Rapport de séries temporelles - Modélisation ARMA

BOMBARD Thierry

Décembre 2020

1 Introduction

Dans ce rapport nous allons tenter d'ajuster un modèle tiré du domaine des séries temporelles, dans le but d'effectuer des prévisions sur certaines périodes. La démarche sera la suivante : nous allons simuler au hasard 200 observations qui suivent un certain processus stationnaire sous R, puis nous allons les combiner avec des données stockées dans un fichier « datas.txt ». La prochaine étape consistera à identifier les relations pertinentes entre ces observations, qui nous permettront par la suite d'estimer un modèle qui suivra ces mêmes relations. Le modèle en question sera de type ARMA (Auto Regressive Moving Average). Les packages utilisés dans notre analyse sont les suivants : tseries, forecast, TSA, ggfortify.

2 Les données

Les données seront générées à l'aide de la fonction « arima.sim() » du package « stats » inclus dans R. Nos n observations devront suivre un processus stationnaire de la forme suivante :

$$X_t = 0.4X_{t-1} + \varepsilon_t - \theta\varepsilon_{t-1}$$

avec $\theta = 0.54$ et n = 200.

Nous pouvons d'ores et déjà constater que notre processus stochastique est équivalent à des modèles AR et MA d'ordres 1. Effectivement, les décalages (t-i) de notre variable X_t et de notre variable ε sont au maximum de 1. Enfin, les processus AR et MA ont respectivement un coefficient de 0.4 et de 0.54. Toutes ces valeurs seront utilisées dans les arguments de la fonction « arima.sim ».

3 Simulation des données

Notre série est de la forme suivante :

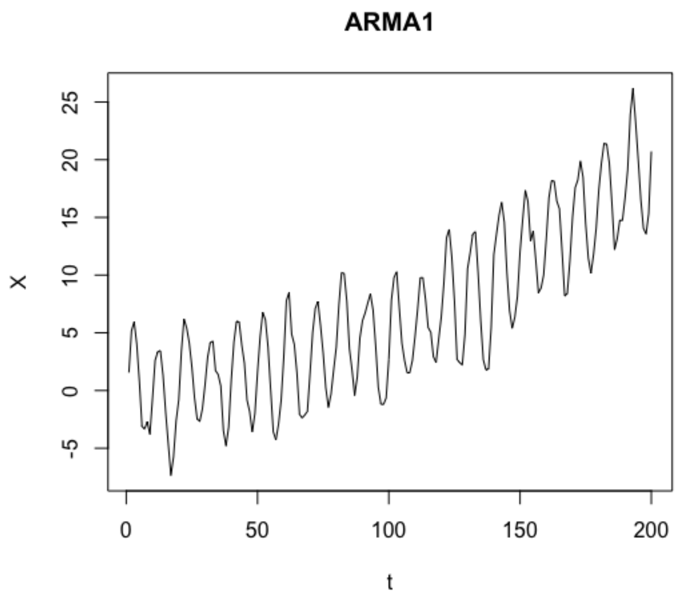

FIGURE 1 – Forme du processus stochastique

Code qui a permis de générer ce processus :

```
Input :
# simulation de nos données :
simul = arima.sim(model = list(ar=c(0.4), ma=c(0.54)), n = 200)
(x = read.table('data.txt', sep = ' '))
# combinaison des données du fichier data.txt + nos données simulées
simul2 = x[,1] + simul
plot(simul2, main = "ARMA1", xlab = "t", ylab = "X")
```

Nos données sont stockées dans la variable simul2.

4 Modélisation de la série

4.1 Stationnarité

Pour une première analyse de la série nous allons tester sa stationnarité avec un test de Kwiatkowski-Phillips-Schmidt-Shin (KPSS).
H_0 : notre simulation suit un processus stationnaire centré
H_1 : la série n'est pas stationnaire.

```
Input :
kpss.test(simul2)

Output :
KPSS Test for Level Stationarity

data:  simul2
KPSS Level = 3.2682, Truncation lag parameter = 4, p-value = 0.01
```

Nous allons rejeter H_0 au seuil de 5%, puisque sa p-valeur est inférieure à 5%. La série n'est donc pas stationnaire.

Une série temporelle peut être décomposée en 2 structures. Une structure déterministe qui est composée d'une tendance (m_t) et d'une composante périodique (s_t), puis une structure aléatoire qui est composée de variations (bruit blanc B_t).

Afin de pouvoir estimer un modèle pertinent par rapport à notre analyse, nous allons retirer les effets saisonniers et de tendances dans notre série, pour pouvoir ensuite identifier les informations qui ont permis de générer de tels effets. Supprimer ces effets-là nous permettra de conserver un bruit blanc qui est une propriété unique, c'est ce qui différencie une série par rapport à une autre.

4.2 Composante saisonnière

La saisonnalité dans une série peut être expliquée par l'inter-dépendance des observations (l'autocorrélation des périodes de notre série). Pour nous faire une idée sur la dépendance dans le temps, nous pouvons utiliser une fonction d'autocorrélation empirique (ACF) et une fonction d'autocorrélation empirique partielle (PACF). En observant les résultats de nos fonctions, nous pourrons déterminer le nombre de décalages nécéssaires (autocorrélés et significativement différents de 0) pour déterminer les ordres de nos processus AR et MA. Pour déterminer l' ordre p du processus AR nous utiliserons la PACF. Pour l'ordre q de MA, nous utiliserons l'ACF. Nous pourrons aussi utiliser l'EACF (fonction d'autocorrélation empirique étendue ou élargie) pour tenter de trouver l'optimum des ordres p et q. Les ordres p et q sont des paramètres à insérer dans notre modèle final ARIMA.

L'ACF va tester les hypothèses suivantes :
H_0 : les observations entre les lignes bleues sont indépendantes et identiquement distribuées (autrement dit iid).
H_1 : les observations entre les lignes bleues sont ne sont ni indépendantes et ni identiquement distribuées.

Voici l'état des inter-dépendances des observations de notre série en utilisant l'ACF :

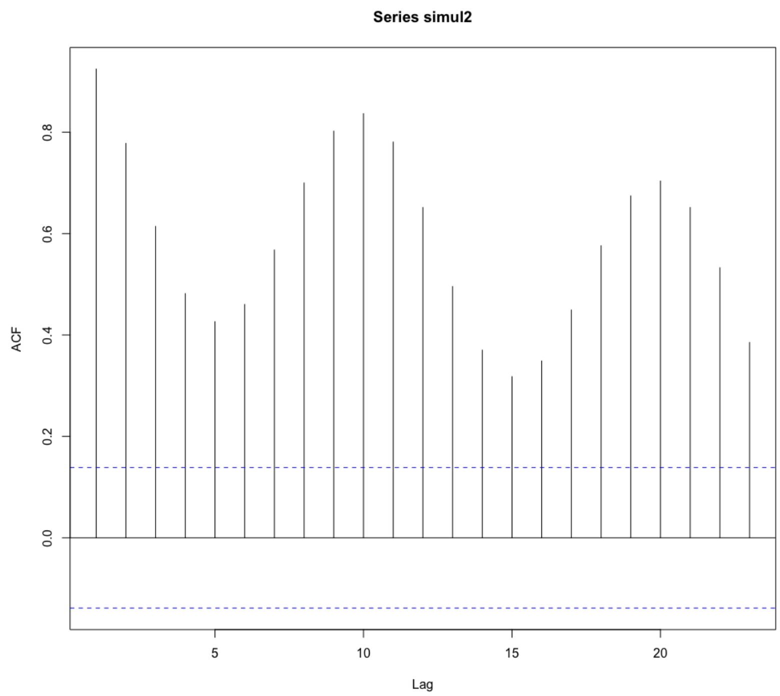

FIGURE 2 – ACF de la série simul2

Nous remarquons qu'il y a des oscillations ayant une forme assez proche d'un sinusoïde, et que tout nos décalages dépassent les seuils indiqués en pointillés bleu. Nous pouvons donc rejeter H_0. La série n'est pas iid.

Nous pouvons aussi visualiser les fréquences prédominantes à l'aide d'un periodogramme et d'un visualiseur de specte. Ces fréquences nous permettront de déterminer la période r de notre effet saisonnier (de nos oscillations précédemment observées).

```
Input :
periodog(simul2)
```

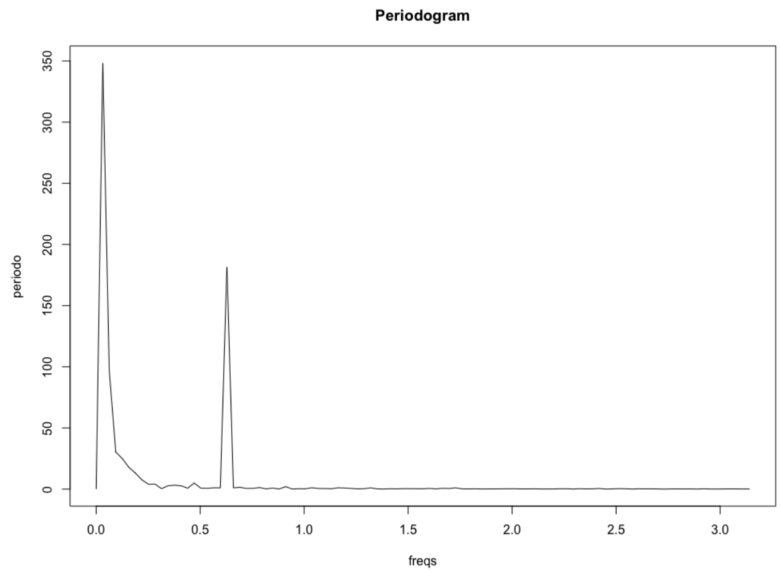

FIGURE 3 – Périodogramme de notre série simul2

Nous pouvons constater l'existence de pics dans notre périodogramme : le plus grand pic nous permet de déterminer la période r de notre saisonnalité. Nous trouverons que r = 10.

```
Input :
ind.pers <- which.max(spectrum(serie)$spec)
round(1/(spectrum(serie)$freq[ind.pers]) * frequency(serie))
Output :
[1] 10
```

Nous pouvons aussi visualiser cette fréquence prédominante sous forme de pic sur le spectre suivant :

```
Input :
spectrum(simul2)
```

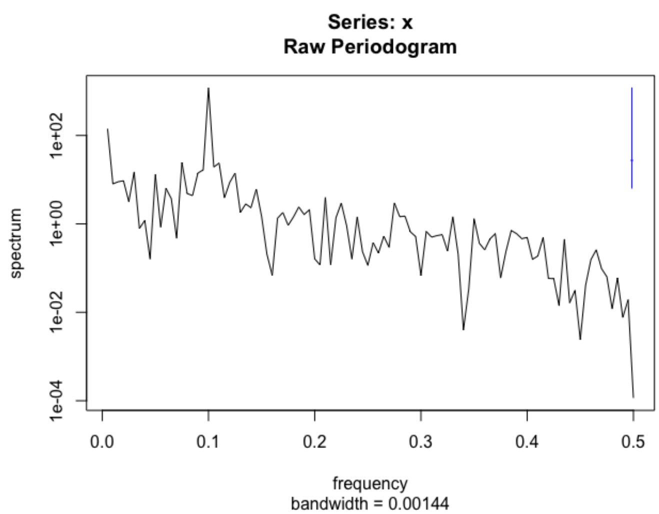

FIGURE 4 – Spectre de notre série simul2

Nous cherchons maintenant à valider notre période $r = 10$ via un test de saisonnalité. Nous allons tester les hypothèses suivantes :
H_0 : pas de saisonnalité de période r dans la série.
H_1 : présence de saisonnalité de période r dans la série.

```
Input :
serie = simul2
Saison.test(serie, 10)

Output :
Periode        d.f.       Tobs          p-valeur
    10.0000      9.0000    167.6291      0.0000
```

Nous avons une p-valeur de 0.0000. Nous rejetons donc H_0 au seuil de 5%, la saisonnalité $r = 10$ est validée.
Nous pouvons aussi visualiser l'allure de notre période :

```
Input :
seasonplot(serie, 10)
```

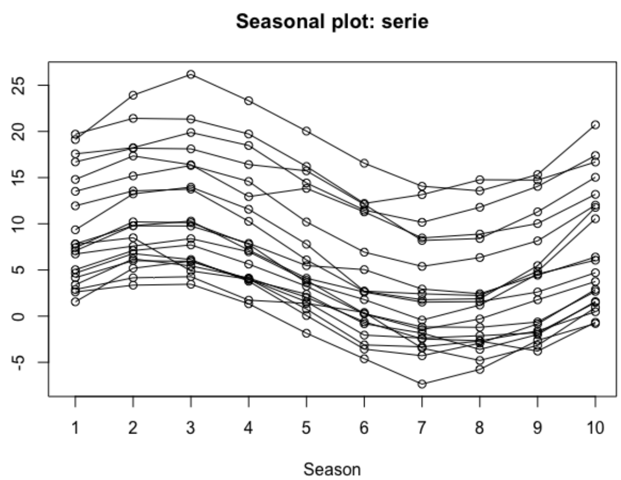

FIGURE 5 – Allure de la période

Nous constatons que l'allure de notre période est clairement sinusoïdale.

Nous cherchons maintenant à faire disparaitre cette composante saisonnière. Nous allons la différencier une fois, et notre futur modèle ARIMA sera un SARIMA de la forme suivante : $SARIMA(p, d, q)(0, 1, 0)[r = 10]$.

```
Input :
serie1 = diff(serie, lag = 10)
acf(serie1)
```

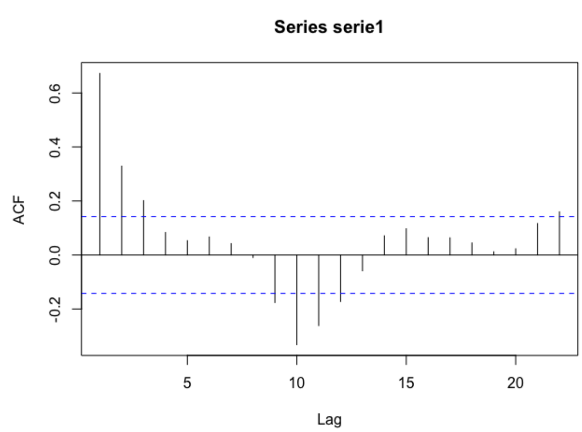

FIGURE 6 – ACF de notre série différenciée « serie1 »

Même si les décalages 9, 10, 11 et 12 rejettent H_0, la plupart de nos autocorrélations sont sous l'hypothèse H_0. La série n'a pas d'allure particulière donc nous pouvons dire que la périodicité a disparu.
Il ne nous reste plus qu'à effacer notre deuxième composante déterministe : la tendance.

4.3 Composante de tendance

Nous allons procéder encore une fois à un test KPSS de stationnarité de la série pour voir si la suppression de la saisonnalité a permis de stationnariser la série :

```
Input :
kpss.test(serie1)
Output :
KPSS Test for Level Stationarity

data:  serie1
KPSS Level = 0.68381, Truncation lag parameter = 4, p-value = 0.01502
```

Nous avons une p-valeur de 0.01502, nous rejetons donc toujours H_0 au seuil de 5%. La série n'est pas stationnaire.

Nous allons ensuite procéder à des tests de points montées et de discordances pour valider l'existence d'une tendance dans notre série. Les deux tests vont valider l'une des hypothèses suivantes :
H_0 : pas de tendance dans la série.
H_1 : présence de tendance dans la série.

```
Input :
PtMont.test(serie1)

Output :
          n          nM        stat    p-valeur
190.0000000  99.0000000   1.1279412   0.2593448
# Nous conservons H0 au seuil de 5%, avec une p-valeur de 0.2593448.

Input :
PtDisc.test(serie1)

Output :
           n           nD          stat     p-valeur
1.900000e+02 7.176000e+03 4.111142e+00 3.937075e-05

# Nous rejetons ici H0.
```

Nous avons deux résultats contradictoires : un test affirme l'inexistence d'une tendance, mais un autre affirme le contraire. Nous pouvons affirmer l'existence d'une tendance dans la série, à partir du moment où un test a affirmé l'existence de cette composante. Nous allons donc différencier notre série jusqu'à ce que la tendance disparaisse :

```
Input :
new_serie1 = diff(serie1)
PtMont.test(new_serie1)

Output :
           n           nM         stat     p-valeur
189.00000000 101.00000000   1.75918641   0.07854585

# Nous conservons ici H0 avec une p-valeur > 5%.
La série n'a pas de tendance.

Input :
PtDisc.test(new_serie1)

Output :
           n           nD         stat     p-valeur
 189.0000000 8806.0000000    0.1771119    0.8594205

# Nous conservons aussi ici H0, avec une p-valeur > 5%.
La série n'a pas de tendance.
```

Au vu de nos deux résultats précédents, nous pouvons affirmer qu'après avoir différencié une fois notre série (donc $d = 1$), sa tendance a disparu.

Nous allons tester une nouvelle fois la stationnarité de la série avec le test de KPSS :

```
Input :
kpss.test(new_serie1)

Output :
KPSS Test for Level Stationarity

data:  new_serie1
KPSS Level = 0.021341, Truncation lag parameter = 4, p-value = 0.1
```

Nous avons trouvé une p-valeur de 0.1, nous conservons au seuil de 5% notre hypothèse H_0 de stationnarité de la série.
Après avoir éliminer la composante saisonnière stationnarisé la série avec une différenciation $d = 1$, nous allons déterminer les ordres p et q de notre ARMA avec une EACF sur la série différenciée « new_serie1 ».

```
Input :
```

```
eacf(new_serie1)

Output :
AR/MA
  0 1 2 3 4 5 6 7 8 9 10 11 12 13
0 o x o x o o o x o x o  o  o  x
1 o x o o o o o x o x x  o  o  x
2 o x o x o o o o o x o  o  o  x
3 o x o x o o o o o x o  o  o  o
4 x x o x o x o o o x o  o  o  o
5 x x o x o x o o o x o  o  o  o
6 x x o o o x o o o x o  o  o  o
7 x x o o o x o o o x o  x  o  o
```

Notre EACF est une matrice qui contient les nombres de décalages potentiellement significatifs en colonne pour le processus MA d'ordre q et en ligne pour le processus AR d'ordre p. Un « o » est un croisement p/q significatif de notre ACF et de notre PACF. « x » est un croisement non-significatif. Dans cette matrice nous cherchons à trouver un croisement optimal, en coin de « x », qui respecte le fait que tous les déterminants soient signifcatifs du côté Est-Sud-Est de la matrice. Pour p = 3 et q = 12, nous avons un croisement qui n'est malheureusement pas en coin (il n'y en a pas dans cette matrice), mais qui respecte le fait qu'il y ait des déterminants significatifs à droite et en bas.
Nous pourrons donc estimer la série avec un modèle $SARIMA(p, d, q)(0, 1, 0)[r]$ $= SARIMA(3, 1, 12)(0, 1, 0)[10]$

4.4 Ajustement du premier modèle

Dans l'estimation du modèle, nous cherchons à savoir si nos résidus forment un bruit blanc, qu'il soit faible ou non. Nous allons donc procéder à une première modélisation SARIMA qui aura la forme suivante : $SARIMA(3, 1, 12)(0, 1, 0)[10]$

```
(sarima1 = Arima(serie,
                 order = c(3,1,12),
                 seasonal = list(order=c(0,1,0),
                                 period=10),
                 method="ML"))

Output :
Series: serie
ARIMA(3,1,12)(0,1,0)[10]

Coefficients:
         ar1     ar2      ar3      ma1      ma2     ma3     ma4      ma5      ma6
      0.0602  0.2617  -0.0711  -0.0647  -0.8649  0.0343  0.0155  -0.0053  -0.0157
s.e.  0.1427  0.1442   0.1163   0.1452   0.1423  0.0939  0.0874   0.1210   0.0537
```

```
        ma7      ma8      ma9     ma10    ma11    ma12
    -0.0274   0.0539  -0.0091  -0.9973  0.1043  0.8407
s.e.  0.1173   0.0943   0.0880   0.1194  0.1270  0.1351

sigma^2 estimated as 1.025:  log likelihood=-277.28
AIC=586.56   AICc=589.72   BIC=638.43
```

Notre modèle nous sort donc 15 coefficients estimés par la méthode ML, qui consiste à maximiser la fonction log-vraisemblance du modèle SARIMA. Le modèle produit 3 critères : AIC, AICc et BIC. Ces derniers permettent d'évaluer la qualité du modèle. Nous devons selectionner le modèle ayant le critère le plus faible possible. Nous allons nous concentrer sur le criètre d'information d'Akaike (AIC), il est de 586.56.
Nous allons ensuite visualiser l'état de nos résidus :

```
Input :
tsdiag(sarima1)
```

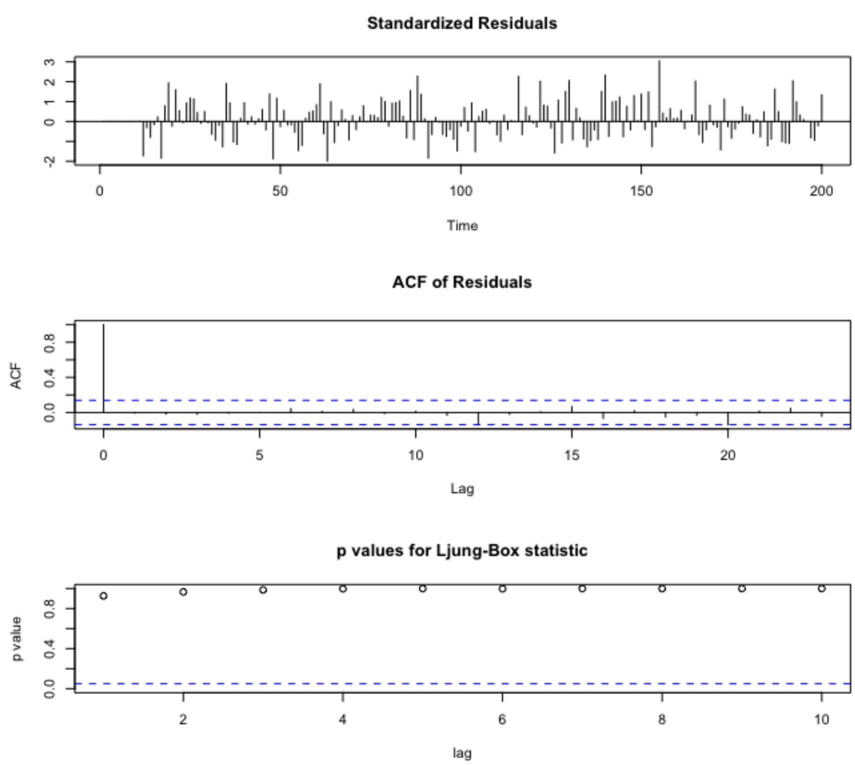

FIGURE 7 – Spectre des résidus standardisés, ACF des résidus, p-valeur du test d'indépendance des observations de Ljung-Box

Dans l'ACF et le test de Ljung-Box, nous constatons que les résidus sont iid. Ces derniers forment donc un bruit blanc.

Voici notre modèle ajusté (en rouge) à notre simulation (en noir) :

```
Input :
plot(serie, type = "l", ylab = " ")+
  lines(sarima1$fitted, type = "l", ylab = " ", col = "red")
```

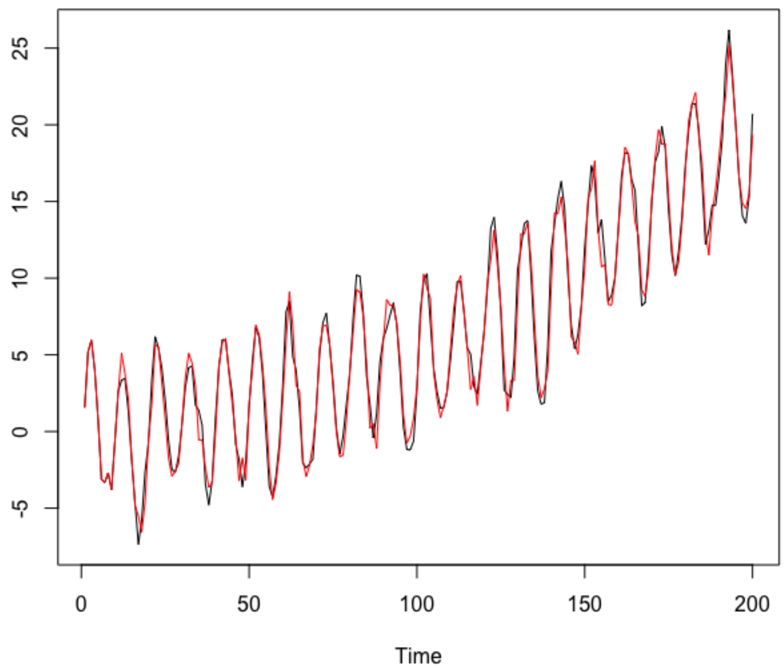

FIGURE 8 – Modèle sarima1 (en rouge) ajusté à notre simulation (série « serie »)

Est-il possible de simplifier notre premier modèle ? Nous allons tester la significativité de nos coefficients estimés précédemment, avec un test de Student. Nous allons chercher un modèle optimal qui contient uniquement des coefficients significatifs.

```
Input :
coefs = sarima1$coef
Npar = length(coefs)
ect = sqrt(diag(sarima1$var.coef))[-Npar]
```

```
test = coefs/ect
abs(test) <= 1.96

Output :
  ar1   ar2   ar3   ma1   ma2   ma3   ma4   ma5   ma6   ma7   ma8   ma9  ma10  ma11
 TRUE FALSE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE
 ma12
FALSE
```

Les coefficients significatifs ont pour valeur FALSE, puisqu'ils dépassent la borne de 1.96 de la loi normale avec un seuil 95%. Nous allons retirer du modèle les coefficients non significatifs, ce qui nous donne un modèle consitué de MA $=$ q $=$ 12, donc notre modèle simplifié sera de la forme $SARIMA(0,1,12)(0,1,0)[10]$.

4.5 Ajustement du modèle simplifié $SARIMA(0,1,12)(0,1,0)[10]$

Estimons les coefficients de notre deuxième modèle, puis testons leur significativité :

```
Input :
(sarima2 = Arima(serie,
                 order = c(0,1,12),
                 seasonal = list(order=c(0,1,0),
                                 period=10),
                 method="ML"))

Output :
Series: serie
ARIMA(0,1,12)(0,1,0)[10]

Coefficients:
          ma1      ma2     ma3      ma4     ma5      ma6      ma7     ma8      ma9
      -0.0567  -0.6853  0.0157  -0.0054  0.0074  -0.0090  -0.0149  0.0240  -0.0290
s.e.   0.0809   0.0927  0.0703   0.0709  0.0804   0.0556   0.0750  0.0706   0.0718
         ma10     ma11    ma12
      -0.9745   0.1147  0.6874
s.e.   0.0798   0.0879  0.0943

sigma^2 estimated as 1.056:  log likelihood=-280.81
AIC=587.61   AICc=589.69   BIC=629.75
```

Avec ce modèle simplifié nous avons un AIC qui a augmenté de peu, il est de 587.61. Mais les AICc et les BIC sont plus faibles. Nous pouvons donc être tenté de garder ce modèle là.

```
tsdiag(sarima2)
```

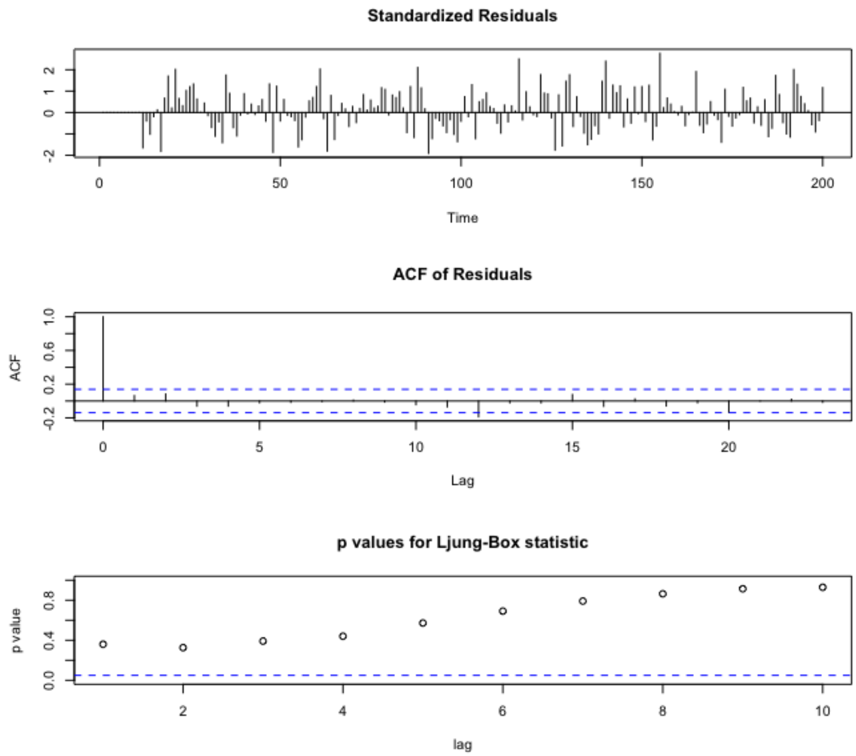

FIGURE 9 – Spectre des résidus standardisés, ACF des résidus, p-valeur du test d'indépendance des observations de Ljung-Box du modèle simplifié « sarima2 »

À travers ces trois graphiques, nous constatons que le modèle a des résidus iid. Les résidus forment donc un bruit blanc.

Ajustement du modèle aux données si mulées :

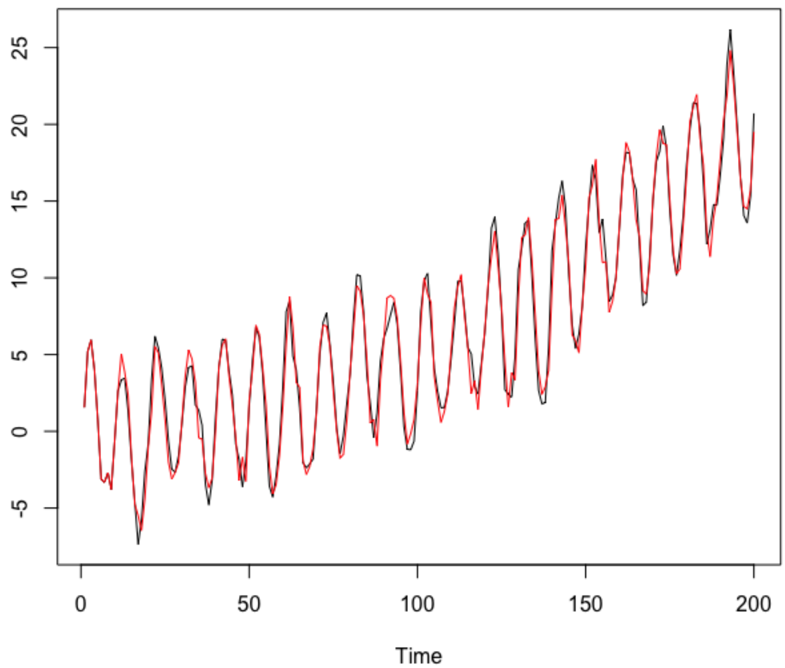

FIGURE 10 – Modèle simplifié sarima2 (en rouge) ajusté à notre simulation (série « serie »)

Notre modèle ajuste tout aussi bien nos données simulées.

Nos deux modèles sont-ils équivalents ? Dans le cadre de cette modélisation, nous souhaiterions garder le modèle le moins complexe possible, donc conserver notre modèle simplifier. Mais pour le conserver, nous devons d'abord vérifier l'équivalence de nos deux modèles avec un test du rapport de vraisemblance. Étant donné un modèle initial et un modèle simplifié emboités, ce test nous permettra de savoir si le modèle simplifié avec un certain nombre de paramètres suffit à représenter à bien représenter les données, ou s'il est nécessaire conserver les paramètres du modèle initial.

Nous allons donc tester les hypothèses suivantes :
H_0 : nos deux modèles sont équivalents, auquel cas nous choisissons de garder le modèle simplifié.
H_1 : nos deux modèles ne sont pas équivalents, nous gardons le modèle initial.

```
Input :
MI = sarima1 # Modèle initial
MS = sarima2 # Modèle simplifié
nS = length(MS$coef) # nb coefs du modèle simplifié
nI = length(MI$coef) # nb coefs du modèle initial

# Test du rapport de vraisemblance :
Tobs = -2 * (MS$loglik - MI$loglik)
# p-valeur :
(pval = 1 - pchisq(Tobs, df = nI - nS))

Output :
0.07032301
```

Au seuil de 5%, nous conservons H_0 puisque nous avons une valeur critique supérieure à 0.05. Nos deux modèles sont équivalents et nous sélectionnons le modèle simplifié.

Nous pouvons maintenant nous interroger sur la significativité de nos coefficients, pour savoir s'il est possible de simplifier notre deuxième modèle. Nous allons utiliser un test de Student :

```
Input :
coefs2 = sarima2$coef
Npar = length(coefs2)
ect = sqrt(diag(sarima2$var.coef))[-Npar]
test2 = coefs2/ect
abs(test2) <= 1.96

Output :
  ma1   ma2   ma3   ma4   ma5   ma6   ma7   ma8   ma9  ma10  ma11  ma12
 TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE FALSE
```

Il n'est pas possible de simplifier le modèle une nouvelle fois. Effectivement, le dernier coefficients est un coefficient significatif. Si nous le retirons, les performances de notre modèle pourraient être grandement impactées.

4.6 Ajustement d'un troisième modèle : estimation des paramètres par la fonction « auto.sarima »

Nous allons essayer de comparer les performances d'un modèle estimé par l'algorithme d'estimation automatique « auto.sarima » afin de déterminer si la machine aurait pu estimer un meilleur modèle.

```
Input :
(sarima3 = auto.arima(serie, ic = "aic"))
Output :
Series: serie
ARIMA(4,1,1) with drift

Coefficients:
         ar1      ar2     ar3      ar4      ma1   drift
      1.0219  -0.5845  0.2646  -0.4941  -0.8052  0.1034
s.e.  0.0661   0.0949  0.0940   0.0648   0.0421  0.0204

sigma^2 estimated as 1.334:  log likelihood=-310.51
AIC=635.01   AICc=635.6   BIC=658.06
```

L'algorithme a estimé 6 coefficients pour ce modèle : 4 coefficients pour le processus AR, 1 seul pour le MA, et il a ajouté du drift. Il a également différencié 1 fois pour éliminer la tendance de la série.
L'AIC est de 635.01 : la valeur est plus élevée que notre celle du modèle sarima2. Nous pourrions nous interroger sur le fait que notre modèle n'ait pas introduit de composante saisonnière dans son estimation, alors que l'on avait identifié des oscillations dans l'acf de notre simulation. Enfin, du drift a été introduit par notre estimation automatique (la moyenne n'est pas nulle).

Observons l'état de nos résidus :

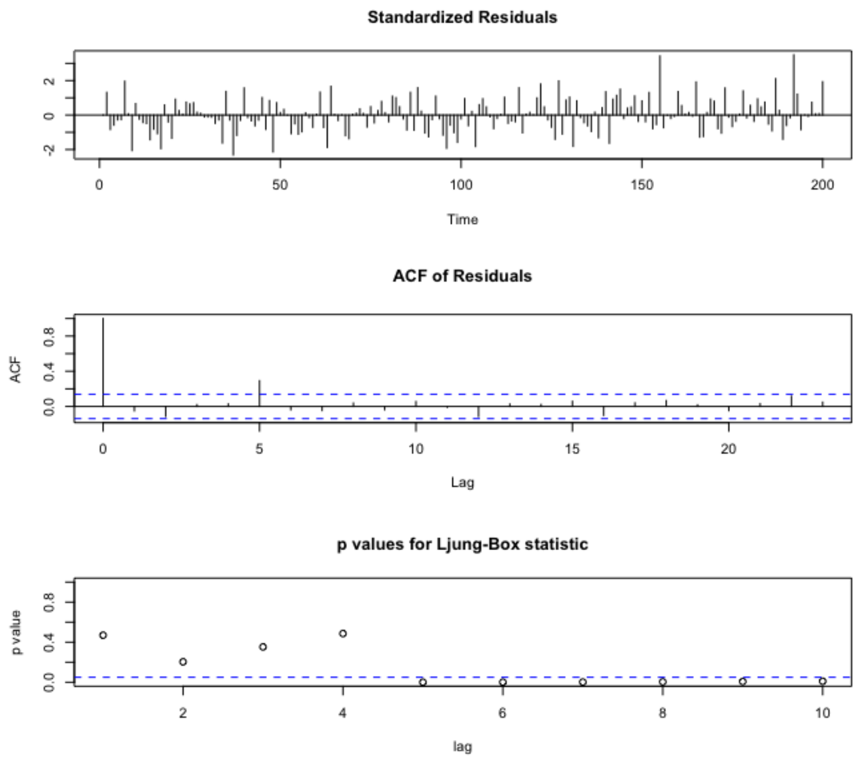

FIGURE 11 – Spectre des résidus standardisés, ACF des résidus, p-valeur du test d'indépendance des observations de Ljung-Box du modèle simplifié « sarima3 »

Nous pouvons constater que nos résidus ne forment pas particulièrement un bruit blanc : les observations sont dépendantes selon le test de Ljung-Box (6 lags sur 10 rejettent H_0) et le décalage numéro 5 de l'ACF est significatif.
Théoriquement le modèle est moins viable que les précédents, nous allons donc l'écarter de notre analyse.
Mais attention, cela ne signifie pas que l'estimation automatique est un algorithme non-pertinent. Cela signifie simplement qu'avec les paramètres donnés dans la fonction « auto.sarima() », l'algorithme n'est pas efficace. En changeant certains paramètres de la fonction, l'algorithme pourrait être plus pertinent.

5 Prévisions du modèle $SARIMA(0,1,12)(0,1,0)[10]$

Nous allons estimer les périodes futures n = 201 et 202 à l'aide de notre modèle simplifié. Voici les résultats de nos prévisions :

```
Input :
temps = time(serie)
(previsions = forecast(sarima2, h = 10, conf = 80))

Output :
    Point Forecast    Lo 80    Hi 80    Lo 95    Hi 95
201       23.99097 22.65022 25.33172 21.94047 26.04147
202       25.13331 23.28907 26.97755 22.31278 27.95383
203       25.53053 23.65570 27.40536 22.66323 28.39784
204       23.49590 21.58919 25.40260 20.57984 26.41195
205       20.82019 18.88160 22.75878 17.85537 23.78501
206       17.82039 15.85072 19.79007 14.80804 20.83275
207       16.17156 14.17293 18.17019 13.11492 19.22820
208       16.57930 14.55500 18.60361 13.48339 19.67521
209       18.18360 16.13053 20.23666 15.04371 21.32348
210       21.76027 19.68327 23.83726 18.58378 24.93676
```

Pour l'observation n = 201, nous avons estimé une valeur future de 23.9907. Et pour l'observation n = 202, notre valeur future estimée est de 25.13331.
Au seuil de 80%, nos prévisions se situent entre les bornes inférieures et supérieures de notre intervalle de confiance, et sont donc correctement estimées statistiquement.

Voici l'état de nos prévisions sur une période de 10 observations :

```
Input :
plot(previsions,
     plot.conf = T,
     shaded = T,
     col = "black",
     fcol = "blue")
segments(temps[200],
         serie[200],
         temps[200] + 1/frequency(serie),
         previsions$mean[1],
         col = "blue")
```

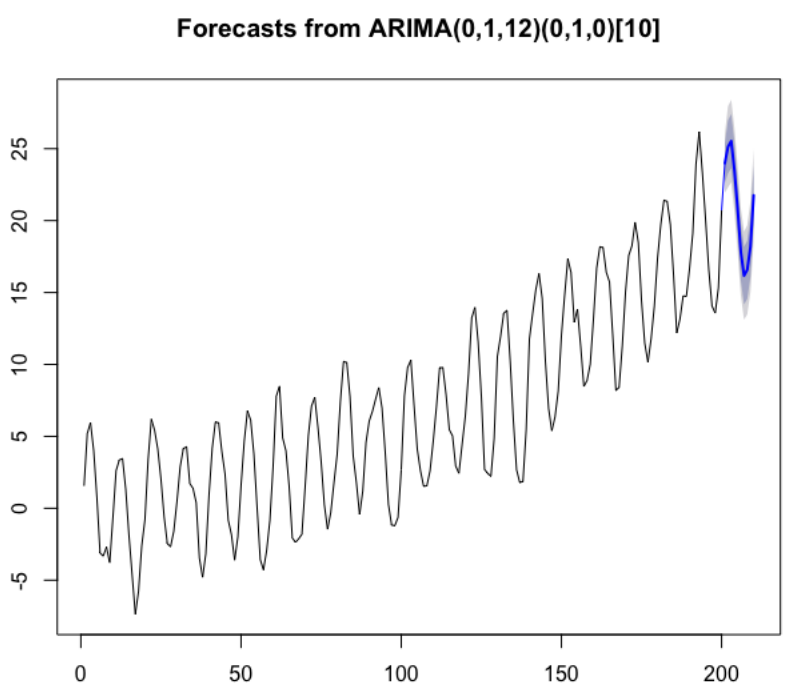

FIGURE 12 – Prévisions de 10 observations (en bleu) de notre modèle $SARIMA(0,1,12)(0,1,0)[10]$

Notre oscillation estimée de couleur bleue semble avoir une forme qui s'inscrit dans la continuité de notre série.

6 Synthèse

Après avoir reproduit le processus stationnaire présenté dans la deuxième section du rapport à l'aide d'un modèle $SARIMA(0, 1, 12)(0, 1, 0)[10]$, nous pouvons déjà constater que le modèle le plus pertinent pour notre estimation ne ressemble pas à celui de notre simulation.
Au cours de notre analyse, nous avons listé plusieurs estimations possibles de modèles SARIMA : notre première estimation $SARIMA(3, 1, 12)(0, 1, 0)[10]$ n'était finalement pas la plus optimale pour prévoir. Effectivement, le modèle était plus complexe que le modèle simplifié puisqu'il disposait de 15 paramètres. Même s'il disposait d'un AIC plus faible, les autres critères (AICc, BIC) étaient plus élevés. L'estimation de nos premiers modèles a été effectué avec la stratégie de maximisation de la vraisemblance de nos paramètres : on essaie d'estimer les paramètres optimaux qui vont maximiser la vraisemblance de nos données estimées. Il est fortement possible qu'en changeant la stratégie d'estimation de nos paramètres, en utilisant la minimisation de la somme des carrés résiduels par exemple, que les résultats de nos critères d'évaluations soient différents. Pour conclure, nous aurions pu aller plus loin dans notre analyse. Effectivement, exploiter d'autres modèles de séries temporelles et comparer leurs performances avec notre modèle simplifié était une piste pertinente pour estimer encore mieux des observations, mais nous aurions pu également tenter de changer les paramètres de la fonction d'estimation automatique du modèle sarima (sarima.auto), utiliser d'autres stratégies pour estimer nos paramètres comme un mélange entre la minimisation de la somme des carrés résiduels et la maximisation de la fonction de vraisemblance du modèle ARIMA, par exemple.