Paul-Antoine Fruchtenreich & Julien Peignon
2A ENSAE

Assignment

Linear time series

---

1 The data

1.1 Question 1

The chosen series (X_t) is the CVS-CJO index of industrial production - Manufacture of agricultural and forestry machinery; monthly data starting from January 1990. The series is in base 100 in 2015; the values represent the monthly variations of agricultural and forestry machinery manufacture compared to 2015. Furthermore, the series has been corrected for seasonal variations and working days. We drop the two last values to make predictions out of the sample in Section 3.

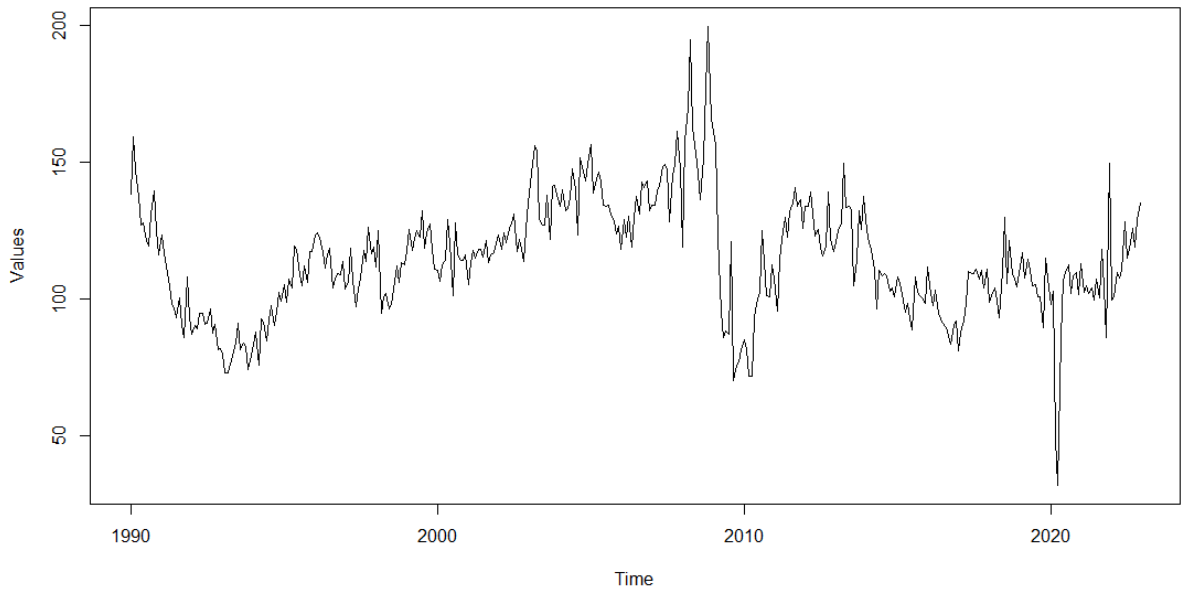


Figure 1: CVS-CJO index of industrial production - Manufacture of agricultural and forestry machinery

1.2 Question 2

Before fitting an ARMA model to a time series, one has to check that stationarity is a plausible assumption. If data exhibit visible deviations from stationarity, preliminary transformations are called for. In this question, we will perform several tests to determine if we have sufficient evidence

to believe that our series is stationary[1].

First, we do not need to check for seasonality because the series has already been corrected. To calibrate the parameters of our tests, we need to determine if our series exhibits a time trend and admits a mean. To do so, we regress the series on a constant and time, and we conclude, thanks to Table 1, that there is no time trend but a significant constant. Indeed, the estimated coefficient associated with the time variable is not statistically significant at any usual levels. Therefore, we perform an Augmented Dickey-Fuller (ADF) test with only a constant in order to test if whether or not it is likely that our series is stationary. For our ADF test to be valid, we need to consider the autocorrelation of residuals up to five lags (see the code for more details). Our ADF test rejects the null hypothesis of non-stationarity with a p-value of 0,041. Therefore, we have strong reasons to believe that our series is stationary. To confirm the ADF test, we also perform a Phillips-Perron and a KPSS test. The Philipps-Perron test confirms the previous result. However, the KPSS test rejects the null hypothesis of stationarity (the hypothesis are inverted regarding the ADF test) with a p-value of 0,086. The different tests do not provide an unambiguous result; we decide to differentiate our series.

We differentiate our series, *i.e.* we create $\Delta X_t = X_t - X_{t-1}$, and we regress the series on a constant and time (Table 2). Both the constant and the time trend are not statistically significant. Therefore, we perform an ADF test with no constant or time trend. We consider autocorrelation of residuals up to 4 lags (see the code for more details). The ADF test rejects the null hypothesis of non-stationarity with a p-value $< 0,01$. Again, we have strong reasons to believe that the differentiated series is stationary. Both the Phillips-Perron and the KPSS tests confirm these results. The KPSS test does not reject the null hypothesis of stationarity with a p-value $> 0,10$.

In what follows, we will work on the differentiated series, *i.e.* $\Delta X_t = X_t - X_{t-1}$, considering that the latter is stationary.

1.3 Question 3

Figure 2 represents the chosen series before and after differentiation.

2 ARMA models

2.1 Question 4

To choose the parameters of an ARMA model for our differentiated series, we plot the partial and total autocorrelograms of the series on 2 periodicities (24 lags). They are displayed on Figure 6.

[1] As we use ARMA models, we are interested in second-order stationarity. A series (X_t) is second-order stationary if

1. $\mathbb{E}\{\|X_t\|^2\} < \infty$
2. $\mathbb{E}\{X_t\}$ is independent of t
3. $\forall h$, $Cov(X_t, X_{t+h})$ is independent of t

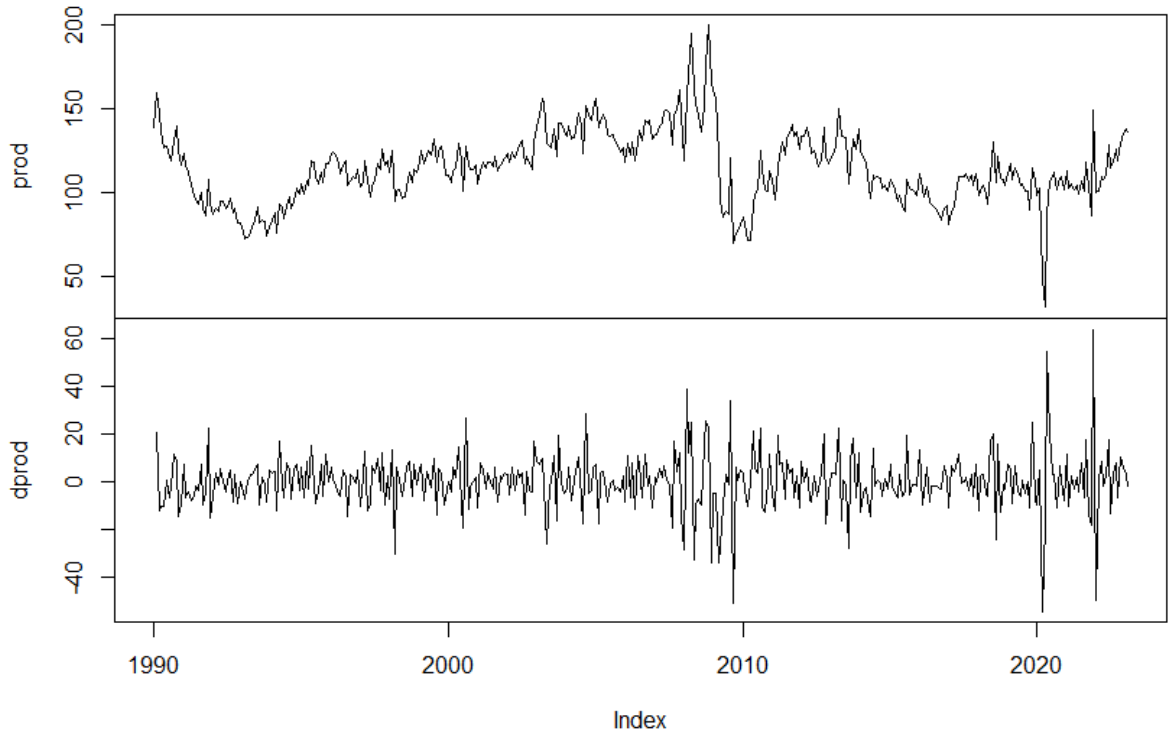


Figure 2: Series before and after differentiation

The total serial correlation functions are significant (*i.e.*, greater than the bounds $\pm 1.96/\sqrt{n}$ of the confidence interval of the test of nullity of the autocorrelation at 95%) until $q^* = 2$, and partial serial correlations until $p^* = 5$. We make a choice to ignore the peaks far after $q^* = 2$ and $p^* = 5$.

If our corrected series follow an ARMA(p, q), then the orders p and q are necessarily such that $p \leq p^*$ and $q \leq q^*$.

We fit all the ARMA and compute each combination's Bayesian Information Criterion and Akaike Information Criterion; the results are shown on Table 3 and Table 4. BIC and AIC are minimized by two different models: ARMA$(5, 0)$ and ARMA$(1, 1)$. The estimated coefficients of these two models are displayed in Table 5 and Table 6, respectively. All coefficients are statistically significant at all usual levels; both models are well-adjusted to our corrected series.

Finally, to ensure these models' validity, we have to check that the residuals are not serially correlated. We perform a Ljung-Box test with the null hypothesis being the joint nullity of the serial correlations until a given order k. We test for autocorrelation of residuals for all $k \in \{1, 2, ..., 24\}$. The results of the Ljung-Box tests are displayed in Table 7. For both models, all the p-values stay above 0.05, meaning that the null hypothesis of joint nullity of the serial correlations is not rejected for all k. The absence of autocorrelation is never rejected. The models are valid.

In a nutshell, we selected two models, ARMA(5,0) and ARMA(1,1), with the following properties:

all their estimated coefficients are statistically significant, their residuals are not serially correlated, and they minimize either the AIC or the BIC. Thus, we have two well-adjusted and valid models that minimize one of the information criteria. To choose the best one, we keep the one that gives the best prediction in our dataset. We compare the adjusted R^2 of both models and select the model with the highest value in Table 8. Based on this methodology, the best model for our series is the ARMA$(5,0)$. It is the model we will be using in the following sections.

2.2 Question 5

According to the course, because our corrected series follows an ARMA$(5,0)$ and is the differentiation of the chosen series, our chosen series (X_t) follows an ARIMA$(5,1,0)$. The series $(\Delta X_t) = \big((1-B)X_t\big)$ is a causal ARMA$(5,0) = AR(5)$, where B is the backward operator, *i.e.* $BX_t = X_{t-1}$. In order to lighten the notations, we define $\Delta X_t \equiv Y_t$ in what follows. Therefore, we can write this model as

$$Y_t = \sum_{i=1}^{5} \phi_i Y_{t-i}$$

The coefficients $(\phi_i)_{i=1}^5$ have been estimated in Table 5. The explicit model is

$$Y_t = -0.320Y_{t-1} - 0.227Y_{t-2} - 0.117Y_{t-3} - 0.181Y_{t-4} - 0.147Y_{t-5} \tag{1}$$

By replacing Y_t by $X_t - X_{t-1}$, we can write the model for (X_t)

$$X_t = 0.680X_{t-1} + 0.093X_{t-2} + 0.110X_{t-3} - 0.064X_{t-4} + 0.034X_{t-5} + 0.147X_{t-6} \tag{2}$$

Figure 3 represents the chosen series (X_t) and the estimated ARIMA$(5,1,0)$ of Equation 2. As can be seen, the estimated series matches almost perfectly the true series.

In order to make predictions, we want our model to be a canonical ARMA, *i.e.*, causal and invertible. First, our model is an AR(5). It is *de facto* invertible because it does not have moving averages. Therefore, the existence of a solution to our ARMA model only depends on the AR polynomial

$$\phi(z) = 1 - \phi_1 z - \phi_2 z^2 - \phi_3 z^3 - \phi_4 z^4 - \phi_5 z^5 \tag{3}$$

Thanks to the course, we know that a causal stationary solution exists if and only if

$$\phi(z) = 1 - \phi_1 z - \phi_2 z^2 - \phi_3 z^3 - \phi_4 z^4 - \phi_5 z^5 \neq 0, \quad \text{for all z such that } |z| \leq 1.$$

Thus, we have to check that the roots of the characteristic polynomial are greater than 1. Figure 7 shows the inverse of the roots of the characteristic polynomial and the unit circle. If the roots of the AR polynomial are greater than one, their inverses should lie within the unit circle. It is the case; the roots are greater than one. Our model is causal and, *a fortiori*, invertible. We can use it to make predictions, which we will be doing in Section 3.

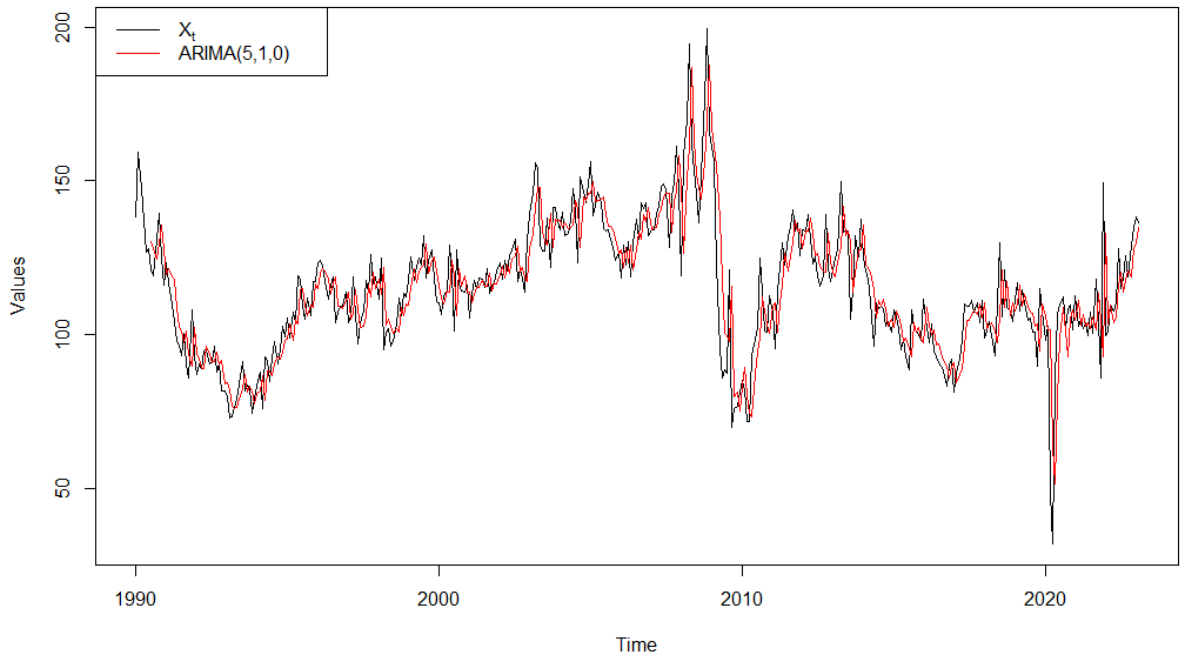


Figure 3: Plot of the series and our model ARIMA(5,1,0)

Finally, we want to check if our residuals could be considered as a weak white noise[2]. In anticipation of future questions, we will also ask whether they can be considered Gaussian. We plot the residuals of our model AR(5) in Figure 8, alongside their distribution and autocorrelograms. First, the residuals are centered around zero, and their variance looks constant, except during some well-defined crises: the subprimes, the Covid-19, and, more recently, the Ukraine crisis. We previously discussed that the residuals were not subject to serial correlation, shown in the ACF, except for one outlier, at the 24^{th} lag. Therefore, we have strong reasons to believe that the residuals of our ARMA(5,0) model are weak white noises.

Nonetheless, the residuals do not seem Gaussian; their histogram does not match the Gaussian distribution. Their distribution is more concentrated around their mean, zero, but exhibit too many values in the distribution tails. We draw a QQplot in Figure 9 to confirm these results further. The distribution is likely Gaussian except in the distribution tails, where the points are no longer aligned.

[2]A weak white noise is a sequence (ϵ_t) of uncorrelated variables with zero means, and a constant variance, *i.e.*, $\mathbb{E}(\epsilon_t) = 0$, $\mathbb{V}(\epsilon_t) = \sigma^2$, $Cov(\epsilon_t, \epsilon_s) = 0 \ \ \forall t \neq s$

3 Predictions

Let T be the length of our series (X_t), $T = 396$. We assume that the residuals of (X_t) are Gaussian.

3.1 Question 6

The best predictions for the future values X_{T+1} and X_{T+2} knowing the previous ones are

$$\hat{X}_{T+1|T} = 0.682X_T + 0.094X_{T-1} + 0.108X_{T-2} - 0.063X_{T-3} + 0.033X_{T-4} + 0.146X_{T-5}$$
$$\hat{X}_{T+2|T} = 0.682\hat{X}_{T+1|T} + 0.094X_T + 0.108X_{T-1} - 0.063X_{T-2} + 0.033X_{T-3} + 0.146X_{T-4}$$

Writing under matrix form, we have :

$$\hat{\mathbf{X}}_{T+1|T} = \begin{pmatrix} \hat{X}_{T+1|T} \\ \hat{X}_{T+2|T} \end{pmatrix} \text{ and } \mathbf{X}_{T+1} = \begin{pmatrix} X_{T+1} \\ X_{T+2} \end{pmatrix}$$

Then, computing the difference $\tilde{\mathbf{X}}_{T+1} = \mathbf{X}_{T+1} - \hat{\mathbf{X}}_{T+1|T}$:

$$\tilde{\mathbf{X}}_{T+1} = \begin{pmatrix} \tilde{X}_{T+1} \\ \tilde{X}_{T+2} \end{pmatrix} = \begin{pmatrix} X_{T+1} - \hat{X}_{T+1|T} \\ X_{T+2} - \hat{X}_{T+2|T} \end{pmatrix} = \begin{pmatrix} \epsilon_{T+1} \\ \epsilon_{T+2} + (1+\phi_1)\,\epsilon_{T+1} \end{pmatrix}$$

We can compute the variance of the residuals by using the fact that they are uncorrelated and with constant variance equal to σ^2:

$$\mathbb{V}[\epsilon_{T+1}] = \sigma^2 \; \mathbb{V}[\epsilon_{T+2} + (1+\phi_1)\,\epsilon_{T+1}] = \sigma^2 \left(1 + (1+\phi_1)^2\right)$$

Because the residuals are Gaussian and independent by hypothesis, the vector $\tilde{\mathbf{X}}_{T+1}$ follows a $\mathcal{N}(0, \mathbf{\Sigma})$ where

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma^2 & \sigma^2\,(1+\phi_1) \\ \sigma^2\,(1+\phi_1) & \sigma^2\left(1 + (1+\phi_1)^2\right) \end{bmatrix}$$

Then, since $\mathbf{\Sigma}$ is invertible if and only if its determinant is non-zero and $|\mathbf{\Sigma}| = \sigma^4 > 0$, we have that:

$$\tilde{\mathbf{X}}'_{T+1}\mathbf{\Sigma}^{-1}\tilde{\mathbf{X}}_{T+1} \sim \chi^2(2)$$

because

$$\tilde{\mathbf{X}}'_{T+1}\mathbf{\Sigma}^{-1}\tilde{\mathbf{X}}_{T+1} = \frac{\epsilon^2_{T+1}}{\sigma^2} + \frac{\epsilon^2_{T+2}}{\sigma^2}$$

which is, under the assumption of independent Gaussian residuals, the sum of two independent squared-centered reduced normal laws. Then

$$\mathbb{P}[\tilde{\mathbf{X}}'_{T+1}\hat{\mathbf{\Sigma}}^{-1}\tilde{\mathbf{X}}_{T+1} \leq q_{\chi^2(2)}(\alpha)] = \alpha$$

where $q_{\chi^2(2)}(\alpha)$ is the α^{th} quantile of a χ^2 law with two degrees of freedom.
The confidence region for the values of (X_{T+1}, X_{T+2}) at the level α is

$$R_\alpha = \left\{ \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \in \mathbb{R}^2 \setminus \begin{pmatrix} x_1 - \hat{X}_{T+1|T} \\ x_2 - \hat{X}_{T+2|T} \end{pmatrix}' \hat{\mathbf{\Sigma}}^{-1} \begin{pmatrix} x_1 - \hat{X}_{T+1|T} \\ x_2 - \hat{X}_{T+2|T} \end{pmatrix} \leq q_{\chi^2(2)}(1-\alpha) \right\}$$

From the univariate point of view, with the estimation of the coefficient ϕ_1 and the variance $\hat{\sigma}^2$ of the residuals, we have the following confidence intervals at the level 95% for X_{T+1} and X_{T+2}:

$$\begin{gathered} [\hat{X}_{T+1|T} - 1.96\hat{\sigma} \ ; \ \hat{X}_{T+1|T} + 1.96\hat{\sigma}] \\ [\hat{X}_{T+2|T} - 1.96\hat{\sigma}\sqrt{1+(1+\phi_1)^2} \ ; \ \hat{X}_{T+2|T} + 1.96\hat{\sigma}\sqrt{1+(1+\phi_1)^2}] \end{gathered}$$

With the values of ϕ_1 and $\hat{\sigma}^2$, we can compute the final univariate confidence intervals at 95% for X_{T+1} and X_{T+2} :

$$\begin{gathered} [108.08 \ ; \ 151.40] \\ [102.80 \ ; \ 155.20] \end{gathered}$$

3.2 Question 7

To obtain this confidence region, we needed to make the following assumptions :

- The model is perfectly known, *i.e.*, the estimated coefficients are the real ones or at least convergent.
- The residuals are Gaussian and independent, with known variance.

Indeed, the innovation process is Gaussian *iid*, and our AR(5) model is causal and invertible so that we can compute the best linear prediction of X_{T+1} and X_{T+2}. However, the second assumption is an essential limitation of our model. As discussed above, the distribution of the residuals does not correspond to that of a Gaussian random variable. Furthermore, we assumed that our residuals' variance was known. To be more rigorous, we should estimate it, and it would bring a new layer of approximation for the confidence intervals.

3.3 Question 8

Figure 4 represents our series and the estimations $\hat{X}_{T+1|T}$ and $\hat{X}_{T+2|T}$ alongside their true values and their univariate confidence intervals at level 95%. We can see that the true values X_{T+1} and X_{T+2} lie within the confidence intervals. Therefore, our estimations are correct.

On the other side, Figure 5 represents the bivariate confidence region we calculated in the previous question, which takes the form of an ellipse. The ellipse center is $(\hat{X}_{T+1}, \hat{X}_{T+2}) = (129.74, 129.00)$. The red point is $(X_{T+1}, X_{T+2}) = (138.27, 136.51)$, the true values that have been dropped out of the series at the beginning in order to make predictions out of the sample. Again, the true values lie within the confidence region.

Nonetheless, as discussed, these confidence intervals were built upon the assumption that the residuals were Gaussian. In our specific case, this assumption does not seem realistic. Thus, it would be unwise to draw any conclusions from these graphics.

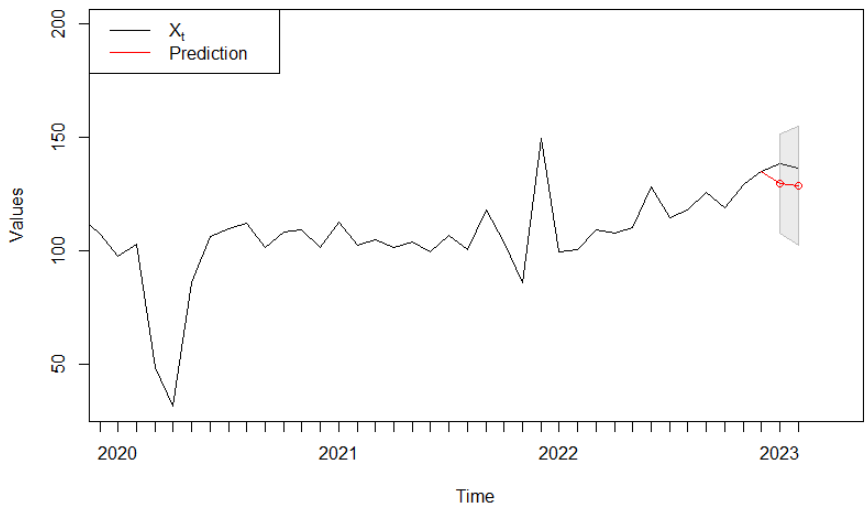


Figure 4: (X_t) and predictions of X_{T+1} and X_{T+2} with the univariate confidence intervals at 95%

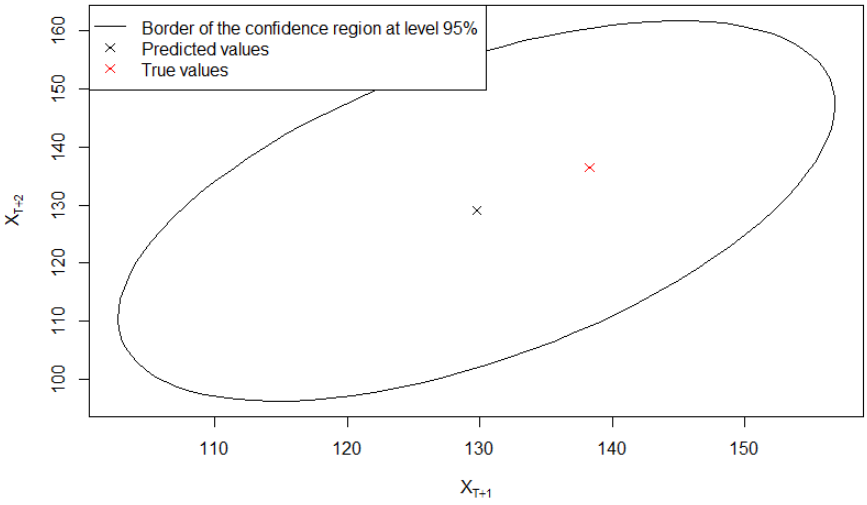


Figure 5: (X_{T+1}, X_{T+2}) and the bivariate confidence region at 95% of this point based on $\hat{X}_{T+1}$ and $\hat{X}_{T+2}$

3.4 Question 9

Let (Y_t) be a stationnary series, avalaible from $t = 1$ to $t = T$. We assume that Y_{T+1} is available quicker than X_{T+1}. Intuitively, this new information Y_{T+1} can be useful to improve the prediction of X_{T+1} if the linear prediction of X_{T+1} changes before and after adding Y_{T+1} in our dataset. More rigorously, we can say that Y_{T+1} improves the prediction of X_{T+1} if Y_{T+1} causes instantaneously X_{T+1} in the Granger sense :

$$\hat{X}_{T+1|\{X_u,Y_u,u\leq T\}\cup\{Y_{T+1}\}} \neq \hat{X}_{T+1|\{X_u,Y_u,u\leq T\}}$$

To test this causality, in the Granger sense, one can use a Wald test. Let $X_t \in \mathbb{R}^{d_1}$ and $Y_t \in \mathbb{R}^{d_2}$ and suppose $(X_t^{'}, Y_t^{'})$ follows a VAR(p). Under regularity assumptions, the LS estimator of $A = [A_1, ..., A_p]$ satisfies

$$\sqrt{n}(vec\hat{A} - vecA) \xrightarrow[n\to\infty]{\mathcal{L}} \mathcal{N}(0, \Sigma_A) \tag{4}$$

The non-causality assumption from X to Y writes, H_0: $Rvec\, A = 0$. Under H_0, we thus have

$$\sqrt{n}R\, vec\hat{A} \xrightarrow[n\to\infty]{\mathcal{L}} \mathcal{N}(0, R\Sigma_A R^{'}) \tag{5}$$

and, if $\hat{\Sigma}_A$ is any consistent estimator of Σ_A, the quadratic form

$$S = n(vec\hat{A}^{'})R^{'}(R\hat{\Sigma}_A R^{'})^{-1}Rvec\hat{A}$$

follows approximately a $\chi^2_{pd_1d_2}$.
Thus, the assumption that X does not cause Y is rejected, at level α, if

$$S > \chi^2_{pd_1d_2}(1-\alpha)$$

where $\chi^2_k(1-\alpha)$ is the $(1-\alpha)$-quantile of χ^2_k.

Appendix

Table 1: Regression of the series on time and a constant

	Dependent variable:
	(X_t)
Time	0.004
	(0.009)
Constant	113.918^{***}
	(2.133)
Observations	398
R^2	0.0004
Adjusted R^2	-0.002
Residual Std. Error	21.234 (df = 396)
F Statistic	0.143 (df = 1; 396)
Note:	*p<0.1; **p<0.05; ***p<0.01

Table 2: Regression of the differentiated series on time and a constant

	Dependent variable:
	$(X_t - X_{t-1})$
Time	0.002
	(0.005)
Constant	-0.354
	(1.193)
Observations	397
R^2	0.0003
Adjusted R^2	-0.002
Residual Std. Error	11.821 (df = 395)
F Statistic	0.114 (df = 1; 395)
Note:	*p<0.1; **p<0.05; ***p<0.01

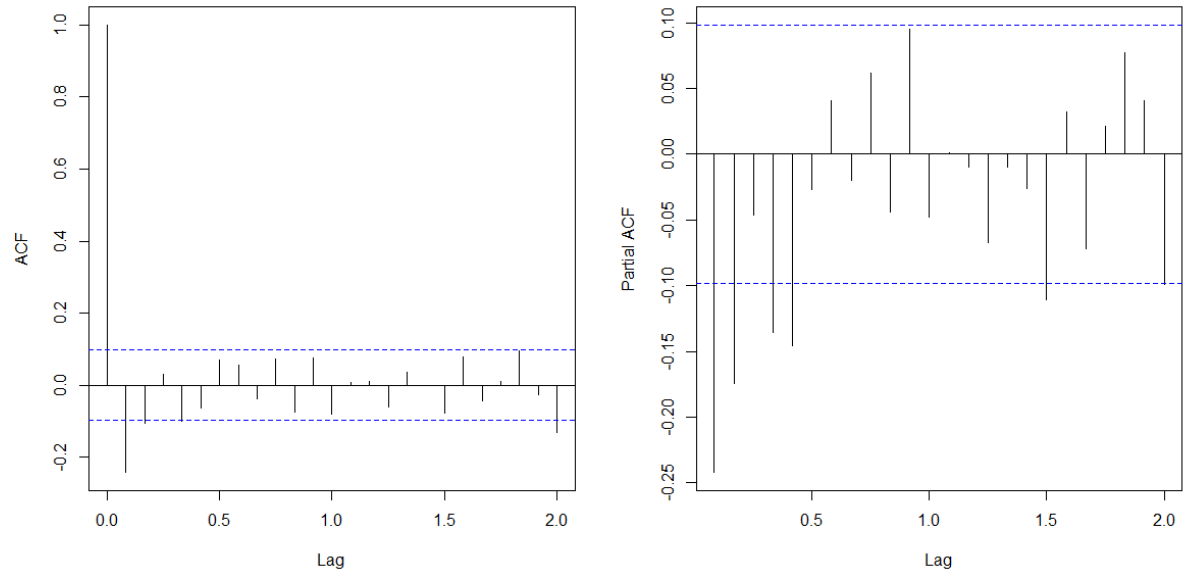


Figure 6: Total and partial autocorrelation functions of the differentiated series

Table 3: AIC

	q=0	q=1	q=2
p=0	3074.17	3041.27	3034.15
p=1	3052.15	3033.12	3035.12
p=2	3041.84	3035.12	3031.68
p=3	3043.04	3034.61	3033.63
p=4	3037.72	3033.89	3033.42
p=5	**3031.17**	3032.99	3034.66

Table 4: BIC

	q=0	q=1	q=2
p=0	3078.15	3049.23	3046.08
p=1	3060.10	**3045.06**	3051.03
p=2	3053.78	3051.03	3051.57
p=3	3058.96	3054.51	3057.50
p=4	3057.62	3057.77	3061.27
p=5	3055.04	3060.84	3066.49

Table 5: ARMA(5,0)

	Dependent variable:
	$(X_t - X_{t-1})$
ϕ_1	-0.320^{***}
	(0.050)
ϕ_2	-0.227^{***}
	(0.052)
ϕ_3	-0.117^{**}
	(0.053)
ϕ_4	-0.181^{***}
	(0.052)
ϕ_5	-0.147^{***}
	(0.050)
Observations	395
Log Likelihood	$-1{,}509.583$
σ^2	122.093
Akaike Inf. Crit.	3,031.17
Bayesian Inf. Crit.	3,055.04
Note:	*p<0.1; **p<0.05; ***p<0.01

Table 6: ARMA(1,1)

	Dependent variable:
	$(X_t - X_{t-1})$
ϕ_1	0.388^{***}
	(0.102)
θ_1	-0.709^{***}
	(0.078)
Observations	395
Log Likelihood	$-1{,}513.561$
σ^2	124.607
Akaike Inf. Crit.	3,033.12
Bayesian Inf. Crit.	3,045.06
Note:	*p<0.1; **p<0.05; ***p<0.01

Table 8: Choice of the best model

	Adjusted R^2
ARMA(5,0)	0.102
ARMA(1,1)	0.097

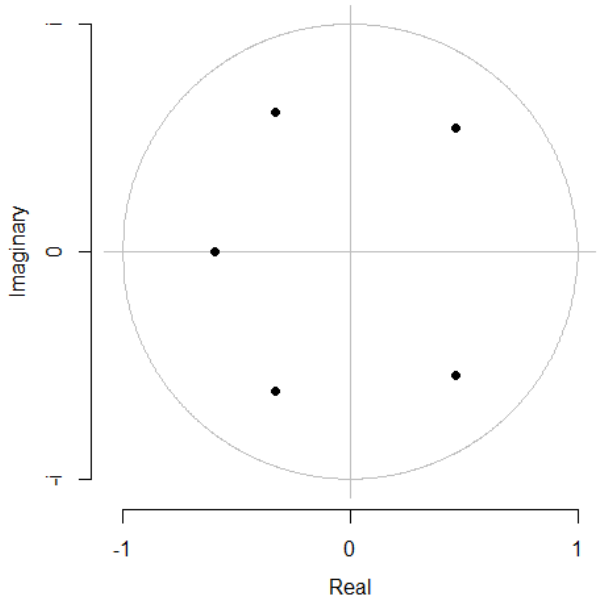


Figure 7: Inverse roots of ARMA(5,0) characteristic polynomial

Table 7: Ljung-Box tests for the ARMA(5,0) and ARMA(1,1)

	p-value	
k	ARMA(5,0)	ARMA(1,1)
1		
2		0.86
3		0.65
4		0.33
5	0.91	0.35
6	0.98	0.25
7	0.78	0.20
8	0.85	0.28
9	0.74	0.23
10	0.68	0.23
11	0.58	0.24
12	0.45	0.18
13	0.49	0.23
14	0.57	0.28
15	0.46	0.23
16	0.54	0.29
17	0.62	0.34
18	0.58	0.26
19	0.60	0.26
20	0.64	0.29
21	0.67	0.35
22	0.61	0.28
23	0.64	0.30
24	0.24	0.09

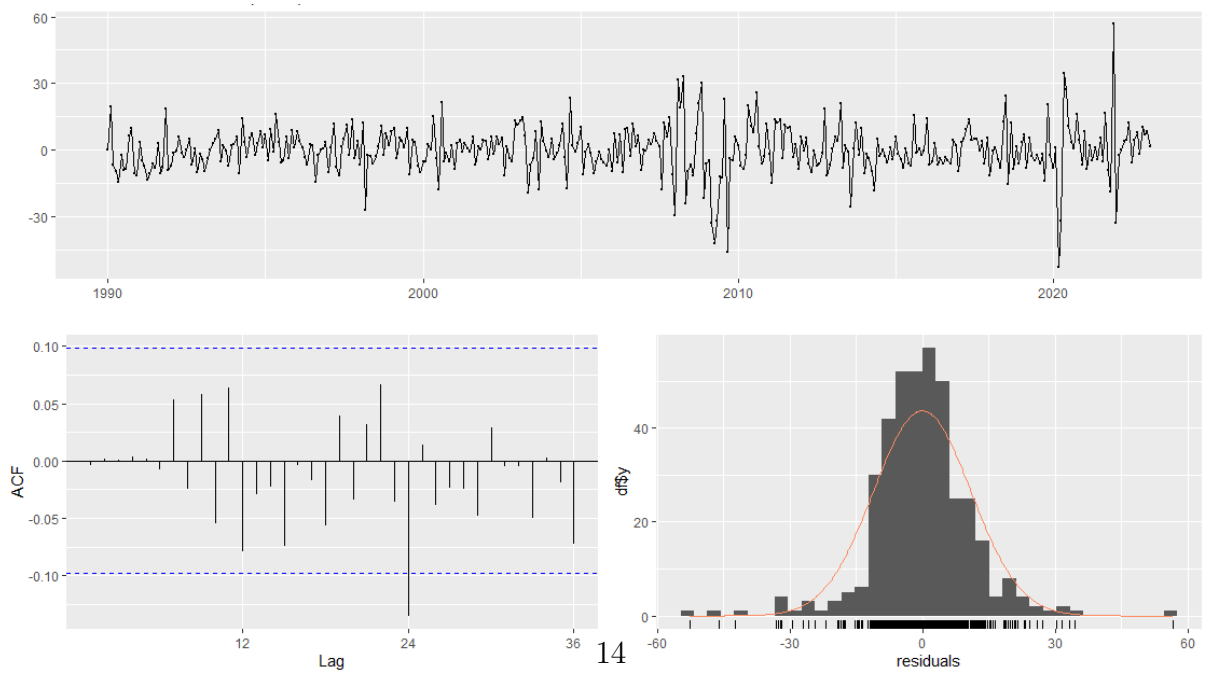


Figure 8: Residuals of the ARMA(5,0)

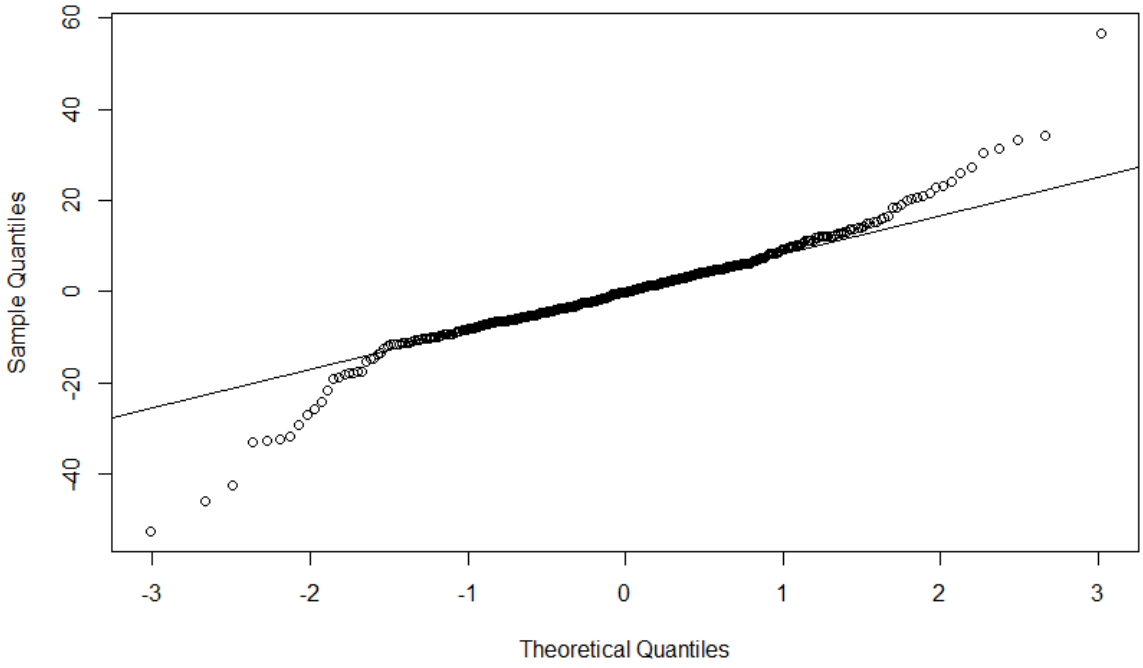


Figure 9: QQplot of the residuals of the ARMA(5,0)

Linear Time Series

Paul-Antoine Fruchtenreich & Julien Peignon

16/05/2023

SET-UP

Set working directory

```
setwd(dirname(rstudioapi::getActiveDocumentContext()$path))
```

Downloading packages

```
list.of.packages <- c("readr", "zoo", "tseries", "stargazer", "fUnitRoots", "dplyr",
                      "aTSA", "xtable", "forecast", "ellipse", "graphics", "knitr",
                      "rmarkdown", "markdown")
new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()[,"Package"])]
if(length(new.packages)) install.packages(new.packages, repos = "http://cran.us.r-project.org")

invisible(lapply(list.of.packages, library, character.only = TRUE))
```

Reset environment

```
rm(list=ls())
```

IMPORTING & TRANSFORMING THE DATA

Importing the data

```
serie <- read.csv("valeurs_mensuelles.csv", sep=";")
```

Creating the serie of dates

```
dates <- as.yearmon(seq(from=1990+0/12,to=2023+1/12,by=1/12))
```

Inverting the time values

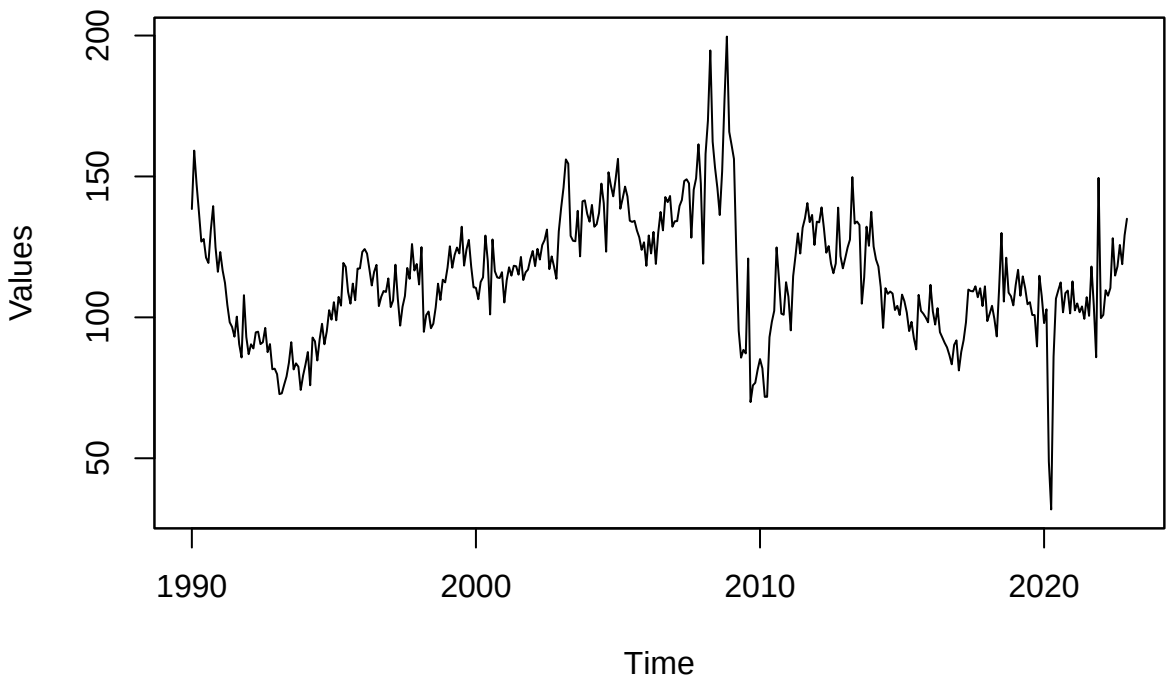
```
serie <- serie[dim(serie)[1]:1,]
serie$Time <- seq(1,nrow(serie),1)
```

Transforming our serie into a zoo object

```
prod <- zoo(serie$Values, order.by=dates)
```

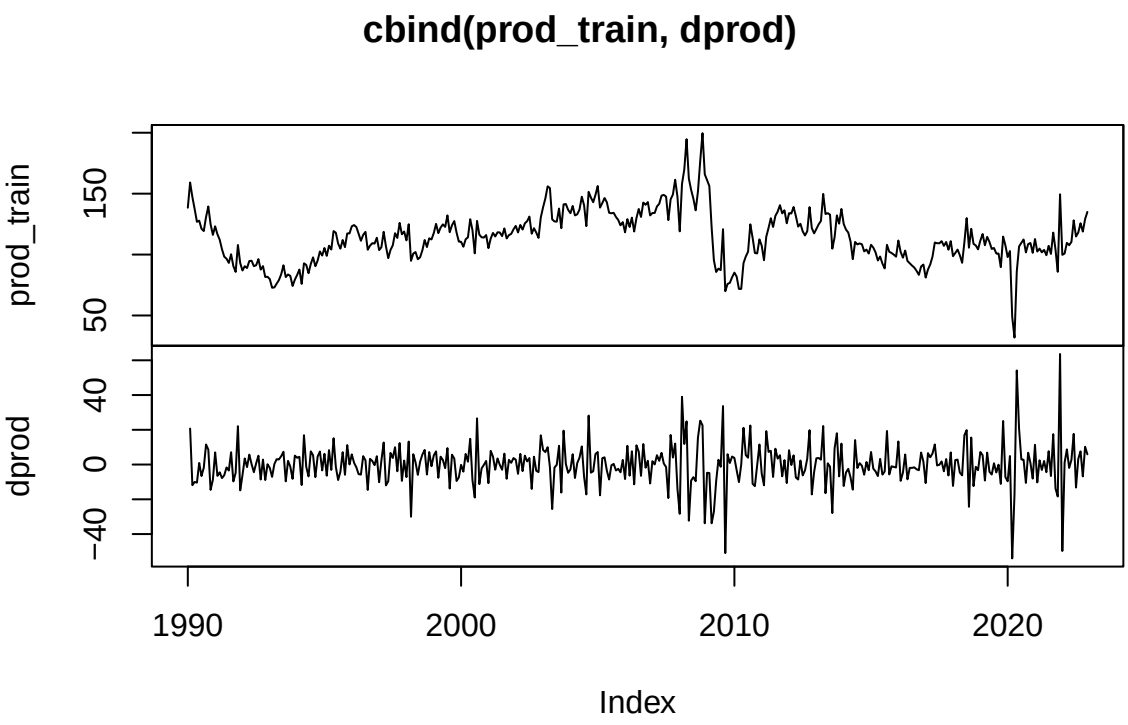
We drop the last two observations for predictions

```
prod_train <- prod[1:(length(prod)-2)]
plot(prod_train, xlab = "Time", ylab = "Values")
```



We create the differenciated serie

```
dprod <- diff(prod_train,1)
plot(cbind(prod_train,dprod))
```



PART 1 : STATIONARITY

We regress the values of our serie on time to check if there is a trend

```
reg <- lm(Values ~ Time, data=serie)
stargazer(reg, type="text")
```

```
##
## ================================================
##                         Dependent variable:
##                     ---------------------------
##                                Values
## ------------------------------------------------
## Time                            0.004
##                                (0.009)
##
## Constant                      113.918***
##                                (2.133)
##
## ------------------------------------------------
## Observations                     398
## R2                              0.0004
## Adjusted R2                     -0.002
## Residual Std. Error       21.234 (df = 396)
```

```
## F Statistic             0.143 (df = 1; 396)
## ================================================
## Note:               *p<0.1; **p<0.05; ***p<0.01
```

Time isn't significative, there is no trend, so we do the adf test with only a constant: "c"

```
adf <- adfTest(prod_train, lag=0, type="c")
adf
```

```
##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 0
##   STATISTIC:
##     Dickey-Fuller: -5.7793
##   P VALUE:
##     0.01
##
## Description:
##  Sat May 27 19:17:52 2023 by user: peign
```

The unit root hypothesis is not rejected However, this test isn't valid because we didn't consider residuals autocorrelation

We define two functions to determine the number of lags we have to consider to get rid of autocorrelation

Qtest

```
Qtests <- function(series, k, fitdf=0) {
  pvals <- apply(matrix(1:k), 1, FUN=function(l) {
    pval <- if (l<=fitdf) NA else Box.test(series, lag=l, type="Ljung-Box", fitdf=fitdf)$p.value
    return(c("lag"=l,"pval"=pval))
  })
  return(t(pvals))
}
```

adfTest

```
adfTest_valid <- function(series,kmax,type){ #ADF tests until no more autocorrelated residuals
  k <- 0
  noautocorr <- 0
  while (noautocorr==0){
    cat(paste0("ADF with ",k, " lags: residuals OK? "))
    adf <- adfTest(series,lags=k,type=type)
    pvals <- Qtests(adf@test$lm$residuals,kmax,fitdf=length(adf@test$lm$coefficients))[,2]
    if (sum(pvals<0.05,na.rm=T) == 0) {
      noautocorr <- 1; cat("OK \n")}
    else cat("No \n")
    k <- k + 1
  }
```

```
  return(adf)
}

Qtests(adf@test$lm$residuals, 24, fitdf = length(adf@test$lm$coefficients))
```

```
##       lag         pval
##  [1,]   1           NA
##  [2,]   2           NA
##  [3,]   3 0.0003513985
##  [4,]   4 0.0012489485
##  [5,]   5 0.0038653705
##  [6,]   6 0.0007790498
##  [7,]   7 0.0003175156
##  [8,]   8 0.0007375004
##  [9,]   9 0.0002171155
## [10,]  10 0.0003987027
## [11,]  11 0.0001296744
## [12,]  12 0.0002134764
## [13,]  13 0.0003584635
## [14,]  14 0.0005245817
## [15,]  15 0.0008255793
## [16,]  16 0.0009178465
## [17,]  17 0.0014546720
## [18,]  18 0.0016820492
## [19,]  19 0.0008388933
## [20,]  20 0.0013058526
## [21,]  21 0.0018499740
## [22,]  22 0.0007208392
## [23,]  23 0.0011343724
## [24,]  24 0.0003799619
```

All residuals of the adf test with 0 lags are autocorrelated We need to fix a number of lags to consider for the adf test To do so we use the function adfTest_valid defined above

```
adfTest_valid(prod_train, 24, "c")
```

```
## ADF with 0 lags: residuals OK? No
## ADF with 1 lags: residuals OK? No
## ADF with 2 lags: residuals OK? No
## ADF with 3 lags: residuals OK? No
## ADF with 4 lags: residuals OK? No
## ADF with 5 lags: residuals OK? OK
```

```
##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 5
##   STATISTIC:
##     Dickey-Fuller: -2.9692
```

```
##   P VALUE:
##     0.04087
##
## Description:
##  Sat May 27 19:17:52 2023 by user: peign
```

The test indicates that we have to consider 5 lags to obtain a valid adfTest The adf test with 5 lags reject the non stationarity assumption but with a p-value close to 0.05 We will consider other tests to confirm the (or not) the stationarity of the series

```
pp.test(x=as.vector(prod_train), output=TRUE) #Phillips-Perron test
```

```
## Phillips-Perron Unit Root Test
## alternative: stationary
##
## Type 1: no drift no trend
##  lag  Z_rho p.value
##    5 -0.872   0.499
## -----
##  Type 2: with drift no trend
##  lag Z_rho p.value
##    5   -48    0.01
## -----
##  Type 3: with drift and trend
##  lag Z_rho p.value
##    5   -48    0.01
## ---------------
## Note: p-value = 0.01 means p.value <= 0.01
```

```
kpss.test(x=as.vector(prod_train)) #KPSS
```

```
## KPSS Unit Root Test
## alternative: nonstationary
##
## Type 1: no drift no trend
##  lag stat p.value
##    4 0.55     0.1
## -----
##  Type 2: with drift no trend
##  lag  stat p.value
##    4 0.379  0.0864
## -----
##  Type 1: with drift and trend
##  lag  stat p.value
##    4 0.373    0.01
## -----------
## Note: p.value = 0.01 means p.value <= 0.01
##     : p.value = 0.10 means p.value >= 0.10
```

We will thus differenciate our serie to obtain a better rejected adf test

Creating lags and first difference

```
serie$lag1 <- lag(serie$Values)
serie$dif <- serie$Values - serie$lag1

dprod <- diff(prod_train,1)
```

Again, we run a regression to test for the presence of a time trend

```
reg_diff <- lm(dif ~ Time, data=serie)
stargazer(reg_diff, type="text")
```

```
##
## ================================================
##                         Dependent variable:
##                     ---------------------------
##                                 dif
## ------------------------------------------------
## Time                             0.002
##                                 (0.005)
##
## Constant                        -0.354
##                                 (1.193)
##
## ------------------------------------------------
## Observations                      397
## R2                               0.0003
## Adjusted R2                      -0.002
## Residual Std. Error       11.821 (df = 395)
## F Statistic              0.114 (df = 1; 395)
## ================================================
## Note:                *p<0.1; **p<0.05; ***p<0.01
```

There is no time trend and significant constant so we run an adf test of type “nc”

```
dadf <- adfTest(dprod, lag=0, type="nc")
dadf
```

```
##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 0
##   STATISTIC:
##     Dickey-Fuller: -25.4826
##   P VALUE:
##     0.01
##
## Description:
##  Sat May 27 19:17:52 2023 by user: peign
```

We reject the non stationarity but we didn’t consider lags so the test isn’t valid Again, all the residuals are correlated if we use an adf test with 0 lags

```
Qtests(dadf@test$lm$residuals, 24, fitdf = length(dadf@test$lm$coefficients))
```

```
##       lag         pval
##  [1,]   1           NA
##  [2,]   2 0.0004896895
##  [3,]   3 0.0022064363
##  [4,]   4 0.0003696929
##  [5,]   5 0.0003401784
##  [6,]   6 0.0002810359
##  [7,]   7 0.0003126507
##  [8,]   8 0.0006729572
##  [9,]   9 0.0008056012
## [10,]  10 0.0011972066
## [11,]  11 0.0016432424
## [12,]  12 0.0016215196
## [13,]  13 0.0027947207
## [14,]  14 0.0046972369
## [15,]  15 0.0052219265
## [16,]  16 0.0076924575
## [17,]  17 0.0117143932
## [18,]  18 0.0095143430
## [19,]  19 0.0089672650
## [20,]  20 0.0125853709
## [21,]  21 0.0174964140
## [22,]  22 0.0066482668
## [23,]  23 0.0084729164
## [24,]  24 0.0012674457
```

We run the same test as before to determine the number of lags we have to consider

```
adfTest_valid(dprod,24,"nc")
```

```
## ADF with 0 lags: residuals OK? No
## ADF with 1 lags: residuals OK? No
## ADF with 2 lags: residuals OK? No
## ADF with 3 lags: residuals OK? No
## ADF with 4 lags: residuals OK? OK
```

```
##
## Title:
##  Augmented Dickey-Fuller Test
##
## Test Results:
##   PARAMETER:
##     Lag Order: 4
##   STATISTIC:
##     Dickey-Fuller: -12.6478
##   P VALUE:
##     0.01
##
## Description:
##  Sat May 27 19:17:53 2023 by user: peign
```

Thus, we consider 4 lags and the test is strongly rejected (pvalue < 0.01)

```
pp.test(x=as.vector(dprod), output=TRUE) #Phillips-Perron test
```

```
## Phillips-Perron Unit Root Test
## alternative: stationary
##
## Type 1: no drift no trend
##  lag Z_rho p.value
##    5  -413    0.01
## -----
##  Type 2: with drift no trend
##  lag Z_rho p.value
##    5  -413    0.01
## -----
##  Type 3: with drift and trend
##  lag Z_rho p.value
##    5  -413    0.01
## ---------------
## Note: p-value = 0.01 means p.value <= 0.01
```

```
kpss.test(x=as.vector(dprod)) #KPSS
```

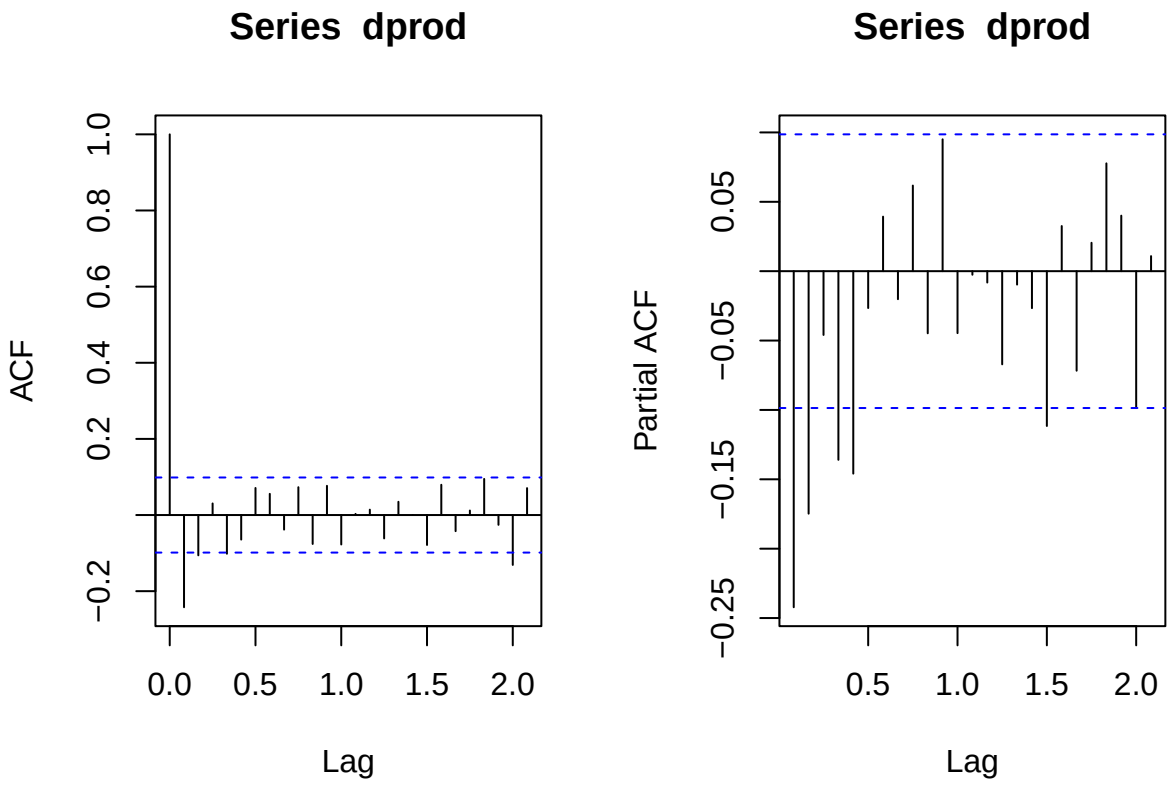
```
## KPSS Unit Root Test
## alternative: nonstationary
##
## Type 1: no drift no trend
##  lag   stat p.value
##    4 0.0931     0.1
## -----
##  Type 2: with drift no trend
##  lag   stat p.value
##    4 0.0613     0.1
## -----
##  Type 1: with drift and trend
##  lag   stat p.value
##    4 0.0403     0.1
## -----------
## Note: p.value = 0.01 means p.value <= 0.01
##     : p.value = 0.10 means p.value >= 0.10
```

Phillips-Perron and KPSS tests confirm the stationarity of the differenciated series

PART 2 : ARMA MODELS

We plot the acf and pacf to determine qmax and pmax

```
par(mfrow=c(1,2))
acf(dprod);pacf(dprod)
```



From the plot we see that

```
qmax <- 2
pmax <- 5
```

We create a loop to calculate the AIC/BIC for each possibility in the grid (qmax,pmax)=(2,5)

```
mat <- matrix(NA,nrow=pmax+1,ncol=qmax+1)
rownames(mat) <- paste0("p=",0:pmax)
colnames(mat) <- paste0("q=",0:qmax)
AICs <- mat
BICs <- mat
pqs <- expand.grid(0:pmax,0:qmax)
for (row in 1:dim(pqs)[1]){
  p <- pqs[row,1]
  q <- pqs[row,2]
  estim <- try(arima(dprod,c(p,0,q),include.mean = F))
  AICs[p+1,q+1] <- if (class(estim)=="try-error") NA else estim$aic
  BICs[p+1,q+1] <- if (class(estim)=="try-error") NA else BIC(estim)
}
```

Print the AICs

```
AICs
```

```
##           q=0       q=1       q=2
```

```
## p=0 3074.171 3041.270 3034.148
## p=1 3052.145 3033.121 3035.116
## p=2 3041.842 3035.117 3031.676
## p=3 3043.040 3034.613 3033.628
## p=4 3037.724 3033.894 3033.419
## p=5 3031.166 3032.991 3034.658
```

```
AICs==min(AICs)
```

```
##       q=0   q=1   q=2
## p=0 FALSE FALSE FALSE
## p=1 FALSE FALSE FALSE
## p=2 FALSE FALSE FALSE
## p=3 FALSE FALSE FALSE
## p=4 FALSE FALSE FALSE
## p=5  TRUE FALSE FALSE
```

The ARIMA(5,1,0) minimizes the AIC. We thus keep it.

```
arima510 <- arima(prod_train,c(5,1,0),include.mean=F)
```

Print the AICs

```
BICs
```

```
##          q=0      q=1      q=2
## p=0 3078.150 3049.228 3046.084
## p=1 3060.103 3045.058 3051.032
## p=2 3053.778 3051.032 3051.570
## p=3 3058.956 3054.507 3057.501
## p=4 3057.619 3057.768 3061.271
## p=5 3055.039 3060.843 3066.489
```

```
BICs==min(BICs)
```

```
##       q=0   q=1   q=2
## p=0 FALSE FALSE FALSE
## p=1 FALSE  TRUE FALSE
## p=2 FALSE FALSE FALSE
## p=3 FALSE FALSE FALSE
## p=4 FALSE FALSE FALSE
## p=5 FALSE FALSE FALSE
```

The ARIMA(1,1,1) minimizes the BIC. We then keep it.

```
arima111 <- arima(prod_train,c(1,1,1),include.mean=F)
```

Tables of the ARIMA

```
stargazer(arima510, type="text")
```

```
##
## ===============================================
##                         Dependent variable:
##                     ---------------------------
##                              prod_train
## -----------------------------------------------
## ar1                           -0.320***
##                                (0.050)
##
## ar2                           -0.227***
##                                (0.052)
##
## ar3                           -0.117**
##                                (0.053)
##
## ar4                           -0.181***
##                                (0.052)
##
## ar5                           -0.147***
##                                (0.050)
##
## -----------------------------------------------
## Observations                     395
## Log Likelihood                -1,509.583
## sigma2                         122.093
## Akaike Inf. Crit.             3,031.166
## ===============================================
## Note:               *p<0.1; **p<0.05; ***p<0.01
```

```
stargazer(arima111, type="text")
```

```
##
## ===============================================
##                         Dependent variable:
##                     ---------------------------
##                              prod_train
## -----------------------------------------------
## ar1                            0.388***
##                                (0.102)
##
## ma1                           -0.709***
##                                (0.078)
##
## -----------------------------------------------
## Observations                     395
## Log Likelihood                -1,513.561
## sigma2                         124.607
## Akaike Inf. Crit.             3,033.121
## ===============================================
## Note:               *p<0.1; **p<0.05; ***p<0.01
```

However, we will carry out a further study to determine the best model. We determine the significance of the coefficients of the ARs and MAs, and analyse the autocorrelation of the residuals.

To do so, we define three functions: This function returns the p-values of the estimated ARMA

```
p_value <-
  function(estim) {
    coef <- estim$coef
    se <- sqrt(diag(estim$var.coef))
    t <- coef / se
    pval <- (1 - pnorm(abs(t)))*2
    return(rbind(coef, se, pval))
  }
```

This function estimates an arima and checks the fit and validity of the model with p-value

```
model_choice <-
  function(p, q, data = dxm, k = 24) {
  estim <-
    try(arima(prod_train, c(p, 1, q), optim.control = list(maxit = 20000)))
  if (class(estim) == "try-error")
    return(c(
      "p" = p,
      "q" = q,
      "arsignif" = NA,
      "masignif" = NA,
      "resnocorr" = NA,
      "ok" = NA
    ))
  arsignif <- if (p == 0)
    NA
  else
    p_value(estim)[3, p] <= 0.05
  masignif <- if (q == 0)
    NA
  else
    p_value(estim)[3, p + q] <= 0.05
  resnocorr <-
    sum(Qtests(estim$residuals, 30, length(estim$coef) - 1)[, 2] <= 0.05, na.rm =
          T) == 0
  checks <- c(arsignif, masignif, resnocorr)
  ok <-
    as.numeric(sum(checks, na.rm = T) == (3 - sum(is.na(checks))))
  return(
    c(
      "p" = p,
      "q" = q,
      "arsignif" = arsignif,
      "masignif" = masignif,
      "resnocorr" = resnocorr,
      "ok" = ok
    )
  )
}
```

This function runs the previous one with all p<pmax & q<qmax

```
arma_model_choice <-
  function(pmax, qmax) {
  pqs <- expand.grid(0:pmax, 0:qmax)
  t(apply(matrix(1:dim(pqs)[1]), 1, function(row) {
    p <- pqs[row, 1]
    q <- pqs[row, 2]
    cat(paste0("Computing ARMA(", p, ",", q, ") \n"))
    model_choice(p, q)
  }))
}

arma_models <- arma_model_choice(pmax, qmax)
```

```
## Computing ARMA(0,0)
## Computing ARMA(1,0)
## Computing ARMA(2,0)
## Computing ARMA(3,0)
## Computing ARMA(4,0)
## Computing ARMA(5,0)
## Computing ARMA(0,1)
## Computing ARMA(1,1)
## Computing ARMA(2,1)
## Computing ARMA(3,1)
## Computing ARMA(4,1)
## Computing ARMA(5,1)
## Computing ARMA(0,2)
## Computing ARMA(1,2)
## Computing ARMA(2,2)
## Computing ARMA(3,2)
## Computing ARMA(4,2)
## Computing ARMA(5,2)
```

```
arma_models
```

```
##       p q arsignif masignif resnocorr ok
##  [1,] 0 0       NA       NA         0  0
##  [2,] 1 0        1       NA         0  0
##  [3,] 2 0        1       NA         0  0
##  [4,] 3 0        0       NA         0  0
##  [5,] 4 0        1       NA         0  0
##  [6,] 5 0        1       NA         1  1
##  [7,] 0 1       NA        1         0  0
##  [8,] 1 1        1        1         1  1
##  [9,] 2 1        0        1         1  0
## [10,] 3 1        1        1         0  0
## [11,] 4 1        1        1         1  1
## [12,] 5 1        1        0         1  0
## [13,] 0 2       NA        1         1  1
## [14,] 1 2        0        0         1  0
## [15,] 2 2        1        1         0  0
## [16,] 3 2        0        1         0  0
```

```
## [17,] 4 2        1        0         1  0
## [18,] 5 2        0        0         1  0
```

We only keep models with the column "ok" equals to 1 (significant coefficients & non-autocorrelated residuals)

```
selection <- arma_models[arma_models[, "ok"] == 1 & !is.na(arma_models[, "ok"]),]
selection
```

```
##      p q arsignif masignif resnocorr ok
## [1,] 5 0        1       NA         1  1
## [2,] 1 1        1        1         1  1
## [3,] 4 1        1        1         1  1
## [4,] 0 2       NA        1         1  1
```

Thus, we will study between the four following models: ARMA(5,0), ARMA(1,1), ARMA(4,1), and ARMA(0,2)

```
pqs <- apply(selection, 1, function(row) list("p" = as.numeric(row[1]), "q" = as.numeric(row[2])))
names(pqs) <- paste0("arma(", selection[, 1], ",", selection[, 2], ")")
models <- lapply(pqs, function(pq) arima(dprod, c(pq[["p"]], 0, pq[["q"]])))
vapply(models, FUN.VALUE = numeric(2), function(m) c("AIC" = AIC(m), "BIC" = BIC(m)))
```

```
##     arma(5,0) arma(1,1) arma(4,1) arma(0,2)
## AIC  3033.153  3035.108  3035.882  3036.136
## BIC  3061.005  3051.024  3063.734  3052.051
```

Again we choose according to the AIC / BIC We know that the ARMA(5,0) minimizes the AIC And the ARMA(1,1) minimizes the BIC Thus, we keep these two models

Test of validity of the ARMA:

```
qtest_arima510 <- Qtests(arima510$residuals,24,fitdf= length(arima510$coef)-1)
qtest_arima510
```

```
##       lag      pval
##  [1,]   1        NA
##  [2,]   2        NA
##  [3,]   3        NA
##  [4,]   4        NA
##  [5,]   5 0.9146154
##  [6,]   6 0.9833411
##  [7,]   7 0.7758029
##  [8,]   8 0.8480182
##  [9,]   9 0.7411369
## [10,]  10 0.6841753
## [11,]  11 0.5843735
## [12,]  12 0.4510504
## [13,]  13 0.4924201
## [14,]  14 0.5679063
## [15,]  15 0.4595167
## [16,]  16 0.5435495
## [17,]  17 0.6155197
```

```
## [18,]  18 0.5804533
## [19,]  19 0.6042085
## [20,]  20 0.6379465
## [21,]  21 0.6719863
## [22,]  22 0.6092539
## [23,]  23 0.6400092
## [24,]  24 0.2359899
```

```
qtest_arima111 <- Qtests(arima111$residuals,24,fitdf= length(arima111$coef)-1)
qtest_arima111
```

```
##       lag       pval
##  [1,]   1         NA
##  [2,]   2 0.86421728
##  [3,]   3 0.65425925
##  [4,]   4 0.33090435
##  [5,]   5 0.35217536
##  [6,]   6 0.24956323
##  [7,]   7 0.19814387
##  [8,]   8 0.28279931
##  [9,]   9 0.23087289
## [10,]  10 0.22994108
## [11,]  11 0.24313948
## [12,]  12 0.18421282
## [13,]  13 0.22798175
## [14,]  14 0.28481527
## [15,]  15 0.22706034
## [16,]  16 0.28586910
## [17,]  17 0.33711563
## [18,]  18 0.26306543
## [19,]  19 0.25988039
## [20,]  20 0.29400348
## [21,]  21 0.34594330
## [22,]  22 0.28460604
## [23,]  23 0.30317328
## [24,]  24 0.08508351
```

We never reject the absence of residual autocorrelation. Both models are valid.

Choice between the two models: We create a function that calculate the Adjusted R2

```
adj_r2 <- function(model){
  ss_res <- sum(model$residuals^2)
  p <- model$arma[1]
  q <- model$arma[2]
  ss_tot <- sum(dprod[-c(1:max(p,q))]^2)
  n <- model$nobs-max(p,q)
  adj_r2 <- 1-(ss_res/(n-p-q-1))/(ss_tot/(n-1))
  return(adj_r2)
}

adj_r2(arima510)
```

```
## [1] 0.1022145
```

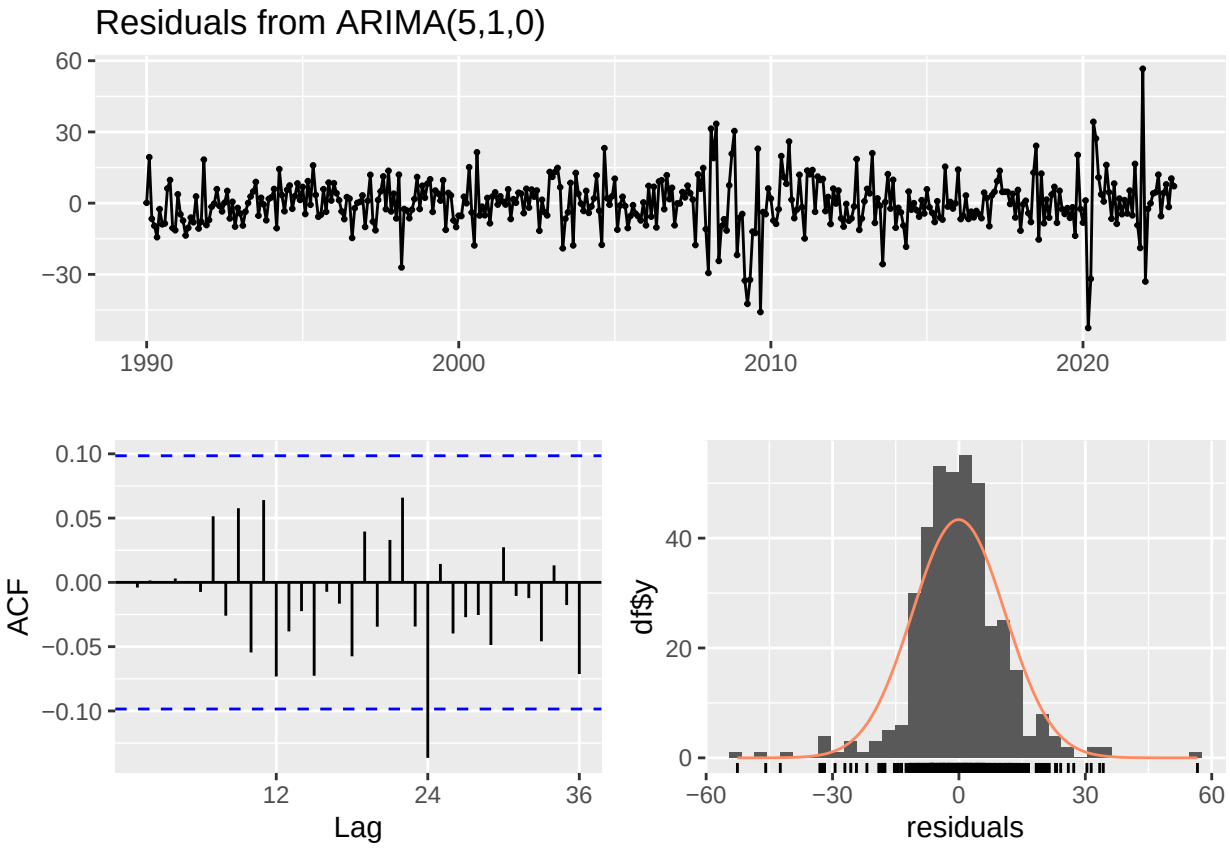
```
adj_r2(arima111)
```

```
## [1] 0.09669288
```

The ARIMA(5,1,0) has the largest adjusted R2, it gives the best prediction in the sample. We keep it as the best model.

We take a look at the residuals of our arima, thanks to the function checkresiduals

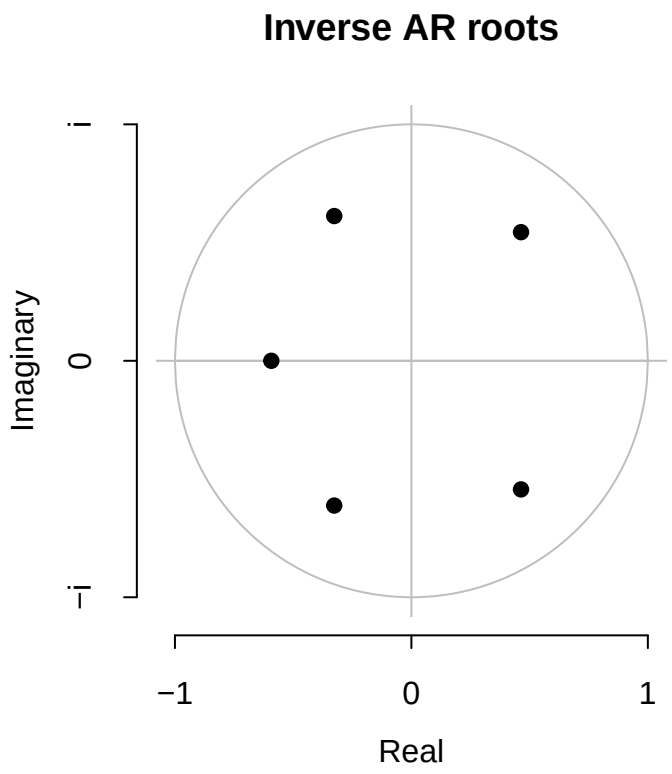
```
checkresiduals(arima510)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(5,1,0)
## Q* = 24.148, df = 19, p-value = 0.1905
##
## Model df: 5.   Total lags used: 24
```

We plot the inverse of our roots

```
model <- Arima(prod,c(5,1,0))
plot(model)
```



All the roots of our model are > 1

PLOTTING OUR ESTIMATION

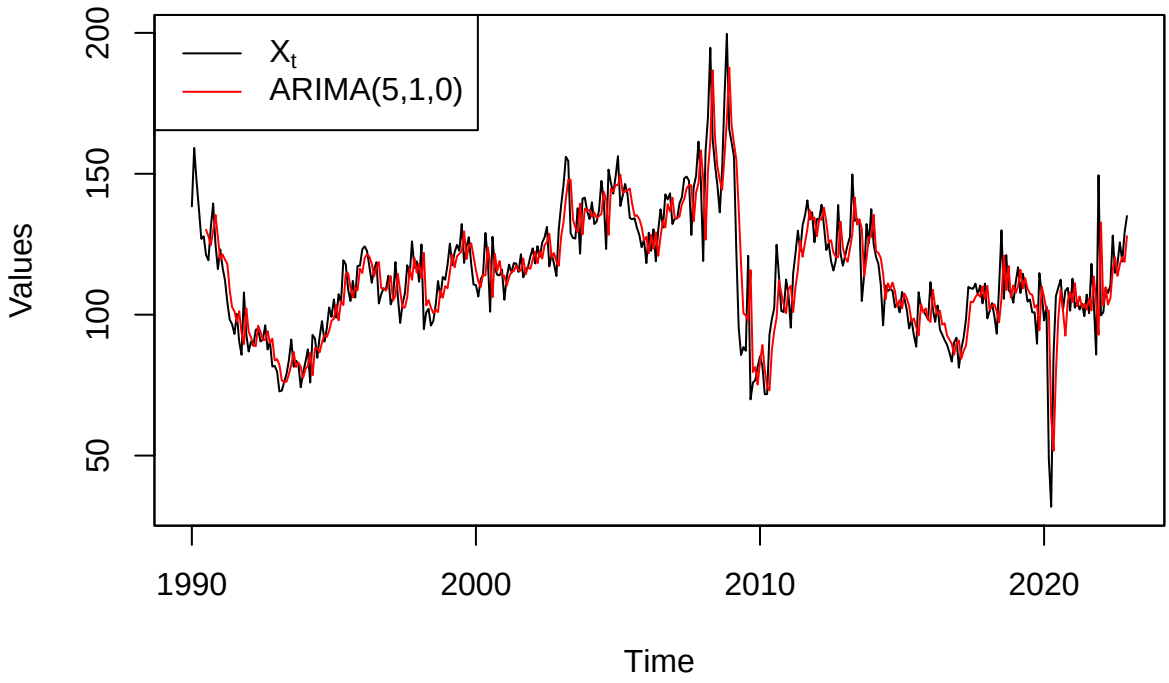
```
serie$lag1 <- lag(serie$Values)
serie$lag2 <- lag(serie$lag1)
serie$lag3 <- lag(serie$lag2)
serie$lag4 <- lag(serie$lag3)
serie$lag5 <- lag(serie$lag4)
serie$lag6 <- lag(serie$lag5)

phi1 <- arima510$coef[1]
phi2 <- arima510$coef[2]
phi3 <- arima510$coef[3]
phi4 <- arima510$coef[4]
phi5 <- arima510$coef[5]

serie$pred <- (1+phi1)*serie$lag1 + (phi2-phi1)*serie$lag2 +
  (phi3-phi2)*serie$lag3 + (phi4-phi3)*serie$lag4 +
  (phi5-phi4)*serie$lag5 - phi5*serie$lag6

predict <- zoo(serie$pred,order.by=dates)
predict <- predict[1:(length(predict)-2)]
```

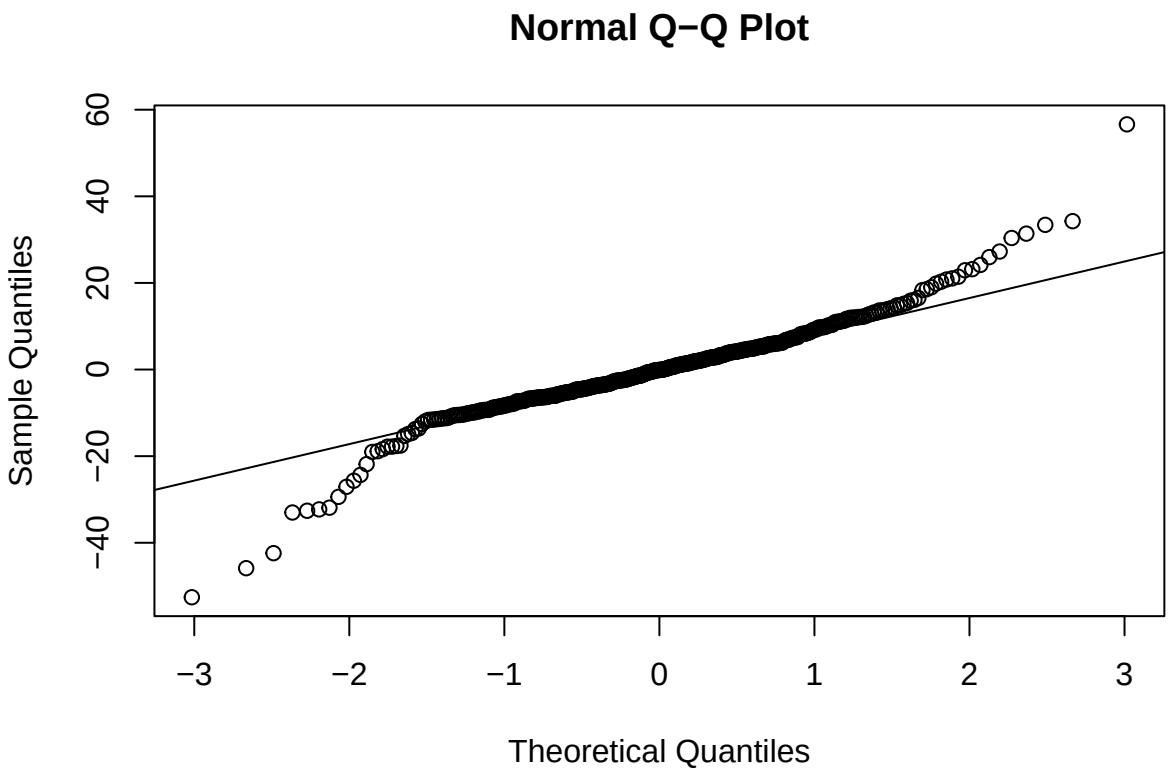
```
plot(prod_train, col = "black", xlab = "Time", ylab = "Values")
lines(predict, col = "red")
legend("topleft", legend = c(expression(X[t]), "ARIMA(5,1,0)"), col = c("black", "red"), lty = 1)
```



We take a look at the QQplot of our residuals

```
residuals <- prod - predict
residuals <- na.omit(residuals)

qqnorm(residuals)
qqline(residuals)
```



PART 3 : PREDICTION

We supposed that our residuals are gaussian We want to estimate their standard error

```
sigma <- sd(residuals)
sigma
```

```
## [1] 11.05078
```

The standard error of our residuals is sigma=11

We seek to predict the two next values of our serie thanks to the ARIMA model

```
model_pred <- predict(arima510, n.ahead=2) #prediction
pred <- zoo(model_pred$pred, order.by=as.yearmon(c(2023+0/12,2023+1/12))) #transforming our predictions
link <- rbind(prod_train[length(prod_train)],pred[1])
```

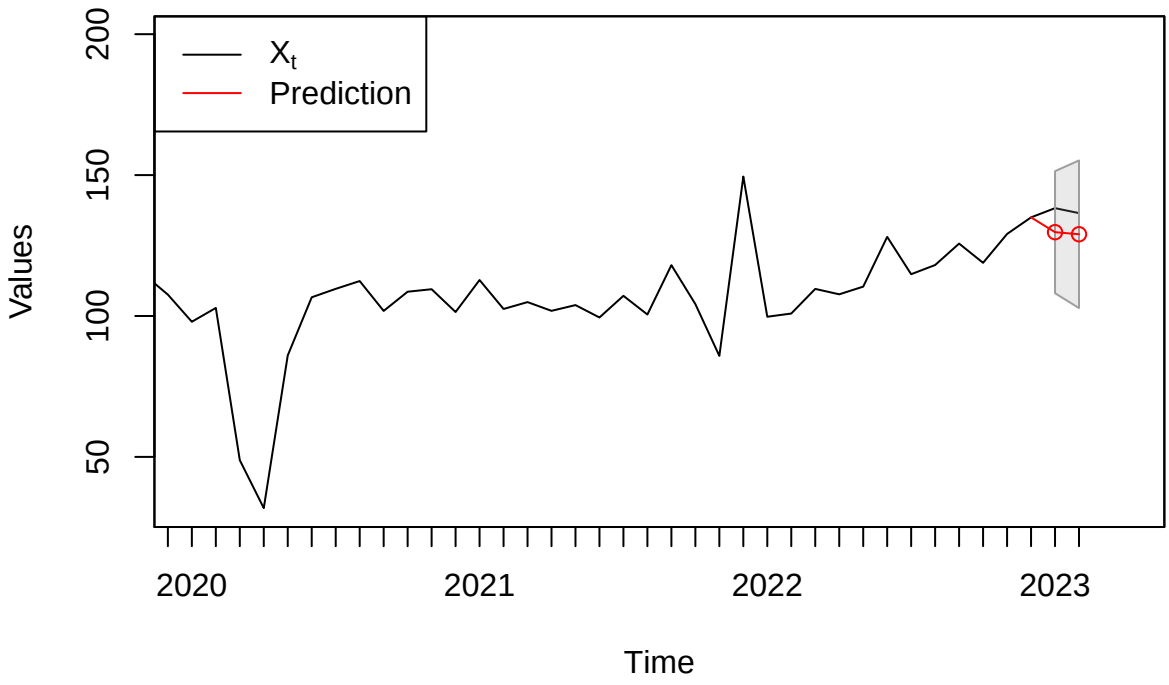
We create a function that plots the prediction, starting the serie at a given date

```
plot_pred <- function(start){
  plot(prod, col = "black", ylab = "Values", xlab="Time",
       xlim = c(start, 2023+3/12))
  U <- model_pred$pred + 1.96*model_pred$se
```

```
  L <- model_pred$pred - 1.96*model_pred$se
  Upper1 <- model_pred$pred[1] + 1.96*sigma
  Lower1 <- model_pred$pred[1] - 1.96*sigma
  Upper2 <- model_pred$pred[2] + 1.96*sigma*sqrt((1+(1+phi1)^2))
  Lower2 <- model_pred$pred[2] - 1.96*sigma*sqrt((1+(1+phi1)^2))
  Upper <- c(Upper1,Upper2)
  Upper <- zoo(Upper, order.by=as.yearmon(c(2023+0/12,2023+1/12)))
  Lower <- c(Lower1,Lower2)
  xx <- c(time(Upper), rev(time(Upper)))
  yy <- c(Lower,rev(Upper))
  polygon(xx,yy,border="8", col=gray(0.6,alpha=0.2))
  lines(pred, type = "p", col = "red")
  lines(pred, type = "l", col = "red")
  lines(link, type = "l", col = "red")
  legend("topleft", legend=c(expression(X[t]), "Prediction"), col=c("black", "red"), lty=1)
}
```

We plot our prediction, starting the serie in January 2020

```
plot_pred(2020)
```



ELLIPSE

```
Sigma <- matrix(data=c(sigma^2, sigma^2*(1+phi1), sigma^2*(1+phi1), sigma^2*(1+(1+phi1)^2)), nrow=2, by

legend_labels <- c("Border of the confidence region at level 95%", "Predicted values", "True values")
legend_colors <- c("black", "black", "red")


ell <- ellipse(x=Sigma, centre = c(model_pred$pred[1], model_pred$pred[2]), t = sqrt(qchisq(0.95, 2)))
plot(ell, type='l', xlab = expression(X[T+1]), ylab = expression(X[T+2]))
points(x=model_pred$pred[1], y=model_pred$pred[2], pch=4)
points(x=as.vector(prod)[397], y=as.vector(prod)[398], pch=4, col="red")
legend("topleft", legend=legend_labels, col=legend_colors, lty=c(1,0,0), pch=c(NA,4,4))
```