

# Projet de séries temporelles linéaires (ARMA)

Maël Duverger, Louis Geist

15 mai 2023

## Table des matières

<b>1</b>	<b>Les données</b>	<b>2</b>
1.1	Que représente la série choisie ? . . . . .	2
1.2	Estimation du degré d'intégration de la série . . . . .	2
1.2.1	Choix des spécifications du test . . . . .	2
1.2.2	Choix du nombre de lags . . . . .	2
1.2.3	Interprétation du test ADF . . . . .	3
1.3	Représentation des séries . . . . .	3
<b>2</b>	<b>Modélisation ARMA</b>	<b>3</b>
2.1	Choix des paramètres $p$ et $q$ de la modélisation ARMA . . . . .	3
2.1.1	ACF et PACF pour le choix des ordres maxima . . . . .	3
2.1.2	Ajustement du modèle . . . . .	4
2.1.3	Validation du modèle . . . . .	4
2.1.4	Critère d'information . . . . .	5
2.2	Justification de la modélisation ARIMA . . . . .	5
<b>3</b>	<b>Prévision</b>	<b>5</b>
3.1	Région de confiance de niveau $\alpha$ sur les valeurs futures $(X_{T+1}, X_{T+2})$ . . . . .	5
3.2	Hypothèses pour obtenir la région de confiance . . . . .	5
3.3	Représentation graphique de la région de confiance . . . . .	6
3.4	Question ouverte . . . . .	6
<b>A</b>	<b>Régression linéaire pour le choix des spécifications</b>	<b>7</b>
<b>B</b>	<b>Résultats AIC et BIC sur les modèles ajustés et valides</b>	<b>7</b>
<b>C</b>	<b>Code R</b>	<b>8</b>

# Introduction

Chaque sous-section du rapport répond à une des 9 questions du sujet.

## 1 Les données

### 1.1 Que représente la série choisie ?

La série étudiée est l'indice CVS-CJO (*correction des variations saisonnières - correction des effets de jours ouvrables*) production industrielle de l'industrie alimentaire. Elle a été téléchargée le 8 avril 2023 sur le site de l'[INSEE](#). L'INSEE définit l'**indice de la production industrielle** comme un "instrument statistique qui permet de suivre l'évolution mensuelle de l'activité industrielle de la France". L'indice utilisé est de base 100 en 2015.

Nous ne remarquons pas de régime particulier pendant la crise sanitaire de 2020-2022. C'est pourquoi nous avons gardé toutes les dates.

### 1.2 Estimation du degré d'intégration de la série

On suit la méthodologie de Box and Jenkins.

#### 1.2.1 Choix des spécifications du test

Dans un premier temps, il s'agit de déterminer les spécifications du test d'ADF en termes de présence d'une constante et d'une tendance non nulles.

Pour cela, nous régressons notre indice d'intérêt sur une constante et les dates. Les résultats de la régression sont disponibles en annexe A. Le coefficient associé à la date est bien positif comme laissé présager la représentation graphique de la série. Les p-valeurs ne sont pas interprétables, car notre échantillon n'est certainement pas indépendant et identiquement distribué. Autrement dit, les résidus de la régression linéaire sont possiblement encore autocorrélés. Pour cette raison, le test de Student n'est pas valide.

Cependant, les t-stats associées aux deux variables sont grandes, ce qui laisse penser que les coefficients sont significatifs. Nous allons donc réaliser le **test ADF avec constante et tendance non nulles**.

#### 1.2.2 Choix du nombre de lags

Une hypothèse du test de racine unitaire ADF est l'absence d'autocorrélation dans la régression du test ADF.

La régression du test ADF avec constante et tendance non nulle est :

$$\Delta X_t = c + bt + \pi X_{t-1} + \sum_{i=1}^{k-1} \pi_i \Delta X_{t-i} + \epsilon_t$$

L'hypothèse à vérifier est l'absence d'autocorrélation dans la série des  $(\epsilon_t)_{t \in \mathbb{Z}}$ .

En pratique, on ajoute donc des lags de  $\Delta X_t$  à la régression du test ADF tant que les résidus sont autocorrélés. On considère que les résidus ne sont pas autocorrélés, si pour tous les horizons de  $h = 1$  à  $h = 24$ , le test de Ljung-Box n'est pas rejeté au niveau de 5%.

Ma fonction "exogeneisation\_residus" retourne le nombre de lags cherchés : on trouve 8.

### 1.2.3 Interprétation du test ADF

On trouve une p-valeur de 0.27. On ne peut alors pas rejeter l'hypothèse nulle à 10%. On conclut donc sur la présence d'une racine unitaire sur la série brute.

On différencie au premier ordre la série et on réitère exactement la même procédure. On trouve cette fois une p-valeur inférieure à 0.01. On rejette donc à 1% l'hypothèse nulle, qui est l'hypothèse de racine unitaire. On conclut donc que la série différenciée est stationnaire et que la série brute est  $I(1)$ .

## 1.3 Représentation des séries

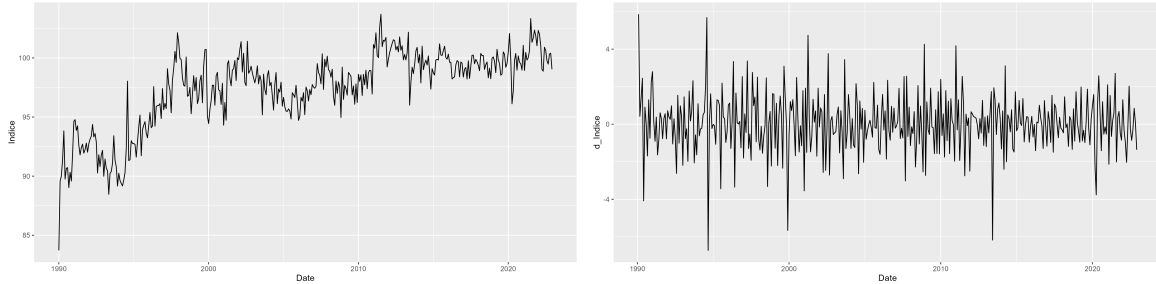


FIGURE 1 – Indice et indice différencié au premier ordre en fonction du temps

## 2 Modélisation ARMA

### 2.1 Choix des paramètres $p$ et $q$ de la modélisation ARMA

#### 2.1.1 ACF et PACF pour le choix des ordres maxima

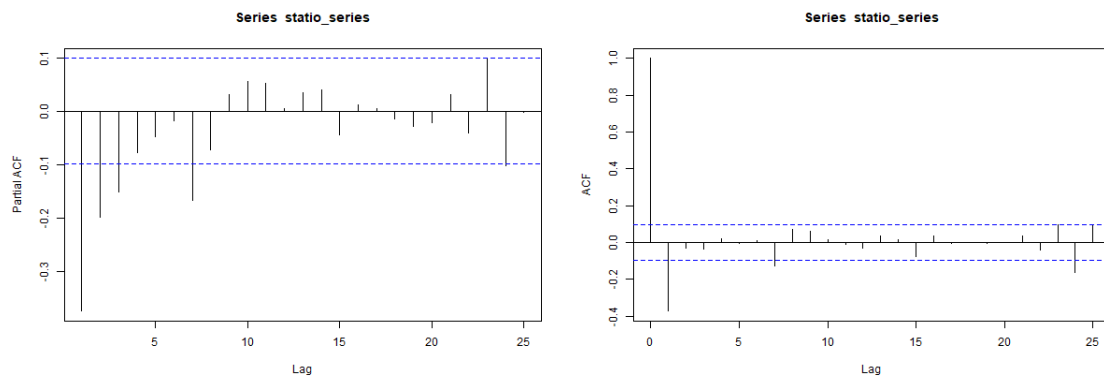


FIGURE 2 – PACF et ACF de la série corrigée

Les deux figures ACF et PACF indiquent au moins une valeur significativement non nulle pour un lag  $h > 0$ . On n'estime donc a priori pas un  $MA(q)$  ou un  $AR(p)$ , dont les ordres pourraient être directement lus sur l'ACF ou le PACF.

On va donc estimer un modèle  $ARMA(p, q)$ , avec  $p > 0$  et  $q > 0$ . Les autocorrélogrammes ACF et PACF indiquent dès lors des ordres maxima du modèle  $ARMA$  :  $p_{max}$  et  $q_{max}$  correspondent aux

	q=0	q=1
p=0	NA	TRUE
p=1	TRUE	TRUE
p=2	TRUE	FALSE
p=3	TRUE	FALSE
p=4	TRUE	FALSE
p=5	FALSE	FALSE
p=6	FALSE	FALSE
p=7	TRUE	FALSE

FIGURE 3 – Modèles  $ARMA(p, q)$  ajustés

dernières valeurs de lag qui donnent respectivement une autocorrélation partielle ou une autocorrélation non nulle. D'où :

$$\begin{cases} p_{max} = 7 \\ q_{max} = 1 \end{cases}$$

Certes, les lag 7 et 24 dépassent très légèrement le seuil de 5%, mais pour avoir des ordres raisonnables dans le modèle ARMA, on choisit de garder  $q_{max} = 1$ .

Les modèles à tester sont donc l'ensemble des  $ARMA(p, q)$ , tels que  $(p, q) \in \llbracket 0, p_{max} \rrbracket \times \llbracket 0, q_{max} \rrbracket = \llbracket 0, 7 \rrbracket \times \llbracket 0, 1 \rrbracket$ .

### 2.1.2 Ajustement du modèle

Un modèle  $ARMA(p, q)$  est dit **ajusté** si les coefficients  $\phi_p$  et  $\psi_q$  sont non nuls.

Il s'agit donc de tester cette condition pour chaque modèle de l'ensemble des modèles décrits à la fin de la partie 2.1.1. On utilise pour cela des tests de Student bilatéraux. La fonction *test\_ajustement*, que j'ai implémentée pour cela, utilise les coefficients et les écarts-types calculés dans la fonction *arima* pour calculer directement la  $p$ -valeur (rappel dans le cadre d'un test de Student :  $p_{value} = 2(1 - F(|\frac{coef}{\hat{\sigma}_{coef}}|))$  avec  $F$  la fonction de répartition de la loi normale).

Après cette étape d'ajustement, nous conservons donc 7 modèles (sur les 15/16 modèles de l'étape précédente).

### 2.1.3 Validation du modèle

Un modèle est dit **validé** si les résidus du modèle ne sont pas autocorrélés.

Cette condition est testée par le test de Ljung-Box. Pour chaque modèle, la fonction *Qtests* réalise le test de Ljung-Box pour les horizons jusqu'à 24 (les premiers horizons ne sont pas testables, car il faut que  $h > p + q$  pour que le test soit valide). On rejette le modèle si le test rejette au seuil de 5% l'absence d'autocorrélation des résidus à un certain ordre.

	q=0	q=1
p=0	NA	FALSE
p=1	FALSE	TRUE
p=2	FALSE	FALSE
p=3	FALSE	FALSE
p=4	FALSE	FALSE
p=5	FALSE	FALSE
p=6	FALSE	FALSE
p=7	TRUE	FALSE

FIGURE 4 – Modèles  $ARMA(p, q)$  ajustés et valides

La figure 4 est le résultat de ma fonction *modele\_valide*. A ce stade, on ne conserve donc que les modèles  $ARMA(1, 1)$  et  $ARMA(7, 0)$ .

### 2.1.4 Critère d'information

Pour départager les deux derniers modèles, on utilise les critères d'information AIC et BIC. Le modèle  $ARMA(1,1)$  donnent des valeurs plus faibles pour les deux critères (voir Annexe B pour les valeurs numériques). C'est donc le modèle qu'on conserve.

## 2.2 Justification de la modélisation ARIMA

Un processus  $Y$  suit un modèle  $ARIMA(p, d, q)$  si  $X_t = (1 - B)^d Y_t$  suit un modèle  $ARMA(p, q)$  causal. La partie précédente donnait  $(p, q) = (1, 1)$ . D'après la partie 1.2, nous avons  $d = 1$ .

Il s'agit donc ici de tester si le modèle  $ARMA(p, q)$  estimé en partie 2.1 est causal. Or, un modèle  $ARMA(p, q)$  est causal si et seulement si le polynôme  $\phi(B) = 1 - \sum_{i=1}^p \phi_i B^i$  n'admet pas de racine dans le disque unité.

J'ai implémenté une routine `arma_causal` qui permet de tester automatiquement cette condition. Dans notre cas, la condition se lit directement le coefficient du modèle, car  $p = 1$ . Cependant, pour des ordres  $p$  supérieurs, cette routine se révèle utile.

La racine vaut environ  $-5,571$  et donc le modèle est bien causal.

## 3 Préviation

On note  $T = 395$  la longueur de la série corrigée  $X_t$ .

On suppose que les résidus de la série sont gaussiens.

### 3.1 Région de confiance de niveau $\alpha$ sur les valeurs futures $(X_{T+1}, X_{T+2})$

On note  $\phi$  et  $\psi$  les coefficients de notre  $ARMA(1,1)$  tel que  $X_t = \phi X_{t-1} + \epsilon_t - \psi \epsilon_{t-1}$ . Notre  $ARMA$  est canonique, car les racines des deux polynômes sont distinctes et hors du disque unité. Donc  $\epsilon_t$  est l'innovation linéaire de  $X_t$  et on obtient directement :

$$- \hat{X}_{t+1|t} = \phi X_t - \psi \epsilon_t$$

$$- \hat{X}_{t+2|t} = \mathbb{E}[\phi X_{t+1|t} - \psi \epsilon_{t+1} + \epsilon_{t+2} | X_t, \dots, X_0] = \phi \hat{X}_{t+1|t} = \phi^2 X_t - \phi \psi \epsilon_t$$

$$\text{On note } X = \begin{pmatrix} X_{T+1} \\ X_{T+2} \end{pmatrix} \text{ et } \hat{X} = \begin{pmatrix} \hat{X}_{t+1|t} \\ \hat{X}_{t+2|t} \end{pmatrix}.$$

Les résidus, qui sont des bruits blancs, sont supposés gaussiens. On note  $\sigma^2$  leur variance. Donc :

$$X - \hat{X} = \begin{pmatrix} \epsilon_{t+1} \\ (\phi - \psi)\epsilon_{t+1} + \epsilon_{t+2} \end{pmatrix} \sim \mathcal{N}(0_2, \sigma^2 \begin{pmatrix} 1 & \phi - \psi \\ \phi - \psi & 1 + (\phi - \psi)^2 \end{pmatrix}).$$

On note  $\Sigma$  la matrice de variance-covariance.  $\sigma^2$  est strictement positif : si elle était nulle, alors  $X_t$  serait déterministe. La trace et le déterminant sont donc clairement strictement positifs. Donc la matrice est définie positive. On peut donc définir sa matrice inverse et la racine carrée de sa matrice inverse.

On obtient  $\Sigma^{-1/2}(X - \hat{X}) \sim \mathcal{N}(0, I_2)$ . Donc  $(X - \hat{X})^T \Sigma^{-1}(X - \hat{X}) \sim \chi^2(2)$ . En notant  $q_\alpha$  le quantile de niveau  $\alpha$  de la loi de  $\chi^2$  avec deux degrés de liberté, on obtient la région de confiance suivante :

$$\mathbb{P}((X - \hat{X})^T \Sigma^{-1}(X - \hat{X}) \in [0, q_{1-\alpha}]) = 1 - \alpha$$

car les  $\chi^2$  sont à support dans  $\mathbb{R}_+$ .

### 3.2 Hypothèses pour obtenir la région de confiance

Les hypothèses suivantes ont été nécessaires pour établir la région de confiance établie dans la partie 3.1 :

- Les résidus sont gaussiens (de variance non nulle),

- Le modèle de la série est connu,
- Les coefficients du modèle sont connus.

### 3.3 Représentation graphique de la région de confiance

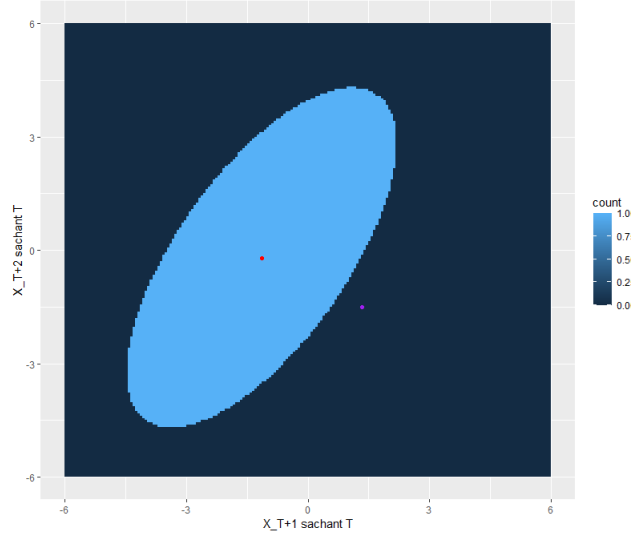


FIGURE 5 – Région de confiance à 95% du couple  $(X_{T+1}, X_{T+2})$  en bleu ciel. Le point rouge est le point de prédiction linéaire optimale, c'est-à-dire  $(\hat{X}_{T+1|T}, \hat{X}_{T+2|T})$ . Le point violet est le point  $(X_{T+1}, X_{T+2})$ .

On constate que la région de confiance forme une ellipse, qui est allongée selon une pente d'environ  $+1$ . C'était prévisible, car la covariance entre les deux composantes du vecteur  $X - \hat{X}$  vaut  $\phi - \psi$  sont tels que  $\hat{\phi} - \hat{\psi} = 0.91$ .

On constate également que la région de confiance est centrée autour de la prévision linéaire. Cela provient de notre choix de quantiles pour la région de confiance.

Enfin, la valeur réelle est en dehors de la région de confiance. Cela arrive une fois sur vingt prédictions.

### 3.4 Question ouverte

Les séries  $X_t$  et  $Y_t$  sont stationnaires par hypothèse. La causalité instantanée de  $Y_t$  vers  $X_t$  est donc bien définie. Si  $Y_t$  cause instantanément  $X_t$ , alors la connaissance de  $Y_{T+1}$  permet d'améliorer la prévision de  $X_{T+1}$ . En effet, dans ce cas là on aurait par définition :  $\mathbb{E}[X_{T+1} | (X_u, Y_u)_{u \leq T} \cup Y_{T+1}] \neq \mathbb{E}[X_{T+1} | (X_u, Y_u)_{u \leq T}]$ . Pour tester l'hypothèse " $Y_t$  cause instantanément  $X_t$ ", on peut utiliser le test de Wald.

Pour utiliser cette information, on peut régresser  $X_t$  sur les données passées de  $X$  et  $Y$  ainsi que sur  $Y_{T+1}$ .

## A Régression linéaire pour le choix des spécifications

```
Call:
lm(formula = source2$Index ~ source2$Date)

Residuals:
    Min       1Q   Median       3Q      Max
-9.4435 -1.3252 -0.1192  1.2947  6.9223

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -420.6686    22.4637  -18.73  <2e-16 ***
source2$Date   0.2582     0.0112   23.06  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.122 on 394 degrees of freedom
Multiple R-squared:  0.5745,    Adjusted R-squared:  0.5734
F-statistic: 531.9 on 1 and 394 DF,  p-value: < 2.2e-16
```

FIGURE 6 – Régression linéaire de l'indice  $Y_t$  sur le temps

On opte pour une spécification "ct" du test adf, c'est-à-dire avec constante et tendance non nulles.

```
Call:
lm(formula = source3$d_Index ~ source3$Date)

Residuals:
    Min       1Q   Median       3Q      Max
-6.8248 -0.8855 -0.0141  0.9019  5.7296

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  11.239779    16.594688   0.677   0.499
source3$Date -0.005582     0.008270  -0.675   0.500

Residual standard error: 1.562 on 393 degrees of freedom
Multiple R-squared:  0.001158,    Adjusted R-squared:  -0.001384
F-statistic: 0.4556 on 1 and 393 DF,  p-value: 0.5001
```

FIGURE 7 – Régression linéaire de l'indice différencié  $X_t = (1 - B)Y_t$  sur le temps

On opte pour une spécification "nc" du test adf, c'est-à-dire avec constante et tendance nulles.

## B Résultats AIC et BIC sur les modèles ajustés et valides

```
> print(paste0("AIC pour ARIMA(7,0,0) : ", AIC(arima700)))
[1] "AIC pour ARIMA(7,0,0) : 1377.55791277653"
> print(paste0("AIC pour ARIMA(1,0,1) : ", AIC(arima101)))
[1] "AIC pour ARIMA(1,0,1) : 1376.32154872672"
> print(paste0("BIC pour ARIMA(7,0,0) : ", BIC(arima700)))
[1] "BIC pour ARIMA(7,0,0) : 1409.38899889573"
> print(paste0("BIC pour ARIMA(1,0,1) : ", BIC(arima101)))
[1] "BIC pour ARIMA(1,0,1) : 1388.25820602142"
```

FIGURE 8 – Valeurs des AIC et BIC des modèles  $ARMA(1,1)$  et  $ARMA(7,0)$

## C Code R

```
!=, ,, *, &, %/ %, % * %, %%, < -, << -, /,,
1 # Projet – Linear time series
2
3 #----- Partie I : Les données -----
4
5 rm(list = ls())
6
7 library("ggplot2")
8 library("zoo")
9 library("dplyr")
10 library("fUnitRoots")
11 # library("tseries")
12 library("polynom") #pour la question 5 (justification ARIMA)
13
14
15 # Préparation des données
16 source = read.csv(file = "valeurs_mensuelles_industrie_alimentaire.csv",
17   sep=";", dec = ".")
18 source2 = source[4:length(source$Libellé),1:2]
19 colnames(source2) = c("Date","Indice")
20 Date_num = seq(1990,1990+397/12,1/12)
21 source2$Indice = as.numeric(source2$Indice)
22 source2 = source2[–1] # on supprime la variable "date" qui n'est pas numé-
   rique
23 source2 = cbind(Date_num,source2)
24 colnames(source2)=c("Date","Indice")
25
26 # On met deux valeurs de côté pour la prévision à la fin
27 source_end = slice_tail(source2, n=2)
28 source2 = slice(source2, 1:(length(source2$Indice)–2))
29
30 #### 2. Transformation de la série pour la rendre stationnaire
31
32 # Etape 1 : choix de la spécification du test ADF
33 summary(lm(source2$Indice ~ source2$Date))
34
35
36 # Etape 2 : choix du nombre de lags
37 Qtests <- function(series, k, fitdf=0) { #réalise le test de Ljung–Box
   pour les horizons jusqu'à 24 de la série "series" mis en argument
38   pvals <- apply(matrix(1:k), 1, FUN=function(l) {
39     pval <- if (l<=fitdf) NA else Box.test(series, lag=l, type="Ljung–Box
       ", fitdf=fitdf)$p.value
40     return(c("lag"=l,"pval"=pval))
41   })
42   return(t(pvals))
43 }
44
45 exogeneisation_residus = function (series,specification) {
```



```

46 lag_max = 36
47 for(lag in 0:lag_max){
48   my_adf = fUnitRoots::adfTest(series, lags = lag, type = specification
49   )
49   tab_p_val_autocorr = Qtests(my_adf@test$lm$residuals, 24, fitdf =
      length(my_adf@test$lm$coefficients))
50
51   non_rejet = c((tab_p_val_autocorr[,2] > 0.05) | is.na(
      tab_p_val_autocorr[,2]))
52   if(sum(non_rejet)==24){ #test si tous les tests de Ljung-Box sont
      soit non rejetés, soit n'ont pas été réalisés car le lag était
      trop faible pour interprétation
53     return(lag)
54   }
55 }
56 return(paste0("Pas d'absence d'autocorrélation trouvé jusqu'à l'ajout
      du lag = ",lag_max))
57 }
58
59 lag_pour_adf_valide = exogeneisation_residus(source2$Indice,"ct")
60
61
62 # Etape 3 : réalisation du test ADF
63 test_adf = fUnitRoots::adfTest(source2$Indice, lags = lag_pour_adf_valide
64   , type = "ct")
65 print(paste0("La pvalue du test ADF est : ", round(test_adf@test$p.value
66   , digits = 2)))
67
68 # Ccl: la série source2$indice admet une racine unitaire
69
70 #### On réitère les 3 étapes pour la série différenciée au premier ordre
71 d_Indice = diff(source2$Indice)
72 Date = source2$Date[2:(length(d_Indice)+1)]
73 source3 = as.data.frame(cbind(Date,d_Indice))
74 val_a_prevoir = as.data.frame(cbind(source_end$Date,c(source_end$Indice
75   [2]-source_end$Indice[1],source_end$Indice[1]-source2$Indice[396])))
76
77 # Etape 1bis :
78 summary(lm(source3$d_Indice ~ source3$Date))
79
80 # Etape 2bis :
81 lag_pour_adf_valide_d = exogeneisation_residus(source3$d_Indice,"nc")
82
83 # Etape 3bis :
84 test_adf_d = fUnitRoots::adfTest(source3$d_Indice, lags =
85   lag_pour_adf_valide, type = "nc")
86
87 print(paste0("La pvalue du test ADF est : ", round(test_adf_d@test$p.
88   value, digits = 2)))

```

```

86 # Ccl : la série source3$d_Indice n'admet pas de racine unitaire et
    donc est stationnaire
87
88
89
90 ### 3. Représentation graphique
91 p = ggplot(data=source2) + geom_line(aes(x=Date,y=Indice))
92 p
93 ggsave("Serie_brute.png",path="./Images_pour_rapport",width = 10, height
    = 5)
94
95
96 p_diff = ggplot(data=source3) + geom_line(aes(x=Date,y=d_Indice))
97 p_diff
98 ggsave("Serie_differenciee.png",path="./Images_pour_rapport",width = 10,
    height = 5)
99
100
101 #----- Partie II : Modèles ARMA -----
102 statio_series = source3$d_Indice
103 # 4. Choix du modèle ARMA
104
105 acf(statio_series)
106 dev.print(device = png, file = "./Images_pour_rapport/acf.png", width =
    600)
107 # ACF => q = 1
108
109 pacf(statio_series)
110 dev.print(device = png, file = "./Images_pour_rapport/pacf.png", width =
    600)
111 # PACF => p = 7
112
113 q_max = 1
114 p_max = 7
115
116
117 #test_ajustement : #renvoie un tableau de Booléens, qui indique quel modè
    le est correctement ajusté,
118 #c'est-à-dire les modèles tels que les coefficients d'ordre p et q soient
    tous les deux significatifs au seuil de 5% (test de Student)
119
120 test_ajustement = function(statio_series, p_max, q_max){
121   res = matrix(NA, nrow = (p_max+1), ncol = (q_max+1))
122   rownames(res) <- paste0("p=",0:p_max)
123   colnames(res) <- paste0("q=",0:q_max)
124
125   for(p in 0:(p_max)){
126     for(q in 0:(q_max)){
127       model = arima(x = statio_series, order = c(p,0,q), include.mean =
          FALSE)
128       p_vals = 2*(1-pnorm(abs(model$coef)/diag(model[["var.coef"]])
          *(1/2))) #test de Student

```

```

129
130     if(p==0 | q==0){
131         if(p!=0 | q!=0){ # on laisse p=0, q=0 en NA..
132             if(p==0){ # si p ou q est nul, alors on ne peut évidemment que
133                 tester la significativité de l'autre coefficient
134                 if(p_vals[q]<0.05){
135                     res[1,q+1]=TRUE
136                 }
137                 else{
138                     res[1,q+1]=FALSE
139                 }
140             }
141             if(q==0){
142                 if(p_vals[p]<0.05){
143                     res[p+1,1]=TRUE
144                 }
145                 else{
146                     res[p+1,1]=FALSE
147                 }
148             }
149         }
150     }
151     else{
152         if(p_vals[p]<0.05 & p_vals[p+q]<0.05){
153             res[p+1,q+1] = TRUE
154         }
155         else {res[p+1,q+1] = FALSE}
156     }
157 }
158 return(res)
159 }
160
161 tab_ajustement = test_ajustement(statio_series, p_max, q_max)
162 print("Liste des modèles correctement ajustés : ")
163 print(tab_ajustement)
164
165 modele_valide = function (series, tab_ajustement, p_max,q_max){
166     validation_matrix = tab_ajustement
167
168
169     for(p in 0:(p_max)){
170         for(q in 0:(q_max)){
171             if (is.na(tab_ajustement[p+1,q+1])==FALSE){
172
173
174                 if (tab_ajustement[p+1,q+1]==TRUE) {# on regarde la validité des modèles
175                     qui sont bien ajustés
176                     model = arima(x = statio_series, order = c(p,0,q), include.mean =
177                         FALSE)
178                     tab_p_val_autocorr = Qtests(model$residuals, 24, fitdf = p+q)

```

```

178     non_rejet = c((tab_p_val_autocorr[,2] > 0.05) | is.na(
179         tab_p_val_autocorr[,2]))
180     if(sum(non_rejet)!=24){ #test si tous les tests de Ljung-Box sont
181         soit non rejetés, soit n'ont pas été réalisés car le lag é
182         tait trop faible pour interprétation
183         validation_matrix[p+1,q+1] = FALSE
184     }
185     # sinon, on laisse TRUE dans le tableau validation_matrix
186 }
187 }
188 return(validation_matrix)
189 }
190
191 tab_valide = modele_valide(statio_series, tab_ajustement, p_max, q_max)
192 print("Parmi les modèles ajustés, voici le tableau des modèles valides (c
193     'est-à-dire tels que leurs résidus ne sont pas autocorrélés)")
194 print(tab_valide)
195 # On garde seulement les modèles ARIMA(7,0,0) et ARIMA(1,0,1) pour la sé
196     rie différenciée à l'ordre 1
197
198 arima700 = arima(x = statio_series, order = c(7,0,0), include.mean =
199     FALSE)
200 arima101 = arima(x = statio_series, order = c(1,0,1), include.mean =
201     FALSE)
202
203 print(paste0("AIC pour ARIMA(7,0,0) : ", AIC(arima700)))
204 print(paste0("AIC pour ARIMA(1,0,1) : ", AIC(arima101)))
205
206 print(paste0("BIC pour ARIMA(7,0,0) : ", BIC(arima700)))
207 print(paste0("BIC pour ARIMA(1,0,1) : ", BIC(arima101)))
208
209 # On conserve donc le modèle ARIMA(1,0,1), car il minimise les deux critè
210     res d'information (AIC et BIC).
211
212 # 5. Modélisation ARIMA
213
214 # Il s'agit de montrer que le modèle ARMA(1,1) qu'on a pour la série diff
215     érenciée est bien causal.
216
217 # Or un ARMA est causal ssi pas de racine dans le disque unité du polynô
218     me phi
219
220 arma_causal = function(model){
221     if(model$ar[1]==0){# gère le cas trivial d'un modèle MA
222         return(TRUE)
223     }
224     else{
225         phi = polynomial(c(1,-model$coef[1:(model$ar[1])]))

```

```

219     racines = polyroot(phi) #les coefficients polynomiaux sont donnés
        dans l'ordre CROISSANT
220
221     for(i in 1:length(racines)){
222         if (abs(racines[i])<=1) {
223             return(FALSE)
224         }
225     }
226     return(TRUE) #si toutes les racines sont de module strictement supé
        rieur à 1, alors ARMA causal
227 }
228 }
229
230 arma_causal(arima101) # renvoie TRUE
231
232 # Remarque : dans ce cas précis, où le polynôme est de degré 1, il était
        clair que la racine est en dehors
233 # du disque unité, mais l'écriture de cette fonction avait pour but d'é
        crire une routine généralisable
234
235 # Le modèle ARMA pour la série différenciée est causal.
236 # Donc, par définition d'un ARIMA, la série non transformée suit un modè
        le ARIMA(1,1,1).
237
238
239 #----- Partie III : Prévvision -----
240 # 8. Equation de la région de confiance de niveau alpha
241 phi = arima101$coef[1]
242 psi = arima101$coef[2]
243 sigma = mean(arima101$residuals**2)-mean(arima101$residuals)**2 #variance
        du bruit blanc
244
245 mat_sigma = matrix( data = c(1,phi-psi,phi-psi,1+(phi-psi)**2) *sigma,
        nrow=2,ncol=2)
246 sigma_inverse = solve(mat_sigma) #calcul de la matrice inverse de Sigma
247
248 X_T = c(slice_tail(source3,n=1)$d_Indice)
249 epsilon_T = arima101$residuals[395]
250
251 X_hat = c(phi*X_T - psi*epsilon_T,phi**2 *X_T -psi*phi*epsilon_T) #
        vecteur ( $\hat{X}_{T+1|T}$ ,  $\hat{X}_{T+2|T}$ ), cad prévision en T+1 et T+2
        sachant T
252
253 alpha = 0.05
254
255 #création de la grille d'affichage
256 library(dplyr)
257 N =200 # découpage
258 abs = seq(-6,6,length.out = N)
259 ord = seq(-6,6,length.out = N)
260
261 grille = tidyr::expand_grid(x=abs,y=ord)

```

```

262
263
264 ### Région confiance avec une loi khi-deux(2)
265 dans_region_confiance_khi_deux = function(X){
266   Z = t(c(X[1],X[2]) - X_hat) %*% sigma_inverse %*% (c(X[1],X[2]) - X_hat)
267   if(Z < qchisq(0.95, df = 2)){
268     return(TRUE)
269   }
270   else return(FALSE)
271 }
272
273 val_B = double(length(grille$x))
274
275 # grille_avec_region = dplyr::mutate(grille, val = dans_region_confiance(
276   c(grille$x, grille$y))) # ne fonctionne pas
277 # Donc j'utilise une boucle classique :
278 for(index in 1:length(grille$x)){
279   mon_x = grille$x[index]
280   mon_y = grille$y[index]
281   val_B[index] = dans_region_confiance_khi_deux(c(mon_x, mon_y))
282 }
283 grille_avec_region_B = as.data.frame(cbind(grille, val_B))
284
285 p = ggplot(data = grille_avec_region_B, aes(x = x, y = y, weight = val_B)) +
286   geom_bin2d(bins = N) + geom_point(aes(x = val_a_prevoir$V2[1], y =
287     val_a_prevoir$V2[2], colour = "purple")) + geom_point(aes(x = X_hat[1], y =
288     X_hat[2], colour = "red")) + xlab("X_T+1 sachant T") + ylab("X_T+2 sachant T")
289
290 p
291 ggsave(filename = "../Images_pour_rapport/region_confiance.png", width = 5.5,
292   height = 4.5) #Sauvegarde manuel avec width = 800, pcq la ggsave fait
293   des traits moches...

```