Indice brut de la production industrielle : Construction aéronautique et spatiale

HADDOUCHE Théo, EL BAZ Avner

2025-04-03

Introduction

Librairies

```r
library(tseries)
```

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

```r
library(readr)
library(forecast)
library(ggplot2)
```

Import et délimitation du dataset

```r
valeurs_mensuelles <- read_delim("valeurs_mensuelles.csv", delim = ";", col_types = cols())
vm <- valeurs_mensuelles[-c(1, 2, 3), ]
colnames(vm) <- c("date","value","code")
vm <- vm[nrow(vm):1, ]
vm$value <- as.numeric(vm$value)
vm$diff <- c(NA,diff(vm$value))

str(vm)
```

```
## tibble [421 x 4] (S3: tbl_df/tbl/data.frame)
##  $ date : chr [1:421] "1990-01" "1990-02" "1990-03" "1990-04" ...
##  $ value: num [1:421] 107.3 96.4 102.6 78.5 71.7 ...
##  $ code : chr [1:421] "A" "A" "A" "A" ...
##  $ diff : num [1:421] NA -10.97 6.2 -24.12 -6.81 ...
```

```r
head(vm)
```

```
## # A tibble: 6 x 4
##   date    value code    diff
##   <chr>   <dbl> <chr>  <dbl>
## 1 1990-01 107.  A      NA
## 2 1990-02  96.4 A     -11.0
## 3 1990-03 103.  A       6.2
## 4 1990-04  78.5 A     -24.1
## 5 1990-05  71.6 A      -6.81
## 6 1990-06  80.8 A       9.20
```

Part I : The Data

1. What does the chosen series represent ? (sector, potential data processing, logarithmic transformation, etc.)

La série représente la production

2. Transform the series to make it stationary if necessary (differentiate it, correct the deterministic trend, etc.). Thoroughly justify your choices.

```r
serie_ts <- ts(vm$value, start = c(1990, 01), frequency = 12)
diff_series <- ts(vm$diff, start = c(1990, 02), frequency = 12)
diff_series <- na.omit(diff_series)
# Dickey-Fuller Test
adf.test(serie_ts, alternative="stationary")
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  serie_ts
## Dickey-Fuller = -2.0775, Lag order = 7, p-value = 0.5447
## alternative hypothesis: stationary
```

```r
adf.test(diff_series, alternative="stationary")
```

```
## Warning in adf.test(diff_series, alternative = "stationary"): p-value smaller
## than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  diff_series
## Dickey-Fuller = -10.457, Lag order = 7, p-value = 0.01
## alternative hypothesis: stationary
```

```r
pp.test(diff_series, alternative="stationary")
```

```
## Warning in pp.test(diff_series, alternative = "stationary"): p-value smaller
## than printed p-value
```

```
##
##  Phillips-Perron Unit Root Test
##
## data:  diff_series
## Dickey-Fuller Z(alpha) = -430.03, Truncation lag parameter = 5, p-value
## = 0.01
## alternative hypothesis: stationary
```

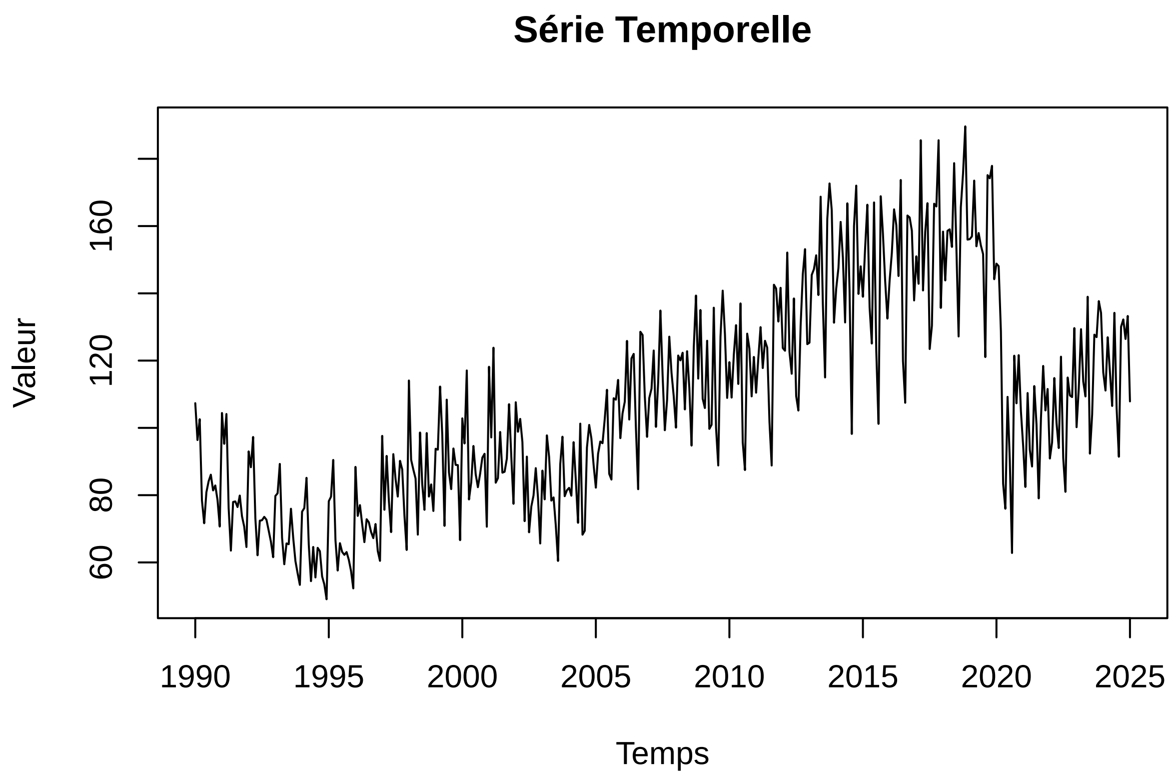
```r
kpss.test((diff_series))
```

```
## Warning in kpss.test((diff_series)): p-value greater than printed p-value
```

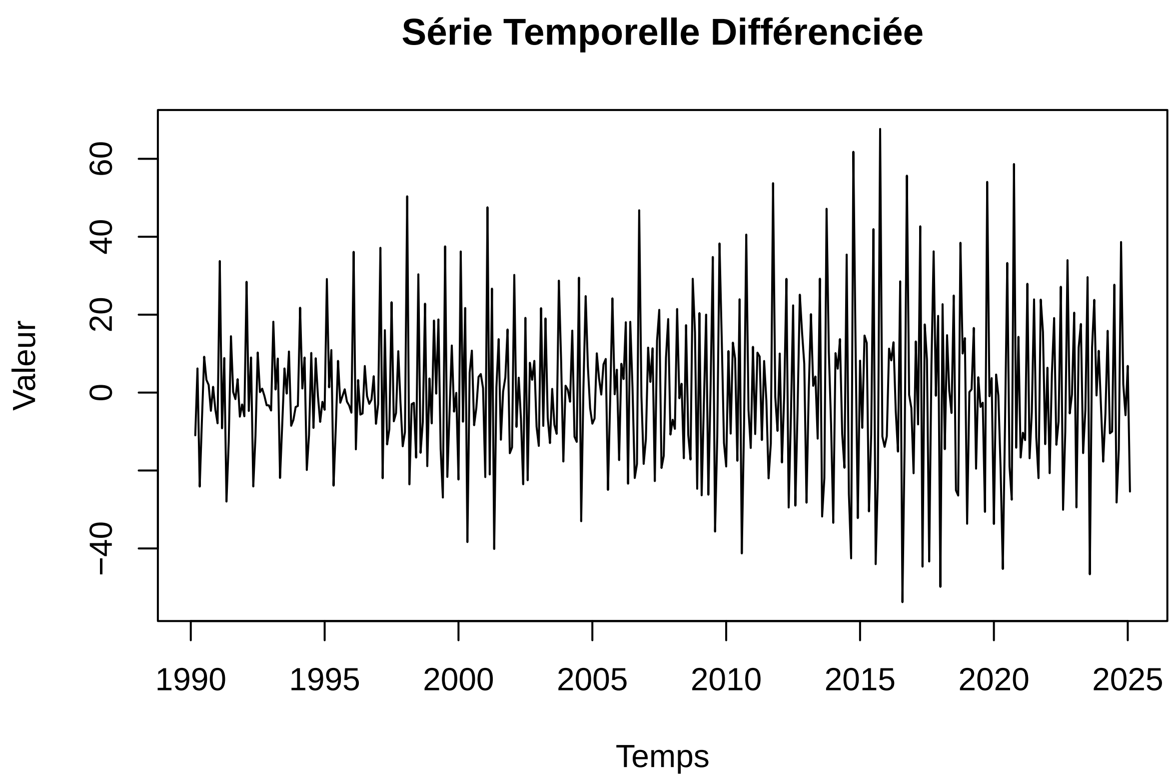
```
##
##  KPSS Test for Level Stationarity
##
## data:  (diff_series)
## KPSS Level = 0.031626, Truncation lag parameter = 5, p-value = 0.1
```

3. Graphically represent the chosen series before and after transforming it.

```r
plot(serie_ts, main="Série Temporelle", xlab="Temps", ylab="Valeur")
```




```r
plot(diff_series, main="Série Temporelle Différenciée", xlab="Temps", ylab="Valeur")
```



Part II : ARMA models

4. Pick (and justify your choice) an ARMA(p,q) model for your corrected time series Xt. Estimate the model parameters and check its validity.

```r
# Calculer l'ACF et le PACF
acf_values <- acf(diff_series, lag.max = 50, plot = FALSE)
pacf_values <- pacf(diff_series, lag.max = 50, plot = FALSE)

# Inspecter la structure de l'objet
str(acf_values)
```

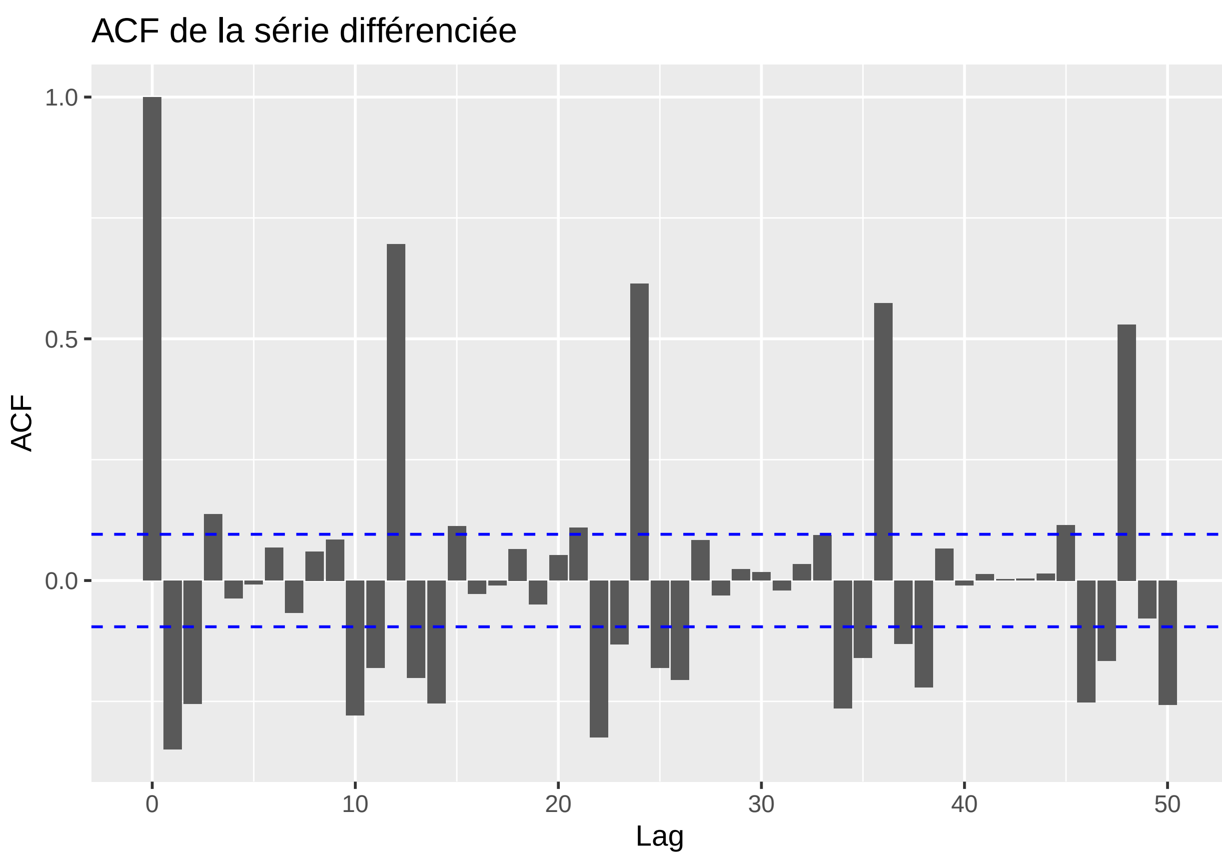
```
## List of 6
##  $ acf   : num [1:51, 1, 1] 1 -0.3485 -0.2547 0.1375 -0.0369 ...
##  $ type  : chr "correlation"
##  $ n.used: int 420
##  $ lag   : num [1:51, 1, 1] 0 0.0833 0.1667 0.25 0.3333 ...
##  $ series: chr "diff_series"
##  $ snames: NULL
##  - attr(*, "class")= chr "acf"
```

```r
str(pacf_values)
```

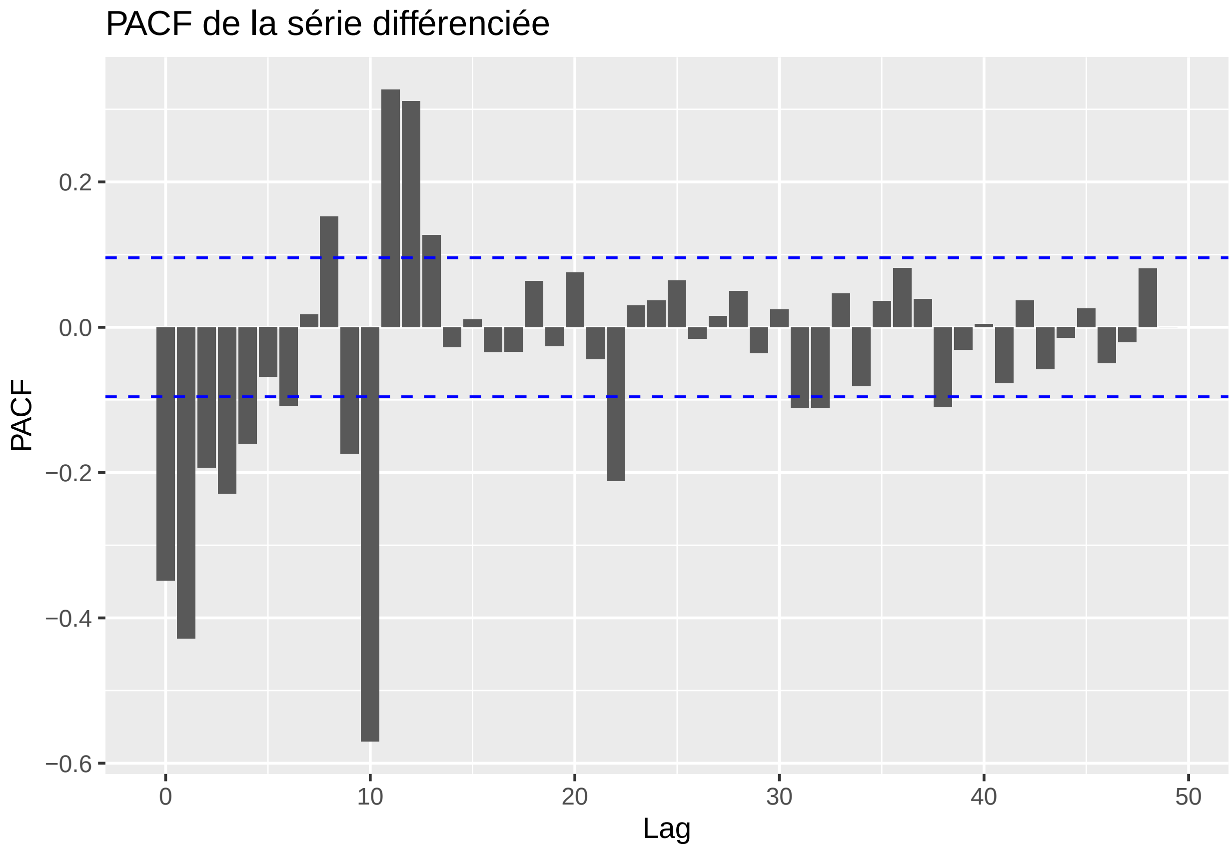
```
## List of 6
##  $ acf   : num [1:50, 1, 1] -0.349 -0.428 -0.193 -0.229 -0.16 ...
##  $ type  : chr "partial"
##  $ n.used: int 420
```

```
##  $ lag   : num [1:50, 1, 1] 0.0833 0.1667 0.25 0.3333 0.4167 ...
##  $ series: chr "diff_series"
##  $ snames: NULL
##  - attr(*, "class")= chr "acf"
```

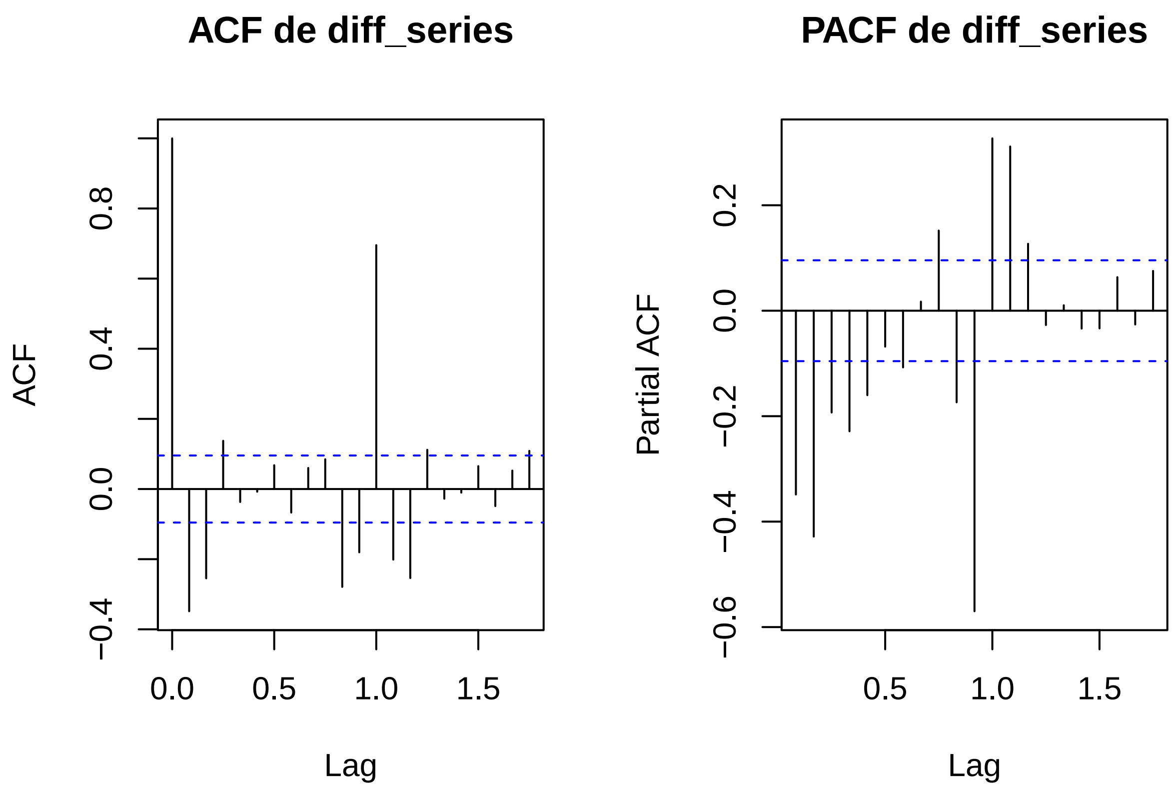
```r
# Extraire les valeurs d'ACF et de PACF correctement
acf_vals <- acf_values$acf[,,1]  # Ajustez selon la structure réelle
pacf_vals <- pacf_values$acf[,,1]  # Ajustez selon la structure réelle

length_acf <- length(acf_vals)
length_pacf <- length(pacf_vals)

# Créer les DataFrames pour ggplot2
acf_df <- data.frame(Lag = 0:(length_acf - 1), ACF = acf_vals)
pacf_df <- data.frame(Lag = 0:(length_pacf - 1), PACF = pacf_vals)

# Tracer l'ACF avec ggplot2
ggplot(acf_df, aes(x = Lag, y = ACF)) +
  geom_bar(stat = "identity") +
  geom_hline(yintercept = c(-1.96/sqrt(length(diff_series)), 1.96/sqrt(length(diff_series))), linetype=
  labs(title="ACF de la série différenciée", x="Lag", y="ACF")
```




```r
# Tracer le PACF avec ggplot2
ggplot(pacf_df, aes(x = Lag, y = PACF)) +
  geom_bar(stat = "identity") +
  geom_hline(yintercept = c(-1.96/sqrt(length(diff_series)), 1.96/sqrt(length(diff_series))), linetype=
  labs(title="PACF de la série différenciée", x="Lag", y="PACF")
```




```r
# Définir la disposition des graphiques
par(mfrow=c(1,2), mar=c(5,4,4,2) + 0.1)

# Tracer l'ACF avec un ajustement de l'axe des abscisses
acf(diff_series, main="ACF de diff_series", lag.max = 21, xlab="Lag")


# Tracer le PACF avec un ajustement de l'axe des abscisses
pacf(diff_series, main="PACF de diff_series", lag.max = 21)
```




```
# Réinitialiser la disposition des graphiques
par(mfrow=c(1,1))
```

On peut voir que

```
# Charger les bibliothèques nécessaires
library(forecast)

# Définir les valeurs de p et q
p_values <- 0:10
q_values <- 0:10

# Créer une matrice pour stocker les résultats
results <- expand.grid(p = p_values, q = q_values)
results$AIC <- NA
results$BIC <- NA

# Boucle pour ajuster les modèles et calculer AIC & BIC
for (i in 1:nrow(results)) {
  p <- results$p[i]
  q <- results$q[i]  # Utiliser "i" au lieu de "j"

  # Ajuster le modèle ARMA(p, q)
  model <- tryCatch(
    arima(diff_series, order = c(p, 0, q)),  # Modèle ARMA sur la série différenciée
    error = function(e) return(NULL)
  )
```

```r
  # Si le modèle est valide, stocker AIC et BIC
  if (!is.null(model)) {
    results$AIC[i] <- model$aic
    results$BIC[i] <- BIC(model)
  }
}
```

```
## Warning in arima(diff_series, order = c(p, 0, q)): possible convergence
## problem: optim gave code = 1
## Warning in arima(diff_series, order = c(p, 0, q)): possible convergence
## problem: optim gave code = 1
## Warning in arima(diff_series, order = c(p, 0, q)): possible convergence
## problem: optim gave code = 1
## Warning in arima(diff_series, order = c(p, 0, q)): possible convergence
## problem: optim gave code = 1
## Warning in arima(diff_series, order = c(p, 0, q)): possible convergence
## problem: optim gave code = 1
## Warning in arima(diff_series, order = c(p, 0, q)): possible convergence
## problem: optim gave code = 1
## Warning in arima(diff_series, order = c(p, 0, q)): possible convergence
## problem: optim gave code = 1
## Warning in arima(diff_series, order = c(p, 0, q)): possible convergence
## problem: optim gave code = 1
## Warning in arima(diff_series, order = c(p, 0, q)): possible convergence
## problem: optim gave code = 1
## Warning in arima(diff_series, order = c(p, 0, q)): possible convergence
## problem: optim gave code = 1
## Warning in arima(diff_series, order = c(p, 0, q)): possible convergence
## problem: optim gave code = 1
## Warning in arima(diff_series, order = c(p, 0, q)): possible convergence
## problem: optim gave code = 1
## Warning in arima(diff_series, order = c(p, 0, q)): possible convergence
## problem: optim gave code = 1
## Warning in arima(diff_series, order = c(p, 0, q)): possible convergence
## problem: optim gave code = 1
```

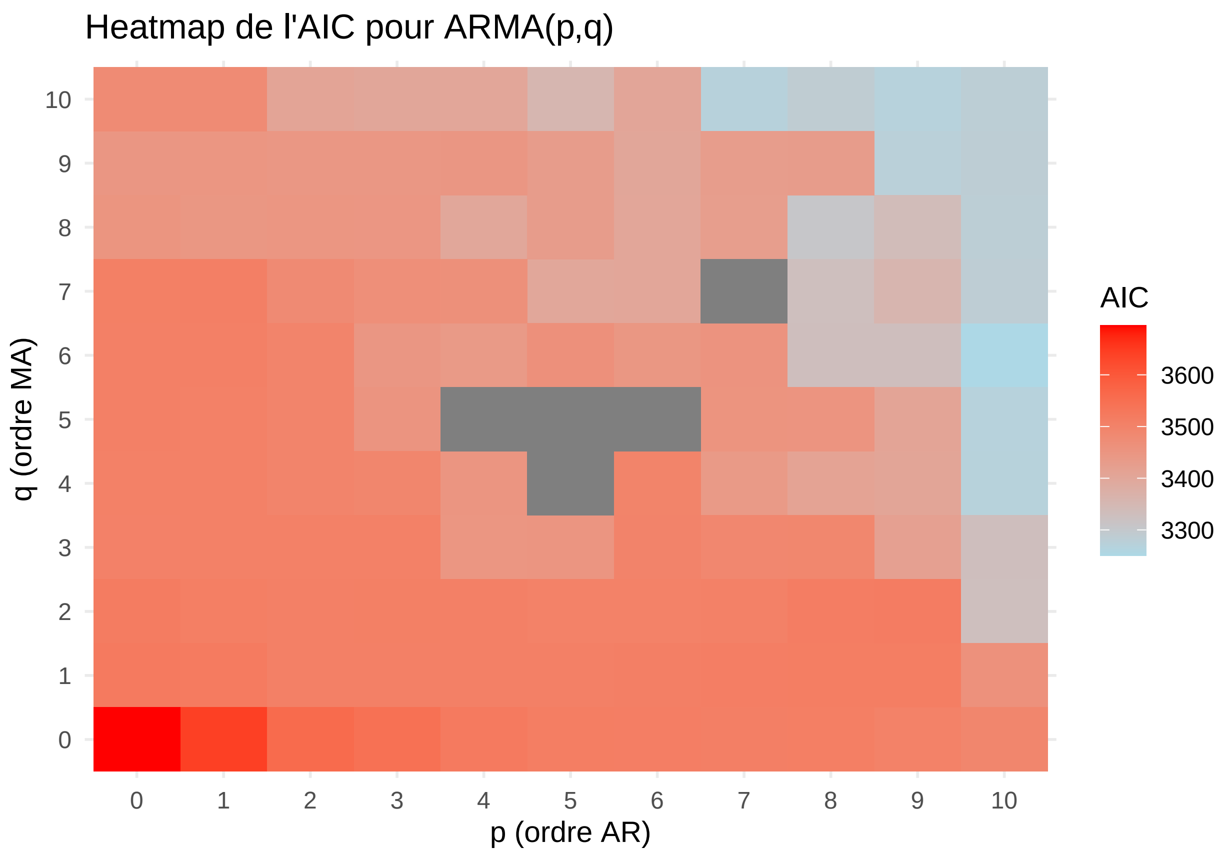
```r
# Afficher le tableau trié par AIC
results <- results[order(results$AIC), ]
print(results)
```

```
##      p  q      AIC      BIC
## 77  10  6 3251.054 3323.779
## 66  10  5 3270.712 3339.397
## 120  9 10 3270.764 3355.610
## 55  10  4 3271.304 3335.948
## 118  7 10 3271.454 3348.219
## 109  9  9 3276.691 3357.496
## 99  10  8 3283.187 3363.992
## 121 10 10 3283.380 3372.266
## 110 10  9 3285.380 3370.226
## 88  10  7 3286.343 3363.108
## 119  8 10 3289.416 3370.221
## 97   8  8 3307.447 3380.172
## 33  10  2 3328.689 3385.252
```

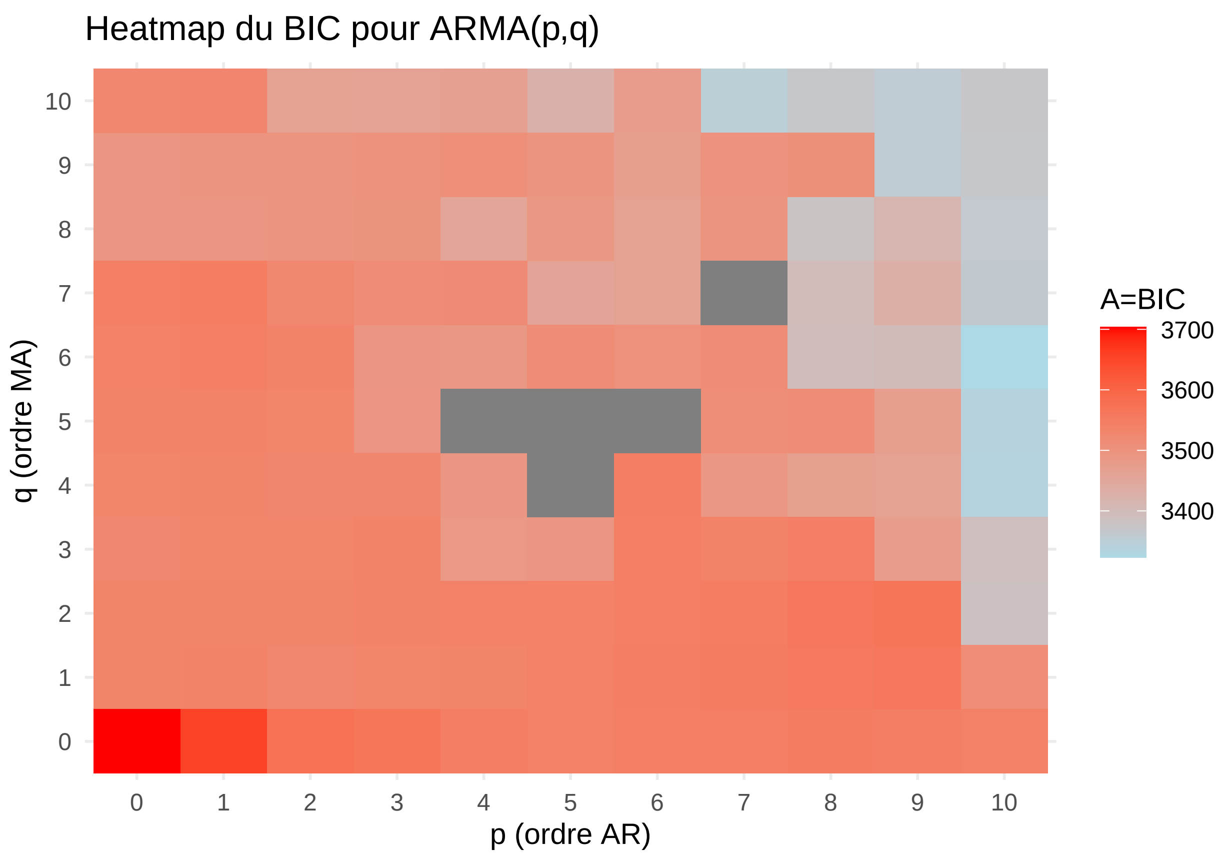
```
## 86    8  7 3329.252 3397.936
## 44   10  3 3330.289 3390.893
## 75    8  6 3330.517 3395.161
## 76    9  6 3331.245 3399.929
## 98    9  8 3337.964 3414.729
## 116   5 10 3356.147 3424.831
## 87    9  7 3358.881 3431.605
## 93    4  8 3400.057 3456.621
## 83    5  7 3401.223 3457.787
## 114   3 10 3402.018 3462.622
## 106   6  9 3402.043 3470.727
## 95    6  8 3402.256 3466.900
## 84    6  7 3403.206 3463.809
## 115   4 10 3403.626 3468.270
## 117   6 10 3405.891 3478.616
## 54    9  4 3406.420 3467.024
## 65    9  5 3408.289 3472.933
## 113   2 10 3409.375 3465.939
## 53    8  4 3412.298 3468.861
## 43    9  3 3419.991 3476.554
## 96    7  8 3427.205 3495.890
## 107   7  9 3428.747 3501.471
## 105   5  9 3430.630 3495.274
## 108   8  9 3430.750 3507.515
## 94    5  8 3431.019 3491.623
## 52    7  4 3438.672 3491.196
## 71    4  6 3438.787 3487.270
## 102   2  9 3445.550 3498.073
## 103   3  9 3446.460 3503.024
## 90    1  8 3447.450 3491.892
## 73    6  6 3447.459 3504.023
## 70    3  6 3447.551 3491.994
## 104   4  9 3448.364 3508.968
## 100   0  9 3448.372 3492.815
## 92    3  8 3448.603 3501.126
## 91    2  8 3449.449 3497.932
## 101   1  9 3449.449 3497.932
## 38    4  3 3450.008 3486.370
## 39    5  3 3451.397 3491.799
## 49    4  4 3452.737 3493.140
## 89    0  8 3453.038 3493.441
## 59    3  5 3453.942 3494.345
## 64    8  5 3455.027 3515.631
## 63    7  5 3455.832 3512.395
## 74    7  6 3456.446 3517.050
## 22   10  1 3461.782 3514.305
## 72    5  6 3464.991 3517.514
## 82    4  7 3466.410 3518.933
## 81    3  7 3467.770 3516.253
## 112   1 10 3478.997 3531.520
## 111   0 10 3479.191 3527.674
## 80    2  7 3480.510 3524.952
## 41    7  3 3489.105 3537.588
## 42    8  3 3490.705 3543.229
```

```
## 11  10  0 3492.799 3541.282
## 48   3  4 3492.851 3529.213
## 47   2  4 3496.975 3529.297
## 58   2  5 3497.116 3533.479
## 69   2  6 3497.597 3538.000
## 51   6  4 3498.734 3547.217
## 40   6  3 3500.292 3544.734
## 28   5  2 3503.256 3539.618
## 10   9  0 3503.792 3548.235
## 29   6  2 3504.162 3544.565
## 34   0  3 3504.328 3524.529
## 36   2  3 3505.175 3533.456
## 46   1  4 3505.221 3533.503
## 30   7  2 3505.789 3550.232
## 45   0  4 3506.253 3530.494
## 35   1  3 3506.557 3530.799
## 57   1  5 3506.597 3538.919
## 37   3  3 3506.670 3538.992
## 14   2  1 3507.034 3527.235
## 16   4  1 3507.087 3535.369
## 25   2  2 3507.569 3531.811
## 67   0  6 3507.679 3540.001
## 56   0  5 3508.069 3536.351
## 68   1  6 3508.259 3544.622
## 15   3  1 3508.582 3532.824
## 17   5  1 3509.086 3541.408
## 27   4  2 3509.086 3541.408
## 78   0  7 3509.280 3545.642
## 26   3  2 3509.328 3537.610
## 8    7  0 3509.982 3546.344
## 79   1  7 3510.119 3550.522
## 18   6  1 3510.800 3547.162
## 9    8  0 3511.904 3552.307
## 24   1  2 3511.976 3532.177
## 19   7  1 3512.550 3552.953
## 7    6  0 3513.294 3545.616
## 6    5  0 3513.453 3541.734
## 20   8  1 3513.982 3558.425
## 21   9  1 3514.036 3562.519
## 31   8  2 3515.204 3563.687
## 32   9  2 3517.218 3569.741
## 23   0  2 3519.250 3535.411
## 13   1  1 3521.562 3537.723
## 12   0  1 3522.605 3534.726
## 5    4  0 3523.103 3547.344
## 4    3  0 3545.001 3565.203
## 3    2  0 3559.514 3575.675
## 2    1  0 3642.803 3654.924
## 1    0  0 3695.342 3703.423
## 50   5  4       NA       NA
## 60   4  5       NA       NA
## 61   5  5       NA       NA
## 62   6  5       NA       NA
## 85   7  7       NA       NA
```

```r
ggplot(results, aes(x = factor(p), y = factor(q), fill = AIC)) +
  geom_tile() +
  scale_fill_gradient(low = "lightblue", high = "red") +
  labs(title = "Heatmap de l'AIC pour ARMA(p,q)",
       x = "p (ordre AR)",
       y = "q (ordre MA)",
       fill = "AIC") +
  theme_minimal()
```




```r
ggplot(results, aes(x = factor(p), y = factor(q), fill = BIC)) +
  geom_tile() +
  scale_fill_gradient(low = "lightblue", high = "red") +
  labs(title = "Heatmap du BIC pour ARMA(p,q)",
       x = "p (ordre AR)",
       y = "q (ordre MA)",
       fill = "A=BIC") +
  theme_minimal()
```




```r
best_model <- auto.arima(diff_series, ic="aic", seasonal = FALSE, approximation = FALSE, trace=TRUE)
```

```
##
##  ARIMA(2,0,2)            with non-zero mean : 3507.569
##  ARIMA(0,0,0)            with non-zero mean : 3695.342
##  ARIMA(1,0,0)            with non-zero mean : 3642.803
##  ARIMA(0,0,1)            with non-zero mean : 3522.605
##  ARIMA(0,0,0)            with zero mean     : 3693.342
##  ARIMA(1,0,2)            with non-zero mean : 3511.976
##  ARIMA(2,0,1)            with non-zero mean : 3507.034
##  ARIMA(1,0,1)            with non-zero mean : 3521.562
##  ARIMA(2,0,0)            with non-zero mean : 3559.514
##  ARIMA(3,0,1)            with non-zero mean : 3508.582
##  ARIMA(3,0,0)            with non-zero mean : 3545.001
##  ARIMA(3,0,2)            with non-zero mean : 3509.328
##  ARIMA(2,0,1)            with zero mean     : 3505.211
##  ARIMA(1,0,1)            with zero mean     : 3519.778
##  ARIMA(2,0,0)            with zero mean     : 3557.522
##  ARIMA(3,0,1)            with zero mean     : 3506.764
##  ARIMA(2,0,2)            with zero mean     : 3505.755
##  ARIMA(1,0,0)            with zero mean     : 3640.804
##  ARIMA(1,0,2)            with zero mean     : 3510.192
##  ARIMA(3,0,0)            with zero mean     : 3543.02
##  ARIMA(3,0,2)            with zero mean     : 3507.512
##
##  Best model: ARIMA(2,0,1)            with zero mean
```

```r
summary(best_model)
```

```
## Series: diff_series
## ARIMA(2,0,1) with zero mean
##
## Coefficients:
##          ar1      ar2      ma1
##       0.0369  -0.2285  -0.7433
## s.e.  0.0602   0.0546   0.0438
##
## sigma^2 = 243.1:  log likelihood = -1748.61
## AIC=3505.21   AICc=3505.31   BIC=3521.37
##
## Training set error measures:
##                    ME     RMSE      MAE      MPE    MAPE     MASE        ACF1
## Training set 0.225414 15.53502 12.27692 17.09322 199.557 1.064932 0.006816636
```

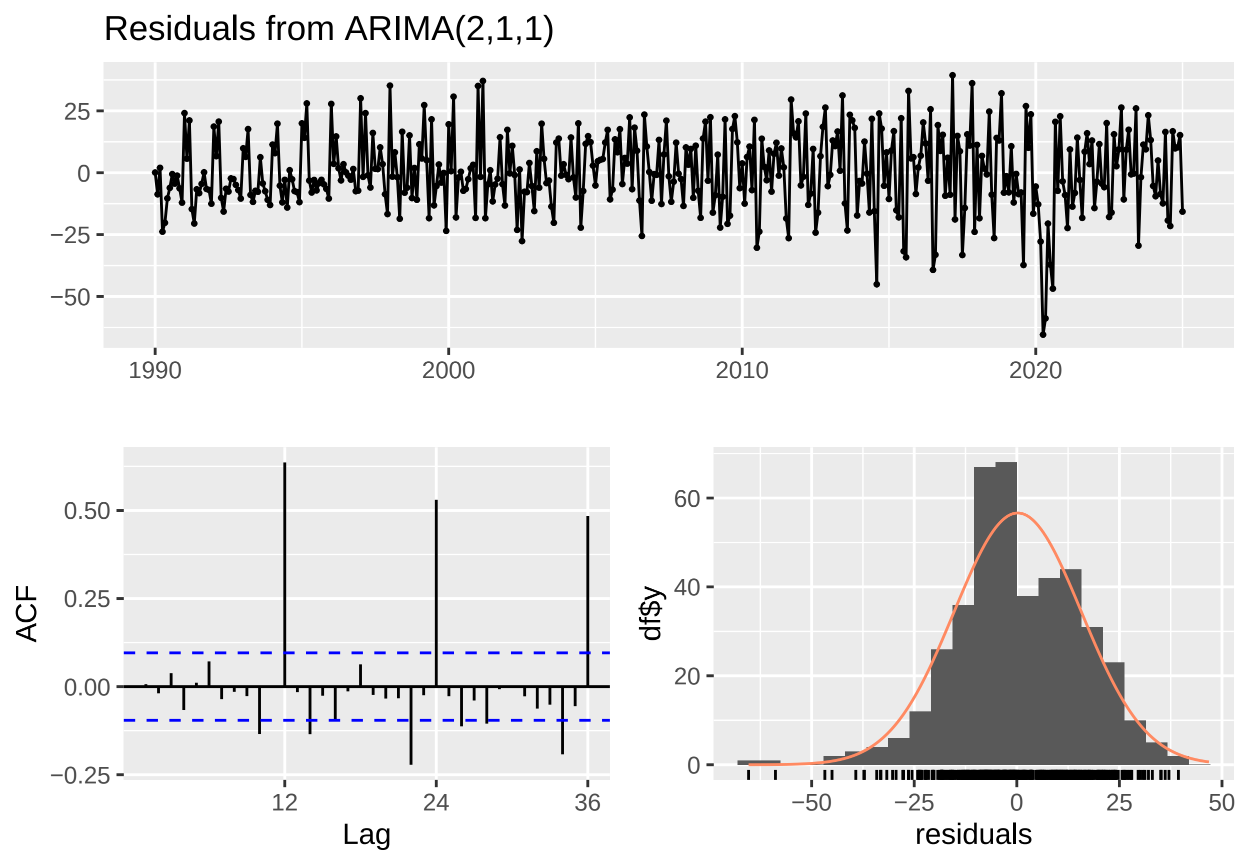
5. Write the ARIMA(p,d,q) model for the chosen series.

```r
modele_arima <- Arima(serie_ts, order = c(2, 1, 1))
summary(modele_arima)
```

```
## Series: serie_ts
## ARIMA(2,1,1)
##
## Coefficients:
##          ar1      ar2      ma1
##       0.0369  -0.2285  -0.7433
## s.e.  0.0602   0.0546   0.0438
##
## sigma^2 = 243.1:  log likelihood = -1748.61
## AIC=3505.21   AICc=3505.31   BIC=3521.37
##
## Training set error measures:
##                    ME     RMSE      MAE       MPE     MAPE     MASE        ACF1
## Training set 0.225134 15.51656 12.24802 -1.752699 11.94216 1.129351 0.006827056
```

Tracé des résidus

```r
checkresiduals(modele_arima)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,1)
## Q* = 353.63, df = 21, p-value < 2.2e-16
##
## Model df: 3.   Total lags used: 24
```