ENSAE Paris

Projet de Série Temporelle Linéaire

Modélisation ARIMA d'une série temporelle

Coppa Maxime

Année 2023-2024

Table des matières

1 Les données

1.1 Que représente la série choisie ?

Dans ce projet, nous nous intéressons à la modélisation et à la prévision d'un indice de production industrielle. Notre indice représente la fabrication de tapis et de moquettes (NAF rev. 2, niveau classe, article 13.93). Cet indice CVS-CJO, corrigé des variations saisonnières et des jours ouvrables, a été mesuré par l'INSEE entre janvier 2013 et mars 2024. Cet indice est représenté en base 100 de l'année 2021, ce qui signifie que les indices ont une moyenne de 100 en 2021. Nous noterons cette série X_t dans ce qui suit. Pour cette étude, nous utiliserons la méthodologie Box-Jenkins qui consiste à identifier et estimer un modèle ARIMA. Elle comprend plusieurs étapes : stationnarisation, identification du modèle, estimation des paramètres, validation du modèle, sélection et enfin prévision ou interprétation. Nous représentons la série brute X_t sur le graphique ci-dessous.

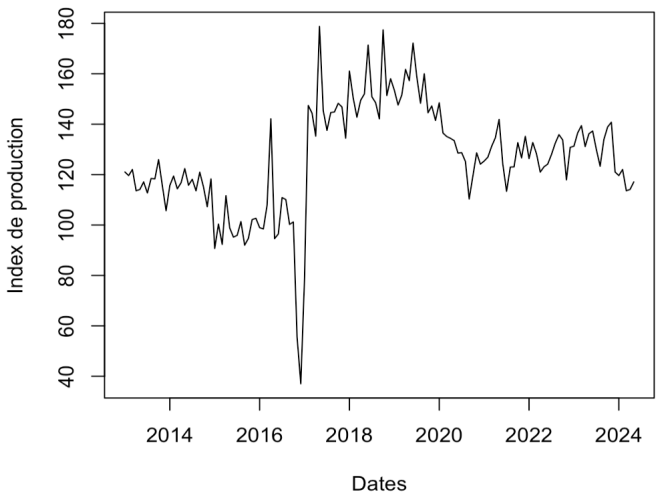


FIGURE 1 – Indice de production industrielle de fabrication de tapis et de moquettes en France

Avant de réaliser les tests de racine unitaire (pour vérifier si notre série temporelle est stationnaire), nous devons vérifier s'il y a une interception et/ou une tendance linéaire non nulle. Régressons notre indice sur ses dates pour vérifier : le coefficient associé à la tendance linéaire (dates) et le coefficient associé à la constante (intercep) sont tous deux non nuls et peut-être statistiquement significatifs (ce qui ne peut être confirmé car le test n'est pas valide lorsqu'il y a peut-être des résidus autocorrélés). Nous devons étudier le cas des tests de racine unitaire avec interception et éventuellement des tendances non nulles.

Nous avons effectué deux tests pour établir la stationnarité de la série différenciée Y_t : un test de Dickey-Fuller augmenté et un test KPSS (Kwiatkowski-Phillips-Schmidt-Shin).

Le test de Dickey-Fuller augmenté (ADF) dans le cas d'interception et de tendance consiste en la régression suivante, avec une variable X donnée :

$$\Delta X_t = c + bt + \beta X_{t-1} + \sum_{l=1}^{k} \phi_l \Delta X_{t-l} + \epsilon_t, \quad k > 0$$

où $\beta + 1$ est l'autocorrélation d'ordre 1 de X et k est le nombre de décalages nécessaires pour rendre nos résidus non autocorrélés. L'hypothèse nulle de racine unitaire $H_0 : \beta = 0$ est testée par la statistique de test $\hat{a}/\hat{\sigma}(\hat{a})$, qui suit une loi Dickey-Fuller en fonction du nombre d'observations et du cas de test que nous étudions.

Avant d'interpréter le test, nous devions vérifier que les résidus du modèle n'étaient pas autocorrélés, sinon le test ne serait pas valide. Étant donné que nous avons une série mensuelle, nous avons testé l'autocorrélation des résidus jusqu'à l'ordre 24 (2 ans), sans oublier de corriger les degrés de liberté du nombre de régresseurs. Comme nous avons rejeté, dans notre cas, l'absence d'autocorrélation des résidus au moins une fois (invalidant ainsi le test ADF sans décalages), nous avons dû ajouter des décalages de

ΔX_t jusqu'à ce que les résidus ne soient plus autocorrélés. Nous avons dû considérer 23 décalages dans le test ADF pour éliminer l'autocorrélation des résidus.

```
> kpss.test(index)

        KPSS Test for Level Stationarity

data:  index
KPSS Level = 0.71631, Truncation lag parameter = 4, p-value = 0.01206
```

```
> adf

Title:
 Augmented Dickey-Fuller Test

Test Results:
  PARAMETER:
    Lag Order: 23
  STATISTIC:
    Dickey-Fuller: -0.2253
  P VALUE:
    0.5421
```

FIGURE 2 – Résultats des tests KPSS et ADF de stationnarité

La racine unitaire n'est pas rejetée au niveau de 95% pour la série en niveaux avec le test ADF puisque la p-valeur est égale à $0,5421 > 0,05$.

Le test KPSS cherche à tester si le processus X_t est stationnaire. En effet, nous voulons, formellement, tester l'hypothèse $H_0 : \sigma_u^2 = 0$ lorsque $X_t = \xi_t + r_t + \epsilon_t$ où $r_t = r_{t-1} + u_t$, ξ_t désigne la tendance déterministe, (ϵ_t) est stationnaire, r_0 est pris constant et $u_t \sim iid(0, \sigma_u^2)$. La p-valeur obtenue en effectuant ce test est inférieure à 0,01. Comme $0,012 < 0,05$, nous rejetons l'hypothèse de stationnarité (H0) du processus X_t aux seuils de 5%. La série est donc au moins I(1).

Nous rejetons donc la stationnarité de la série, grâce à ces 2 tests.

1.2 Transformation de la série temporelle (stationnarisation)

Afin de rendre notre série stationnaire, nous l'avons transformée en la différenciant à l'ordre 1 : $Y_t := X_t - X_{t-1}$. Nous avons ainsi obtenu la représentation suivante :

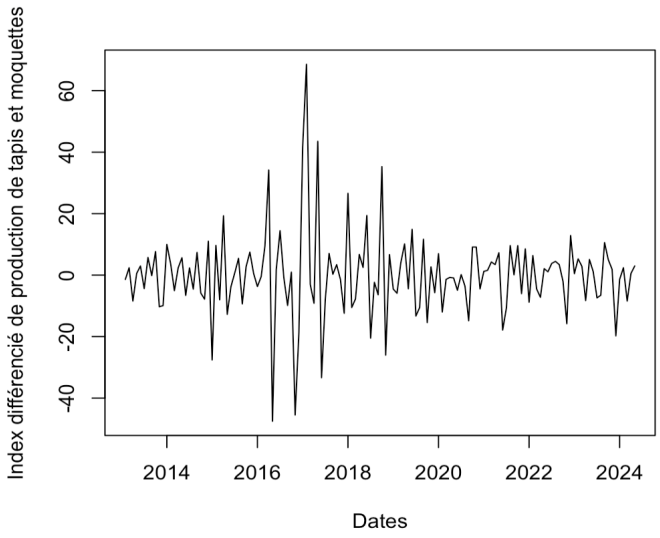


FIGURE 3 – Indice différencié (1er ordre) de la production de tapis et moquettes

Il semble que la série différenciée d'ordre 1 évolue autour de la même moyenne proche de 0. Nous avons vérifié la stationnarité en utilisant la même méthodologie que nous venons d'utiliser pour valider la stationnarité de la série (régression sur les dates, calcul du nombre de décalages, tests ADF et KPSS). La p-valeur obtenue du test ADF est de 0,01, donc le test rejette la racine unitaire. Par conséquent, nous pouvons dire que la série différenciée à l'ordre 1 est stationnaire. (De même, le test KPSS a fourni une p-valeur supérieure à 0,1 et donc 0,05, nous acceptons donc l'hypothèse de stationnarité de la série

différenciée). Ainsi, nous n'avons pas eu besoin de calculer une différenciation supplémentaire. De plus, comme la saisonnalité a déjà été corrigée dans les données, nous n'observons aucune saisonnalité.

2 Modèles ARMA

2.1 Choix d'un modèle ARMA(p,q) pour la série corrigée, estimation des paramètres du modèle et validité du modèle

Nous avons ensuite cherché le meilleur modèle ARMA(p,q) pour notre série différenciée à l'ordre 1 en utilisant la méthodologie Box-Jenkins. Pour ce faire, nous avons d'abord examiné la fonction d'autocorrélation (FAC) de la série différenciée pour trouver le q maximum et la fonction d'autocorrélation partielle (FACP) pour déterminer l'ordre maximum p.

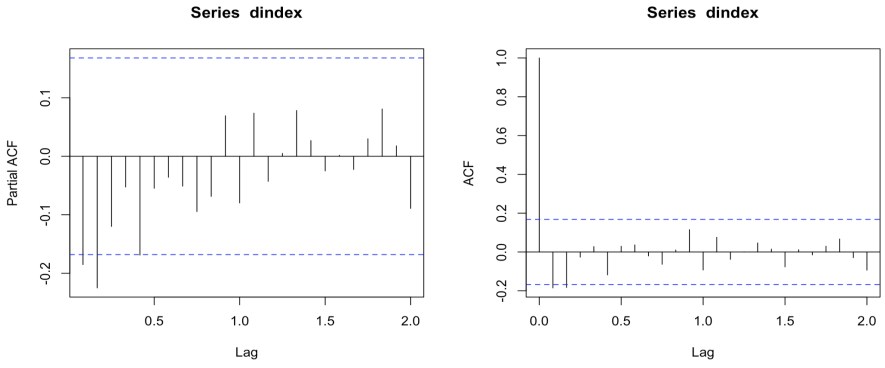


FIGURE 4 – Fonction d'autocorrélation (ACF) et fonction d'autocorrélation partielle (PACF) de la série différenciée

La figure 4 montre que la fonction d'autocorélation (ACF ou FAC) ne présente plus de pic significatif, c'est-à-dire dépassant la bande pointillée, au-delà de 2 décalages, contre 4 décalages pour la fonction d'autocorélation partielles (PACF ou FACP). Nous choisissons donc $p_{max} = 4$ et $q_{max} = 2$. Nous avons donc dû discriminer entre tous les modèles ARMA(p, q) où $0 \leq p \leq 4$ et $0 \leq q \leq 2$. Pour ce faire, nous avons utilisé le test de nullité des paramètres (pour vérifier si le modèle peut être simplifié et donc bien ajusté) et les tests sur les résidus ou le test de Portmanteau (test de Ljung-Box en R) pour vérifier si les résidus sont proches d'un bruit blanc et donc non autocorrélés (donc pour vérifier si le modèle est suffisamment riche). Cette première sélection nous a permis, en estimant tous ces modèles, de ne retenir que les modèles bien ajustés et valides. Nous avons finalement obtenu 3 modèles bien ajustés et valides pour notre série différenciée : ARMA(2,0), ARMA(1,1) et ARMA(0,2).

	ARMA(2,0)	ARMA(1,1)	ARMA(0,2)
AIC	1099.188	1093.688	1094.237
BIC	1110.838	1105.339	1105.887

TABLE 1 – Valeurs des critères d'information AIC et BIC des modèles sélectionnés

Pour choisir entre ces 3 modèles, nous avons ensuite examiné celui qui minimise les deux critères d'information, à savoir l'AIC et le BIC. Nous remarquons que le modèle ARIMA(1,1,1) a le plus petit AIC et BIC, nous avons donc conservé ce modèle pour la suite. Les coefficients estimés de ce modèle et leurs erreurs standard peuvent être lus ci-dessous.

	AR(1)	MA(1)
Coefficient	0.5016	-0.8196
Écart type	0.1275	0.0826

TABLE 2 – Estimation de l'ARMA(1,1)

2.2 Modèle ARIMA(p,d,q)

Selon les estimations faites ci-dessus, notre modèle ARMA(1,1) est le suivant : $X_t - 0.5016X_{t-1} = \epsilon_t + 0.8196\epsilon_{t-1}$.

Ainsi, le modèle ARIMA vérifie : $\phi(B)(1-B)X_t = \psi(B)\epsilon_t$ où $\phi(X) = 1 - 0.501X$ et $\psi(X) = 1 + 0.8196X$.

Par conséquent, nous avons : $\Delta X_t - 0.5016\Delta X_{t-1} = \epsilon_t + 0.8196\epsilon_{t-1}$.

3 Prévision

3.1 Région de confiance de niveau α sur les valeurs futures (X_{T+1}, X_{T+2})

Soit $\epsilon \sim N(0, \sigma_\epsilon^2)$ (i.i.d.) avec $\sigma_\epsilon^2 > 0$, et soit T la longueur de la série. Nous utilisons également un modèle ARMA(1,1) pour la série différenciée $\Delta(X_t)$. Pour simplifier la notation, nous notons dans ce qui suit Y_t la série différenciée :

$$Y_t = \phi_1 Y_{t-1} + \epsilon_t - \psi_1 \epsilon_{t-1}$$

Étant donné que $E[\epsilon_{T+h}|Y_T, Y_{T-1}, \ldots] = 0, \forall h > 0$, les meilleures prévisions linéaires de Y_{T+1}, Y_{T+2} sont définies par :

$$\begin{cases} \hat{Y}_{T+1|T} = EL(X_{T+1|T}, X_{T-1}, \ldots) = \phi_1 Y_T - \psi_1 \epsilon_T \\ \hat{Y}_{T+2|T} = EL(X_{T+2|T}, X_{T-1}, \ldots) = \phi_1 \hat{Y}_{T+1|T} \end{cases}$$

où $\hat{Y}_{T+1|T}$ désigne l'espérance linéaire de Y_{T+1} conditionnelle à T.

Les erreurs de prévision sont donc écrites :

$$\begin{cases} e(T+1) = Y_{T+1} - \hat{Y}_{T+1|T} = \phi_1 Y_T + \epsilon_{T+1} - \psi_1 \epsilon_T - (\phi_1 Y_T - \psi_1 \epsilon_T) = \epsilon_{T+1} \\ e(T+2) = Y_{T+2} - \hat{Y}_{T+2|T} = \phi_1 Y_{T+1} + \epsilon_{T+2} - \psi_1 \epsilon_{T+1} - (\phi_1 \hat{Y}_{T+1|T}) = \phi_1 (Y_{T+1} - \hat{Y}_{T+1|T}) + \epsilon_{T+2} - \psi_1 \epsilon_{T+1} = \phi_1 \epsilon_{T+1} \end{cases}$$

Par conséquent, nous avons :

$$\begin{cases} V(e(T+1)) = V(Y_{T+1} - \hat{Y}_{T+1|T}) = V(\epsilon_{T+1}) = \sigma_\epsilon^2 \\ V(e(T+2)) = V(Y_{T+2} - \hat{Y}_{T+2|T}) = V((\phi_1 - \psi_1)\epsilon_{T+1} + \epsilon_{T+2}) = (\phi_1 - \psi_1)^2 \sigma_\epsilon^2 + \sigma_\epsilon^2 \\ Cov(e(T+1), e(T+2)) = Cov(\epsilon_{T+1}, (\phi_1 - \psi_1)\epsilon_{T+1} + \epsilon_{T+2}) = (\phi_1 - \psi_1)\sigma_\epsilon^2 \end{cases}$$

car nous avons supposé que les résidus sont un bruit blanc fort (gaussien et indépendant).

Par conséquent, nous avons :

$$\begin{pmatrix} e(T+1) \\ e(T+2) \end{pmatrix} = \begin{pmatrix} \epsilon_{T+1} \\ (\phi_1 - \psi_1)\epsilon_{T+1} + \epsilon_{T+2} \end{pmatrix} \sim N(0, \Sigma)$$

où

$$\Sigma = \begin{pmatrix} \sigma_\epsilon^2 & (\phi_1 - \psi_1)\sigma_\epsilon^2 \\ (\phi_1 - \psi_1)\sigma_\epsilon^2 & (\phi_1 - \psi_1)^2 \sigma_\epsilon^2 + \sigma_\epsilon^2 \end{pmatrix}$$

Si nous écrivons : $Y = \begin{pmatrix} Y_{T+1} - \hat{Y}_{T+1|T} \\ Y_{T+2} - \hat{Y}_{T+2|T} \end{pmatrix}$, nous savons que $Y' \Sigma^{-1} Y \sim \chi^2(2)$.

Ainsi, nous pouvons écrire la région de prévision de confiance α pour (Y_{T+1}, Y_{T+2}) :

$$R_\alpha = \{y \in R^2 \mid (y - \begin{pmatrix} \hat{Y}_{T+1|T} \\ \hat{Y}_{T+2|T} \end{pmatrix})' \Sigma^{-1} (y - \begin{pmatrix} \hat{Y}_{T+1|T} \\ \hat{Y}_{T+2|T} \end{pmatrix}) \leq q_{1-\alpha}\}$$

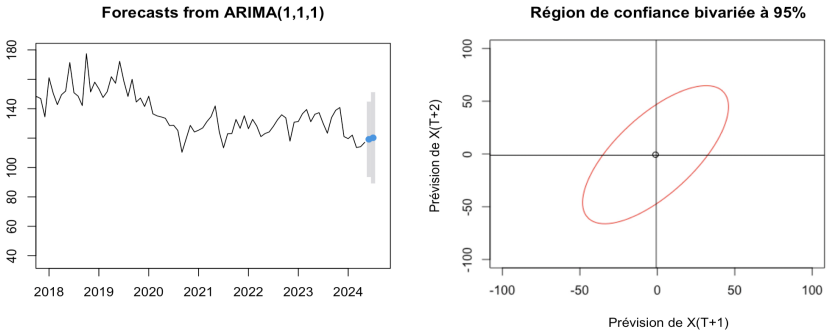
Avec q le quantile d'ordre $1-\alpha$ d'une loi $\chi^2(2)$.

3.2 Hypothèses utilisées pour obtenir cette région

Le calcul de la région de confiance de niveau α pour les valeurs futures repose sur deux hypothèses principales. La première est que le modèle théorique est identifié, c'est-à-dire que les coefficients théoriques sont identiques aux coefficients estimés. Par conséquent, la variance est connue et la variance estimée est égale à la variance théorique σ^2. La deuxième hypothèse est que les innovations (ϵ_t) sont un bruit blanc gaussien fort. Cela signifie qu'elles sont indépendantes et identiquement distribuées, $\epsilon \sim N(0, \sigma_\epsilon^2)$ avec $\sigma_\epsilon^2 > 0$.

3.3 Représentation graphique de la région de confiance

Nous avons représenté les régions de confiance des prévisions dans les figures ci-dessous. La figure de gauche montre la prévision univariée de deux valeurs de X_{T+1}, X_{T+2} avec un modèle ARIMA(1,1,1) et un intervalle de confiance à 5%. La figure de droite représente la région de confiance bivariée, elliptique, à 95% de X_{T+1}, X_{T+2}.



Nous remarquons que notre région de confiance est relativement large et donc peu précise sur sa prévision. D'une part, l'hypothèse de résidus gaussiens est discutable. D'autre part, la crise du covid perturbe la série et ajoute de l'incertitude sur l'avenir.

3.4 Question ouverte

Soit Y_t une série temporelle stationnaire disponible de $t = 1$ à T. Nous supposons que Y_{T+1} est disponible plus rapidement que X_{T+1}.

La connaissance de Y_{T+1} peut améliorer la prévision de X_{T+1} si Y_t cause instantanément X_t au sens de Granger, c'est-à-dire :

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

La corrélation entre les deux résidus d'un modèle VAR de (X, Y) caractérise cette condition, et nous pourrions la tester avec un test de Wald.

```
# install.packages("zoo")
# install.packages("tseries")
# install.packages("fUnitRoots")
# install.packages("ellipsis")
# install.packages("carData")
# install.packages("car")

# Chargement des bibliothèques nécessaires
library(tseries)
library(stats)
library(forecast)
library(fUnitRoots)
library(ggplot2)
library(readr)
library(zoo)         # Gestion facile des séries temporelles
library(ellipse)     # Fonction pour dessiner des ellipses

### Importation des données ###

path_data <- "/Users/maximecoppa/Desktop/TP noté 2"
setwd(path_data)
datafile <- "monthly_values.csv" # Définir le fichier de données

# Importer un fichier .csv dans un objet de classe data.frame
data <- read.csv(datafile, sep = ";")
data.source <- zoo(data[[2]])
head(data)

### PARTIE 1 : les données ###

# Prétraitement des données #
# QUESTION 1 - 2 - 3 #

# Transformation des données afin de pouvoir les étudier
data <- data[4:nrow(data),] # Suppression des trois premières lignes (en-têtes ou
informations non pertinentes)
data <- data[,-3] # Suppression de la dernière colonne (non utilisée)
colnames(data) <- c("date", "index_production") # Renommer les colonnes pour plus de
clarté

head(data)

# Suppression des 4 dernières lignes (pour comparer plus tard nos prévisions aux valeurs
réelles)
data <- data[1:(nrow(data)-4), ]
data

# Conversion des dates
dates <- as.yearmon(seq(from = 2013, to = 2024 + 04/12, by = 1/12))
index <- zoo(as.numeric(data$index_production), order.by = dates)

# Différenciation de la série pour la rendre stationnaire
dindex <- diff(index, 1)



# Tracé de la série index
plot.ts(index, xlab = "Dates", ylab = "Index de production")

# Tracé de la série différenciée dindex
plot.ts(dindex, xlab = "Dates", ylab = "Index différencié de production de tapis et
moquettes")

### Tests de stationnarité
### Nous allons maintenant effectuer trois tests (test de Dickey-Fuller augmenté, test
de Phillips-Perron et test KPSS) afin de vérifier la stationnarité des séries index et
dindex.
```

```
## Tests sur la série index (série non différenciée)

# Test de Dickey-Fuller augmenté

# Avant de réaliser les tests de racine unitaire, nous devons vérifier s'il y a un
intercept et/ou une tendance linéaire non nulle.
# Régressons index sur ses dates pour vérifier
summary(lm(index ~ dates))


# Test ADF sans constante et sans tendance
adf <- adfTest(index, lag=0, type="nc")
print(adf)

# Fonction pour les tests d'autocorrélation des résidus
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))
}

# Tests d'autocorrélation des résidus de l'ADF
Qtests(adf@test$lm$residuals, 24, fitdf = length(adf@test$lm$coefficients))

# Nous rejetons l'absence d'autocorrélation des résidus au moins une fois, invalidant
ainsi le test ADF sans décalages.
# Ajoutons des décalages de ΔXt jusqu'à ce que les résidus ne soient plus autocorrélés.

adfTest_valid <- function(series, kmax=24, adftype="ct") {
  k <- 0
  noautocorr <- 0

  while (noautocorr == 0 && k <= kmax) {
    cat(paste0("ADF avec ", k, " décalages : résidus OK? "))
    adf <- adfTest(series, lags=k, type=adftype)

    residuals <- adf@test$lm$residuals

    pvals <- Qtests(residuals, 24, fitdf = length(adf@test$lm$coefficients))[, "pval"]

    if (all(pvals < 0.05, na.rm=TRUE)) {
      noautocorr <- 1
      cat("OK \n")
    } else {
      cat("non \n")
      k <- k + 1
    }
  }

  return(adf)
}

adf <- adfTest_valid(index, 24, adftype="nc")

# Nous avons dû considérer N décalages sur le test ADF pour éliminer l'autocorrélation
des résidus.
# La racine unitaire n'est pas rejetée au niveau de 95% pour la série en niveaux, la
série est donc au moins I(1).

Qtests(adf@test$lm$residuals, 24, fitdf = length(adf@test$lm$coefficients))
adf
# La racine unitaire n'est pas rejetée au niveau de 95% pour la série en niveaux, la
série est donc au moins I(1).

# Test de Phillips-Perron : l'hypothèse nulle n'est pas rejetée au niveau de 95%, donc
```

```
la série index n'est pas stationnaire.
pp.test(index)

# Test KPSS : l'hypothèse nulle est rejetée au niveau de 95%, ce qui confirme que la
série index n'est pas stationnaire.
kpss.test(index)

# Test adf : nous testons pour vérifier nos résultats
adf.test(index)

## Tests sur la série 'dindex' (la différenciée d'ordre 1)

# La représentation graphique précédente semble montrer l'absence de constante et de
tendance.
# Vérifions avec une régression :

# Régression pour vérifier la série différenciée
summary(lm(dindex ~ dates[-1]))

adf <- adfTest_valid(dindex, 24, "nc")
Qtests(adf@test$lm$residuals, 24, fitdf = length(adf@test$lm$coefficients))
adf
# Le test rejette l'hypothèse de racine unitaire (p-value < 0.05), nous dirons donc que
la série différenciée est "stationnaire".
# Dindex est donc I(1).

pp.test(dindex)
# Le test rejette l'hypothèse de racine unitaire (p-value < 0.05), confirmant ainsi que
la série dindex est stationnaire.

kpss.test(dindex)
# Le test accepte l'hypothèse nulle (p-value > 0.10), confirmant ainsi que la série
dindex est stationnaire.

### PARTIE 2 : Modèles ARMA ###

#### IDENTIFICATION des ARMA(p,q) ####
# Questions 4 et 5 #

# Regardons les autocorrelogrammes complets et partiels de dindex pour trouver p et q.

par(mfrow = c(1, 2))
pacf(dindex, 24)
acf(dindex, 24) # Nous regardons les décalages de deux ans

pmax <- 4
qmax <- 2

# Fonction de test de la signification statistique des coefficients
signif <- 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))
}

# Le modèle est valide si les résidus ne sont pas autocorrélés. Nous pouvons tester cela
en utilisant le test de Ljung-Box de l'hypothèse nulle de nullité conjointe des
autocorrélations jusqu'à un ordre donné k.

# Fonction pour estimer un arima et vérifier son ajustement et sa validité (avec tests
de signification et de Ljung-Box)
modelchoice <- function(p, q, data = dindex, k = 24) {
  estim <- try(arima(data, c(p, 0, 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 signif(estim)[3, p] <= 0.05
```

```
  masignif <- if (q == 0) NA else signif(estim)[3, p + q] <= 0.05
  resnocorr <- sum(Qtests(estim$residuals, 24, length(estim$coef) - 1)[,2] <= 0.05,
na.rm = TRUE) == 0
  checks <- c(arsignif, masignif, resnocorr)
  ok <- as.numeric(sum(checks, na.rm = TRUE) == (3 - sum(is.na(checks))))
  return(c("p" = p, "q" = q, "arsignif" = arsignif, "masignif" = masignif, "resnocorr" =
resnocorr, "ok" = ok))
}

# Fonction pour estimer et vérifier tous les arima(p,q) avec p <= pmax et q <= qmax
armamodelchoice <- 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("Calcul du ARMA(", p, ",", q, ") \n"))
    modelchoice(p, q)
  }))
}

armamodels <- armamodelchoice(pmax, qmax) # Estimation de tous les arima

selec <- armamodels[armamodels[,"ok"] == 1 & !is.na(armamodels[,"ok"]),] # Modèles bien
ajustés et validés
selec
# Nous avons 3 modèles bien ajustés et valides : ARMA (2,0), ARMA(1,1) et ARMA(0, 2)

pqs <- apply(selec, 1, function(row) list("p" = as.numeric(row[1]), "q" =
as.numeric(row[2]))) # Création d'une liste des ordres p et q des modèles candidats
names(pqs) <- paste0("arma(", selec[, 1], ",", selec[, 2], ")") # Renommer les éléments
de la liste
models <- lapply(pqs, function(pq) arima(dindex, c(pq[["p"]], 0, pq[["q"]]))) # Création
d'une liste des modèles candidats estimés
vapply(models, FUN.VALUE = numeric(2), function(m) c("AIC" = AIC(m), "BIC" = BIC(m))) #
Calcul des AIC et BIC des modèles candidats
# L'ARIMA(1,1,1) minimise les critères d'information AIC et BIC.

arima111 <- arima(index, c(1, 1, 1), include.mean = FALSE)
model_arima111 <- arima(index, order = c(1, 1, 1))

# Afficher le modèle complet pour plus de détails
summary(model_arima111)

### PARTIE 3 : Prévision ###

# Prévision de 4 valeurs futures

# Questions 6, 7 et 8 #

# Nous traçons la prévision grâce à la méthode forecast
prev <- forecast(arima111, h = 2, level = 95)
plot(prev, xlim = c(2018, 2024 + 7/12))
prev

# Extraction des coefficients du modèle et de la variance des résidus
arima111

# Extraction des prévisions et de leurs variances
XT1 <- prev$mean[1]
XT2 <- prev$mean[2]
sigma2 <- arima111$sigma2

# La commande coef donne d'abord les coefficients AR puis les coefficients MA
phi_1 <- as.numeric(arima111$coef[1])
psi_1 <- as.numeric(arima111$coef[2])

# Calcul de la matrice de covariance Sigma
Sigma <- matrix(c(sigma2, (phi_1 - psi_1) * sigma2, (phi_1 - psi_1) * sigma2, (phi_1 -
```

```
psi_1)^2 * sigma2 + sigma2), ncol = 2)

# Ensuite, nous représentons la région de confiance bivariée à 95%

require(ellipse)
require(ellipsis)
require(car)
library(ellipse)


plot(XT1, XT2, xlim = c(70,160), ylim = c(50,200),
     xlab = "Prévision de X(T+1)",
     ylab = "Prévision de X(T+2)",
     main = "Région de confiance bivariée à 95%")

# Ajouter l'ellipse de confiance
lines(ellipse(Sigma, centre=c(XT1, XT2)), type="l", col="red")
abline(h=XT2, v=XT1)
```