col="blue",lwd=2)



adj\_price.res=ARI11$residuals  
  
val\_x=c(0,(1:20)\*25,430)  
val\_y=c()  
#win.graph(width=12,height=6)  
plot(adj\_price.res,main="residuals",xlab="quarters",ylab="value",xlim=c(0,430),ylim=c(-0.025,0.025),  
 xaxt="n",yaxt="n",type="h",lwd=3)  
axis(1,val\_x,val\_x)  
axis(2,val\_y,val\_y)  
lines(rep(0,times=n),type="l",col="red",lwd=2)



RMSE=sqrt(mean(adj\_price.res^2,na.rm=TRUE))  
RMSE

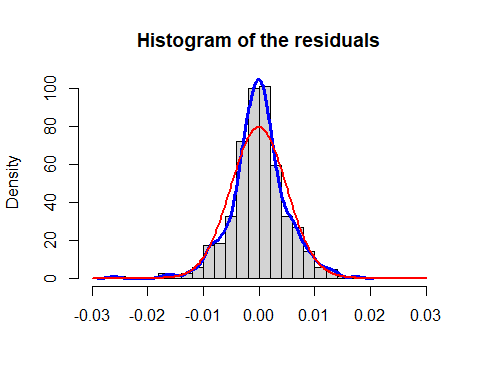
## [1] 0.005003902

MAE=mean(abs(adj\_price.res),na.rm=TRUE)  
MAE

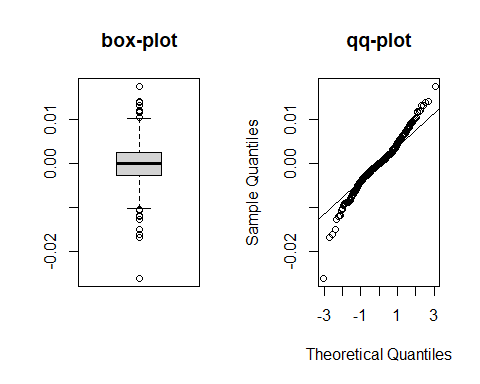
## [1] 0.003646628

Observing the Residuals

# Normality (only for MLE, not for LSE and MoM)  
#win.graph(width=6,height=6)  
hist(adj\_price.res,main="Histogram of the residuals",xlab="",freq=F,  
 xlim=c(-0.03,0.03),ylim=c(),breaks=20)  
lines(density(adj\_price.res),col="blue",lwd=3)  
zz=seq(-0.03,0.03,length=400)  
f.zz=dnorm(zz,mean(adj\_price.res),sd(adj\_price.res))  
lines(zz,f.zz,col="red",lwd=2)



#win.graph(width=12,height=6)  
par(mfrow=c(1,2))  
boxplot(adj\_price.res,main="box-plot")  
qqnorm(adj\_price.res,main="qq-plot")  
qqline(adj\_price.res)



shapiro.test(adj\_price.res) #Shapiro-Wilk Normality - independent data!

##   
## Shapiro-Wilk normality test  
##   
## data: adj\_price.res  
## W = 0.97446, p-value = 7.505e-07

#The residuals are Normal if the p-value is above 0.05. This is not the case. The residuals are not normal.   
  
library (tseries)  
jarque.bera.test(adj\_price.res) #The data are Normal if the p-value is above 0.05. Here we have a really low p-value

##   
## Jarque Bera Test  
##   
## data: adj\_price.res  
## X-squared = 94.473, df = 2, p-value < 2.2e-16

acf(adj\_price.res)

Residuals clearly do not follow a normal distribution. We already could state that since we proved before that the original data are not normally distributed. Although, we can observe by the acf plot that there is no systematic pattern or relationship between the residuals at different time points. This is a desirable property in statistical modeling, as it suggests that the model is adequately capturing the underlying patterns and trends in the data.

FORECAST STEP

In order to check the goodness of the one-step ahead error, we cut the last 5 observations of adj\_price to check how good the forecast actually is, by forecasting the values that we cut

#initialize 'nn' which is the total number of data that we have, plus the number of data that we want to predict.   
nn=length(adj\_price)  
n=nn-5  
  
adj\_price.for=rep(NA,times=nn)  
adj\_price.for[n+1]=est.phi1\*adj\_price[n]  
  
for (t in (n+2):nn) {  
 adj\_price.for[t]=est.phi1\*adj\_price.for[t-1]  
}  
  
adj\_price.for

## [1] NA NA NA NA NA NA  
## [7] NA NA NA NA NA NA  
## [13] NA NA NA NA NA NA  
## [19] NA NA NA NA NA NA  
## [25] NA NA NA NA NA NA  
## [31] NA NA NA NA NA NA  
## [37] NA NA NA NA NA NA  
## [43] NA NA NA NA NA NA  
## [49] NA NA NA NA NA NA  
## [55] NA NA NA NA NA NA  
## [61] NA NA NA NA NA NA  
## [67] NA NA NA NA NA NA  
## [73] NA NA NA NA NA NA  
## [79] NA NA NA NA NA NA  
## [85] NA NA NA NA NA NA  
## [91] NA NA NA NA NA NA  
## [97] NA NA NA NA NA NA  
## [103] NA NA NA NA NA NA  
## [109] NA NA NA NA NA NA  
## [115] NA NA NA NA NA NA  
## [121] NA NA NA NA NA NA  
## [127] NA NA NA NA NA NA  
## [133] NA NA NA NA NA NA  
## [139] NA NA NA NA NA NA  
## [145] NA NA NA NA NA NA  
## [151] NA NA NA NA NA NA  
## [157] NA NA NA NA NA NA  
## [163] NA NA NA NA NA NA  
## [169] NA NA NA NA NA NA  
## [175] NA NA NA NA NA NA  
## [181] NA NA NA NA NA NA  
## [187] NA NA NA NA NA NA  
## [193] NA NA NA NA NA NA  
## [199] NA NA NA NA NA NA  
## [205] NA NA NA NA NA NA  
## [211] NA NA NA NA NA NA  
## [217] NA NA NA NA NA NA  
## [223] NA NA NA NA NA NA  
## [229] NA NA NA NA NA NA  
## [235] NA NA NA NA NA NA  
## [241] NA NA NA NA NA NA  
## [247] NA NA NA NA NA NA  
## [253] NA NA NA NA NA NA  
## [259] NA NA NA NA NA NA  
## [265] NA NA NA NA NA NA  
## [271] NA NA NA NA NA NA  
## [277] NA NA NA NA NA NA  
## [283] NA NA NA NA NA NA  
## [289] NA NA NA NA NA NA  
## [295] NA NA NA NA NA NA  
## [301] NA NA NA NA NA NA  
## [307] NA NA NA NA NA NA  
## [313] NA NA NA NA NA NA  
## [319] NA NA NA NA NA NA  
## [325] NA NA NA NA NA NA  
## [331] NA NA NA NA NA NA  
## [337] NA NA NA NA NA NA  
## [343] NA NA NA NA NA NA  
## [349] NA NA NA NA NA NA  
## [355] NA NA NA NA NA NA  
## [361] NA NA NA NA NA NA  
## [367] NA NA NA NA NA NA  
## [373] NA NA NA NA NA NA  
## [379] NA NA NA NA NA NA  
## [385] NA NA NA NA NA NA  
## [391] NA NA NA NA NA NA  
## [397] NA NA NA NA NA NA  
## [403] NA NA NA NA NA NA  
## [409] NA NA NA NA NA NA  
## [415] NA NA NA NA NA NA  
## [421] NA NA NA NA NA 0.03580274  
## [427] 0.02502335 0.01748939 0.01222374 0.00854345

Here, there are the 5 predicted values. As we can see in the formula, they are estimated using the estimation of the parameter phi computed in the estimation parameters step.

adj\_price.er.for=rep(NA,times=nn) # one-step-ahead forecast error  
  
for (t in (n+1):nn) {  
 adj\_price.er.for[t]=adj\_price[t]-adj\_price.for[t]  
}  
  
RMSE.for=sqrt(mean(adj\_price.er.for^2,na.rm=TRUE))  
  
RMSE.for

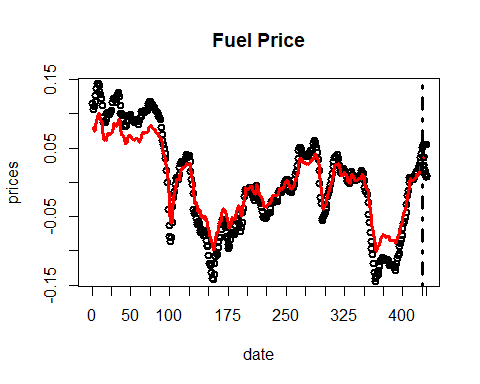
## [1] 0.03648444

MAE.for=mean(abs(adj\_price.er.for),na.rm=TRUE)  
  
MAE.for

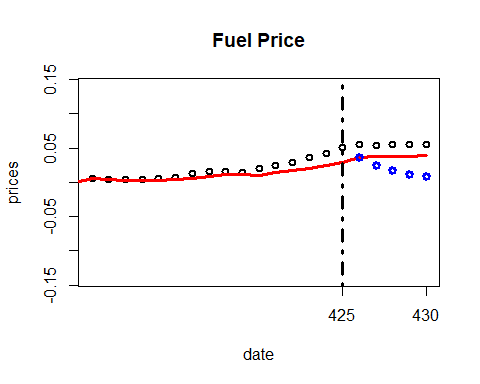
## [1] 0.03503338

given our previous results such as the non normality of the data and the residuals, we already expected MAE for values not similar with the other MAE. To be satisfied with the forecast error we should expect very similar MAE and MAE.for, unfortunately this is not the case.

val\_x=c(0,(1:20)\*25,430)  
val\_y=c()  
#win.graph(width=20,height=10)  
plot(adj\_price,main="Fuel Price",xlab="date",ylab="prices",xlim=c(0,430),ylim=c(-0.14,0.14),  
 xaxt="n",yaxt="n",type="p",lwd=2)  
axis(1,val\_x,val\_x)  
axis(2,val\_y,val\_y)  
lines(adj\_price.fit,col="red",lwd=3)  
lines(adj\_price.for,type="p",lwd=3)  
abline(v=n,col="black",lwd=3,lty=4)



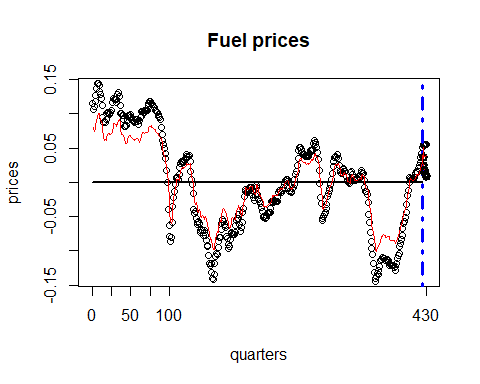
val\_x=c(0,(1:20)\*25,430)  
val\_y=c()  
#win.graph(width=20,height=10)  
plot(adj\_price,main="Fuel Price",xlab="date",ylab="prices",xlim=c(410,430),ylim=c(-0.14,0.14),  
 xaxt="n",yaxt="n",type="p",lwd=2)  
axis(1,val\_x,val\_x)  
axis(2,val\_y,val\_y)  
lines(adj\_price.fit,col="red",lwd=3)  
lines(adj\_price.for,type="p",col="blue",lwd=3)  
abline(v=n,col="black",lwd=3,lty=4)



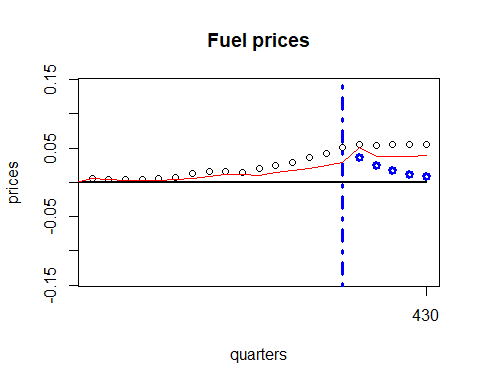
As we can see in the graph, the 5 predicted values are after the vertical line. The predicted values are the blue dots, while the actual values are the black dots. As we can see, and as the MAE analysis showed, the forecast is not reliable.

Let’s try to fit our prediction, by giving continuity to the fitted values that we found above.

adj\_price.fit[n+1]=adj\_price[n] # just for the plot  
  
val\_x=c(0,(1:4)\*25,430)  
val\_y=c()  
#win.graph(width=12,height=6)  
plot(adj\_price,main="Fuel prices",xlab="quarters",ylab="prices",xlim=c(0, 430),ylim=c(-0.14,0.14),  
 type="p",xaxt="n",yaxt="n")  
axis(1,val\_x,val\_x)  
axis(2,val\_y,val\_y)  
lines(rep(mean(adj\_price,na.rm=TRUE),times=n),col="black",lwd=2)  
abline(v=n,col="blue",lwd=3,lty=4)  
lines(adj\_price.for,type="l",col="red",lwd=1)  
lines(adj\_price.for,type="p",lwd=3)  
lines(adj\_price.fit,type="l",col="red",lwd=1)



val\_x=c(0,(1:4)\*25,430)  
val\_y=c()  
#win.graph(width=12,height=6)  
plot(adj\_price,main="Fuel prices",xlab="quarters",ylab="prices",xlim=c(410, 430),ylim=c(-0.14,0.14),  
 type="p",xaxt="n",yaxt="n")  
axis(1,val\_x,val\_x)  
axis(2,val\_y,val\_y)  
lines(rep(mean(adj\_price,na.rm=TRUE),times=nn),col="black",lwd=2)  
abline(v=n,col="blue",lwd=3,lty=4)  
lines(adj\_price.for,type="p",col="blue",lwd=3)  
lines(adj\_price.fit,type="l",col="red",lwd=1)



Let’s compute the prediction intervals. They are computed using Var(one-step-ahead forecast errors).

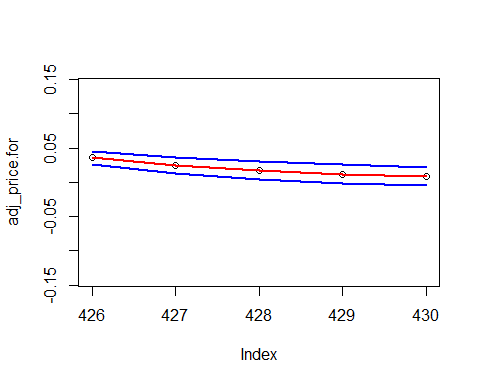
psi=rep(0,times=nn-n)  
psi.0=1  
psi[1]=est.phi1  
for (j in 2:5) {  
 psi[j]=est.phi1\*psi[j-1]  
}  
psi

## [1] 0.6989229 0.4884932 0.3414191 0.2386256 0.1667809

var.er=rep(0,times=nn-n)  
var.er[1]=est.sigma2  
  
for (j in 2:5) {  
 var.er[j]=est.sigma2\*(psi.0+sum(psi[1:(j-1)]^2))  
}  
var.er

## [1] 2.509737e-05 3.735727e-05 4.334614e-05 4.627167e-05 4.770077e-05

# Left (lower bound)  
left.er=rep(NA,times=nn)  
for (t in (n+1):nn) {  
 left.er[t]=adj\_price.for[t]-1.960\*sqrt(var.er[t-n]) #P(|Z|<1.960)=0.95 Z=Standard Normal  
} #P(|Z|<2.576)=0.99  
# Right (upper bound)  
right.er=rep(NA,times=nn)  
for (t in (n+1):nn) {  
 right.er[t]=adj\_price.for[t]+1.960\*sqrt(var.er[t-n])  
}  
  
val\_y=c()  
#win.graph(width=12,height=6)  
plot(adj\_price.for,xlim=c(n+1,nn),type="p",yaxt="n",ylim=c(-0.14,0.14),yaxt="n")  
axis(2,val\_y,val\_y)  
lines(adj\_price.for,type="l",col="red",lwd=2)  
lines(left.er,type="l",col="blue",lwd=2)  
lines(right.er,type="l",col="blue",lwd=2)



CONCLUSIONS

We started by having all non normal distributions, and we chose the one that looked the least complicated, i.e. the one that looked like a normal distribution the most. After that, we tried to normalize data by appliying a log transformation and then by centralizing the distribution. We started to analyze the series by assuming stationarity of the data. As we saw by the model comparison, the best model that fitted our data was AR(2) which was the model that provided us the lowest BIC. In order to understand if our assumption of stationarity was right, we executed an Augmented Dickey Fuller test that showed that our data was non stationary. After that, we applied an ARIMA model, specifically an ARIMA(1,1,0), which is suitable for our data because it executes the integration operation that actually made our data stationary. In the end we forecasted the data and observed that the forecasting process for this model with these data is not reliable since MAE and MAE.for are not similar numbers.