Projet Séries Temporelles

Idir SADAOUI

Dans ce document, nous allons analyser une série temporelle grâce aux outils vus en cours, nous n'aurons besoin que du package `TimeSeries` pour cette analyse.
On commence par importer les données qui m'ont été attribué à savoir la série numéro 19.

Analyse descriptive de la série temporelle

```
donnees <- scan('serie_19.dat')
length(donnees)
```

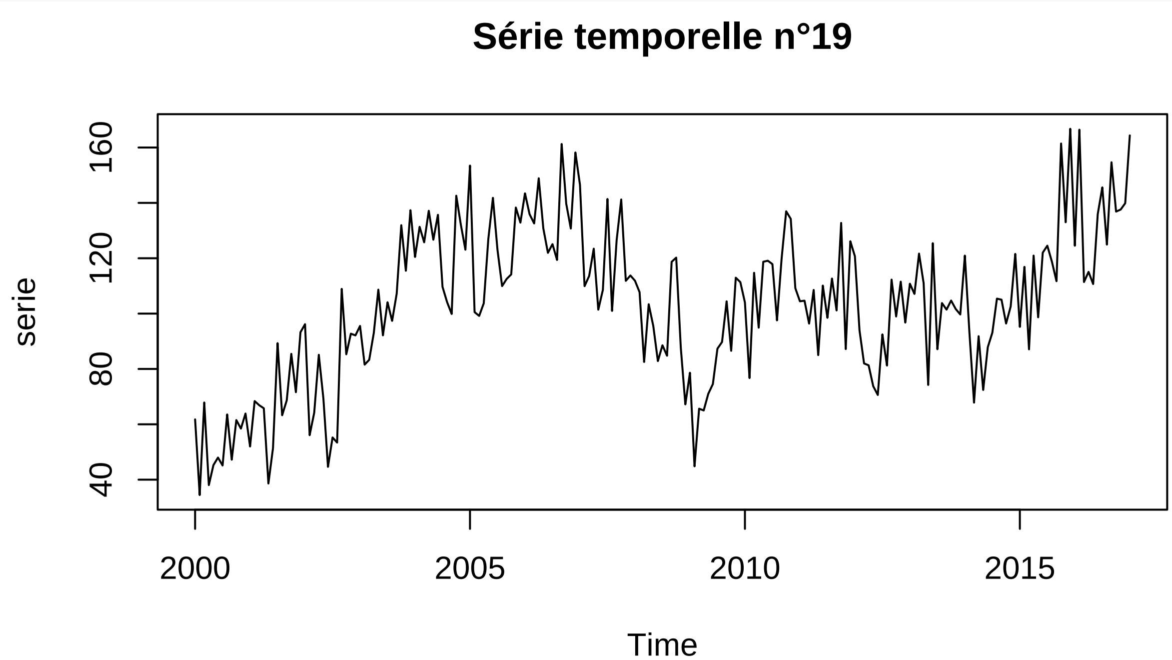
```
## [1] 205
```

On voit qu'il y a 205 valeurs dans le fichier de données, on décide de transformer ces données en série temporelle avec une fréquence de $P = 12$ mois et une date de début fixée au 1 Janvier 2000.
Pour cela on utilise la commande `ts`

```
serie  <- ts(donnees,frequency = 12,start = c(2000,1))
```

On affiche maintenant le graphique correspondant à cette série temporelle avec la commande `ts.plot`

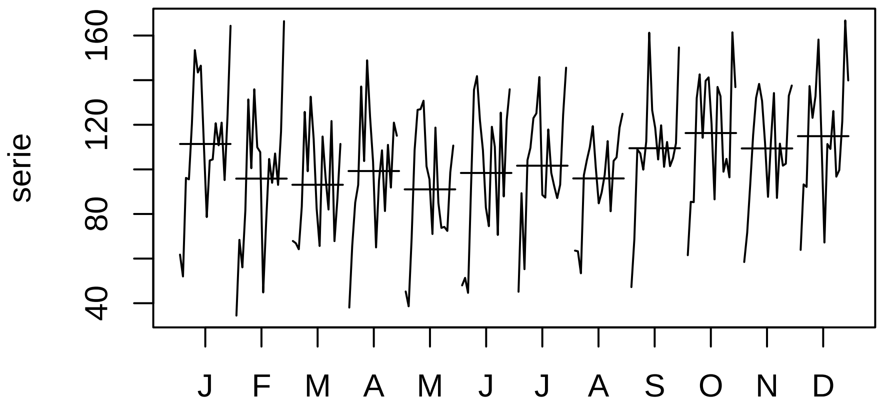
```
ts.plot(serie,main = 'Série temporelle n°19')
```



À première vue, notre série temporelle ne semble pas être stationnaire, elle semble avoir une tendance et une saisonnalité.

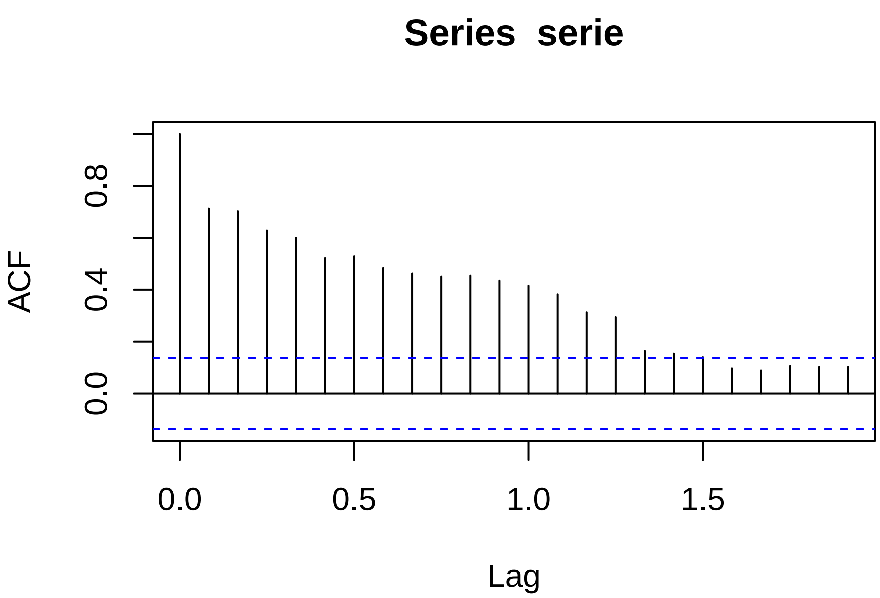
On peut commencer par regarder le `monthplot` de notre série.

```
monthplot(serie)
```



On voit ici l'évolution d'année en année pour un mois donnée et on remarque une tendance croissante.
On peut aussi regarder la représentation graphique de la fonction d'autocorrélation ρ grâce à la commande `acf`.

```
acf(serie)
```

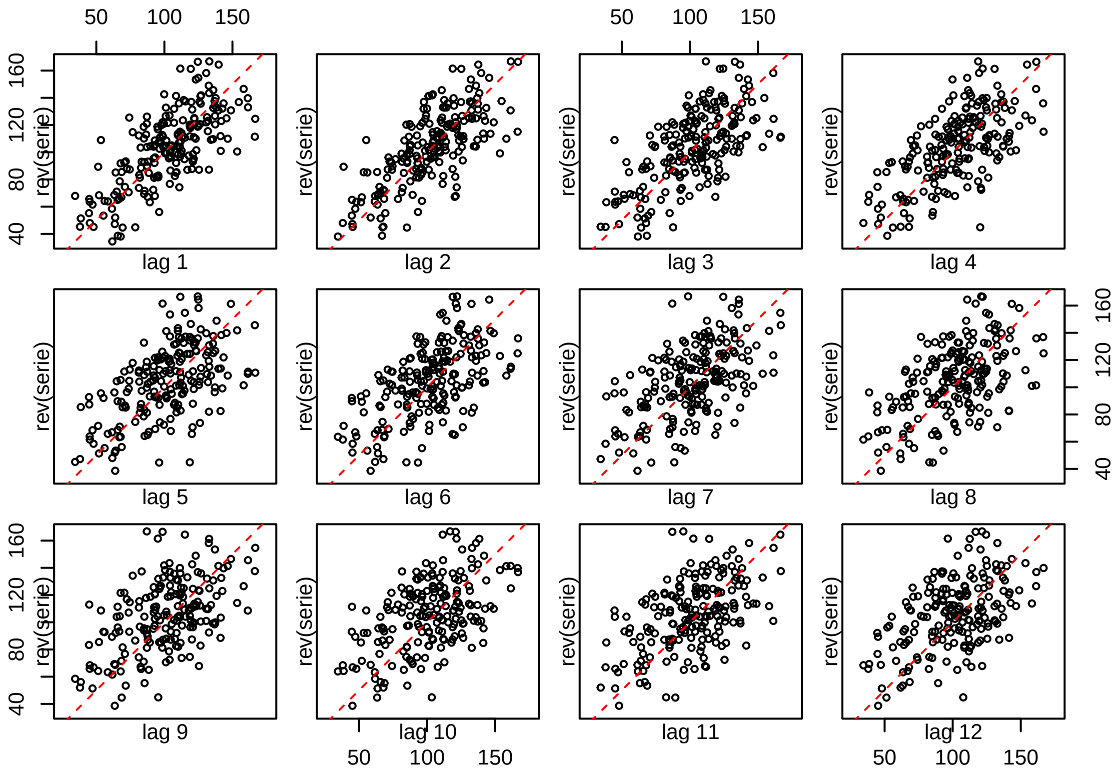


On sait qu'une décroissance rapide de l'ACF est la signature d'une série stationnaire, dans notre cas, il y a bien une décroissance mais elle n'est pas exponentielle malgré le fait que les corrélations finissent par être significativement nulles vers la fin, on en conclut que la série n'est pas stationnaire.

ps : grâce à la commande `adf.test` du package `tseries` on peut faire un test sur la stationnarité d'une série temporelle, dans notre cas on rejette l'hypothèse de stationnarité ce qui confirme nos intuitions.
On ne détaillera pas ce test dans ce document.

On peut aussi regarder le `lag.plot` de notre série.

```
lag.plot(rev(serie),12,layout = c(3,4), diag.col = 'red')
```



Le lagplot est un diagramme de dispersion des points qui a pour abscisse la série retardée de 12 instants et pour ordonnée la série non retardée, par ailleurs on inverse notre série temporelle pour faire en sorte de voir la dépendance de la série par rapport à son passé.
C'est pour cela que l'on utilise la commande `rev`.
On voit ici que les auto-corrélations sont positives et sont fortes ce qui est en adéquation avec l'ACF vue au dessus.

Elimination jointe de la tendance et de la saisonnalité

On va maintenant appliquer la méthode d'élimination jointe de la tendance et de la saisonnalité vue en cours à notre série temporelle.

Premièrement on décompose la tendance :
Soit X_t notre série temporelle alors on a la série corrigée de la tendance qui est définit de la façon suivante :

$$\tilde{S}_t := X_t - X_t^* \text{ , avec } t = m+1, ..., PN - m$$

Dans notre cas la période est paire (=12), alors pour $m = 6$ on applique la moyenne mobile suivante pour construire la série corrigée de la tendance :

$$X_t^* = \frac{1}{12}\left(\frac{1}{2}X_{t-6} + X_{t-5} + ... + X_t + X_{t+1} + ... + X_{t+5} + \frac{1}{2}X_{t+6}\right)$$

Finalement, on a :

```
q = 12/2
l=length(serie)
tend=rep(NA,l)
for(t in (1+q):(l-q)){
  tend[t]=1/12*(0.5*serie[t-q]+sum(serie[(t-q+1):(t+q-1)])+0.5*serie[t+q])
}
tend=ts(tend, frequency = 12,start = c(2000,1),c(2017,1))
SerieSansTend = serie - tend
```

Deuxièmement on décompose la saisonnalité :

Pour estimer la saisonnalité on reprend la série corrigée de la tendance $\tilde{S}_t$ et on réindexe les valeurs de $\tilde{S}_t$ saison par saison.
Ainsi on a

$$\tilde{S}_t = \tilde{S}_{j+(i-1)P} \text{ avec } j = 1, ..., P \text{ et } i = 1, ..., N$$

L'estimateur des coefficients saisonniers est donné par

$$\tilde{s_j} = \frac{1}{N}\sum_{i=1}^{N}\tilde{S}_{j+(i-1)P} \quad , \quad m+1 \leq j \leq P - m$$

et enfin on a l'estimateur de la saisonnalité qui est donné par

$$\hat{S}_j = \tilde{s_j} - \frac{1}{P}\sum_{l=1}^{P}\tilde{s_j} \quad , \quad j = 1, ..., P$$

Finalement, on a :

```
W = rep(NA,12)
p = floor(l/2)
for(k in 1:12){
  W[k] = mean(SerieSansTend[k+((1:p)-1)*12], na.rm=TRUE)
  }
sais = W - mean(W)
sais = c(rep(sais,p), sais[1])
sais = ts(sais, frequency=12, start = c(2000,1), c(2017,1))
SerieSansSais = serie - sais
```

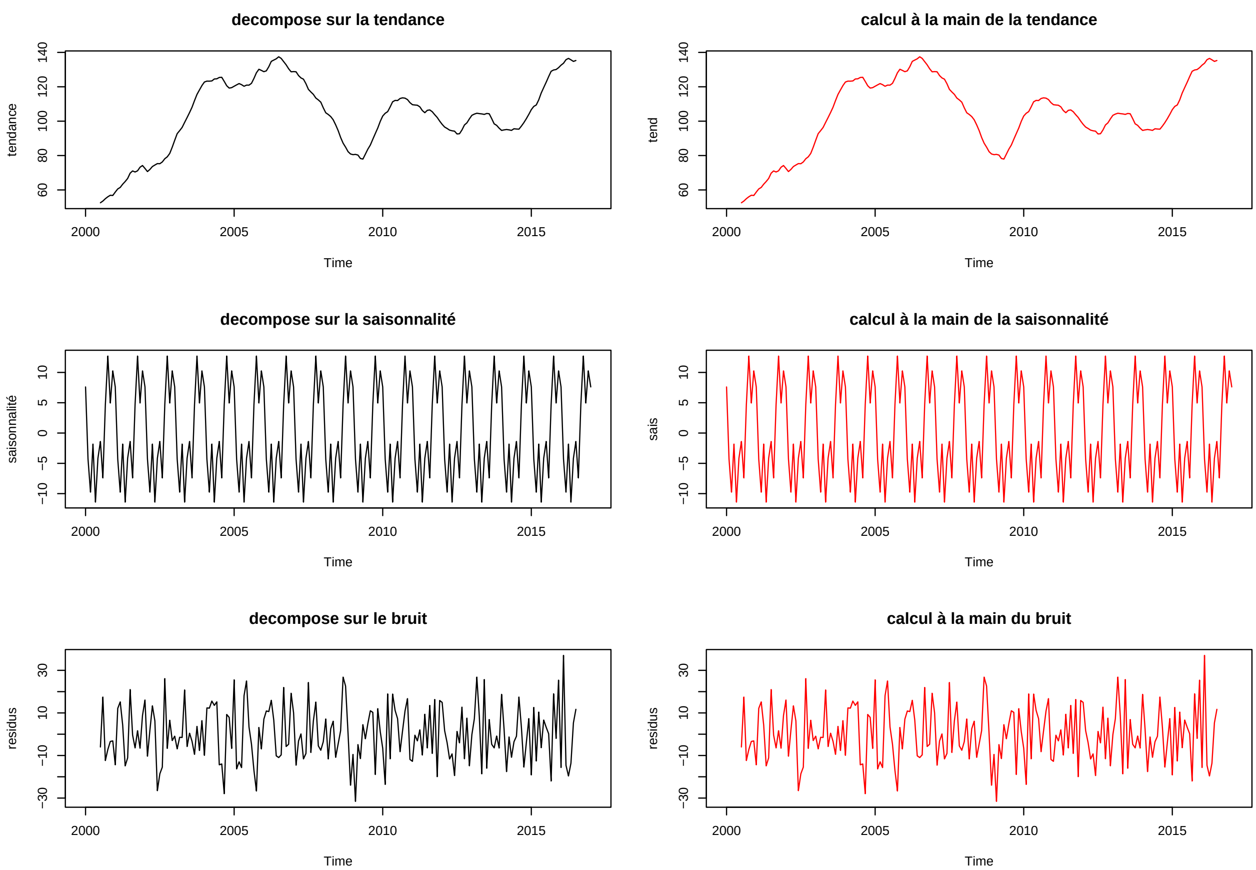
Dernièrement on décompose le bruit :

On estime le résidu en lui retranchant les estimations de la tendance et de la saisonnalité trouvées précédemment, on a :

```
residus <- serie - sais - tend
```

On va maintenant comparer nos estimations avec ceux de la commande `decompose` et afficher les graphiques côte à côte pour chaque estimations.

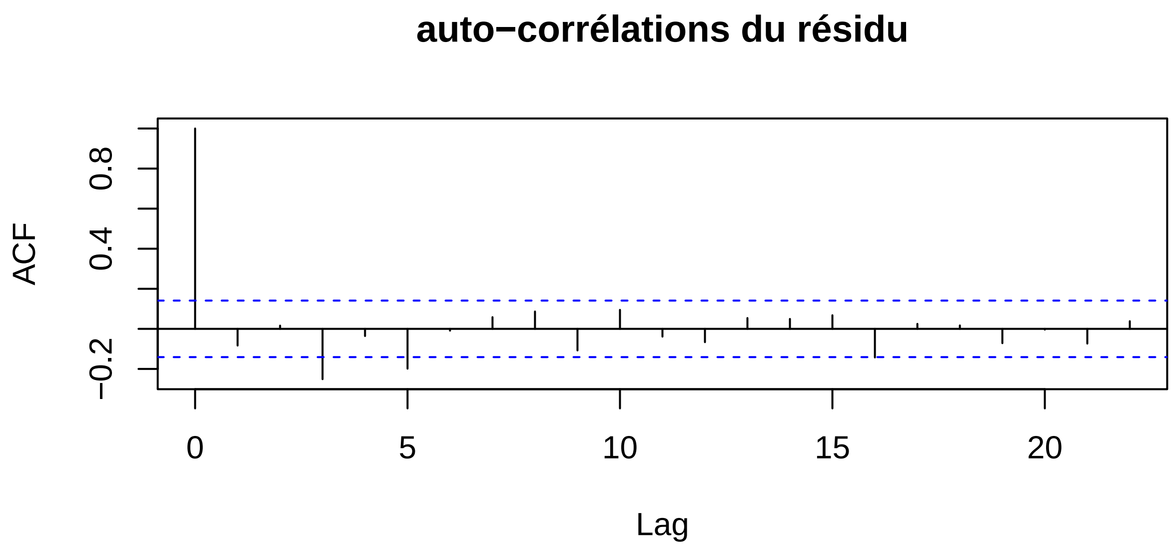
```
par(mfrow=c(3,2))
plot(decompose(serie)$trend,type='l',main='decompose sur la tendance',ylab='tendance')
plot(tend,type='l',col='red',main='calcul à la main de la tendance')
plot(decompose(serie)$seasonal,type='l',main='decompose sur la saisonnalité',ylab='saisonnalité')
plot(sais,type='l',col='red',main='calcul à la main de la saisonnalité')
plot(decompose(serie)$random,type='l',main='decompose sur le bruit',ylab='residus')
plot(residus,type='l',col='red',main='calcul à la main du bruit')
```



On trouve des résultats quasiment identiques à la fonction `decompose`, c'est un bon indicateur de la véracité de nos calculs.

La dernière chose à vérifier dans cette partie est que le résidu doit être la réalisation d'un processus stationnaire, pour cela il faut d'abord supprimer les valeurs manquantes de notre résidu estimé puis tracer sa fonction d'autocorrélation.

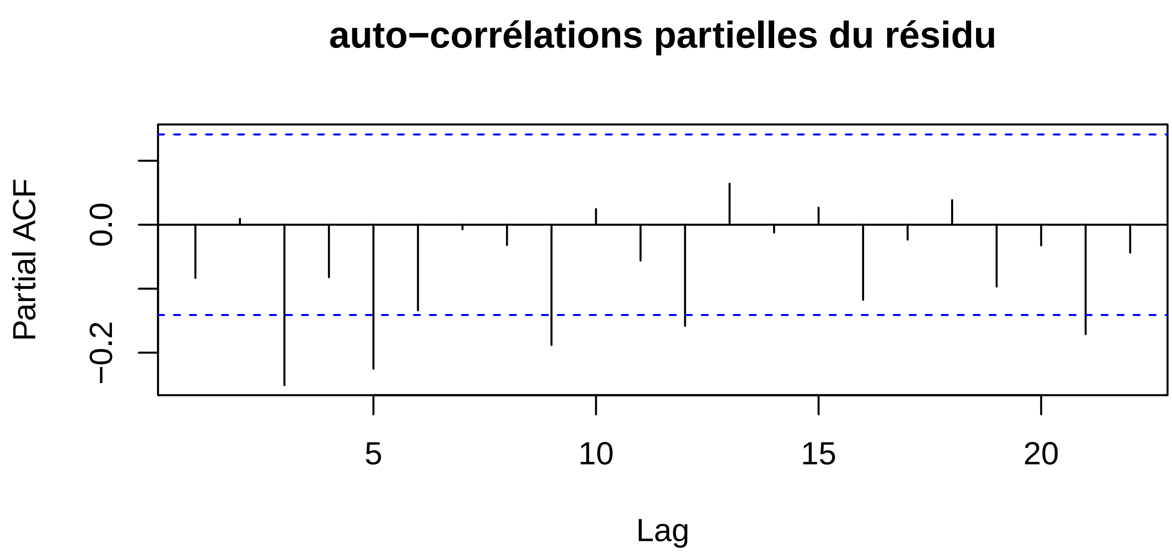
```
residus <- residus[-which(is.na(residus))]
acf(residus, main='auto-corrélations du résidu')
```



On voit bien que le résidu est stationnaire de par sa décroissance exponentielle, il ne reste aucune trace de la tendance et de la saisonnalité comme attendu.

On en profite pour tracer la fonction d'autocorrélation partielle τ (PACF) du résidu.

```
pacf(residus, main='auto-corrélations partielles du résidu')
```



Estimations de modèles et prévisions

On va maintenant proposer et estimer des modèles ARMA pour le résidu.
Premièrement on va couper notre résidu en deux grâce à la commande `window`, la première partie ira du 1 Janvier 2000 au 1 Février 2016 et la deuxième partie du 1 Mars 2016 jusqu'au 1 Juillet 2017.

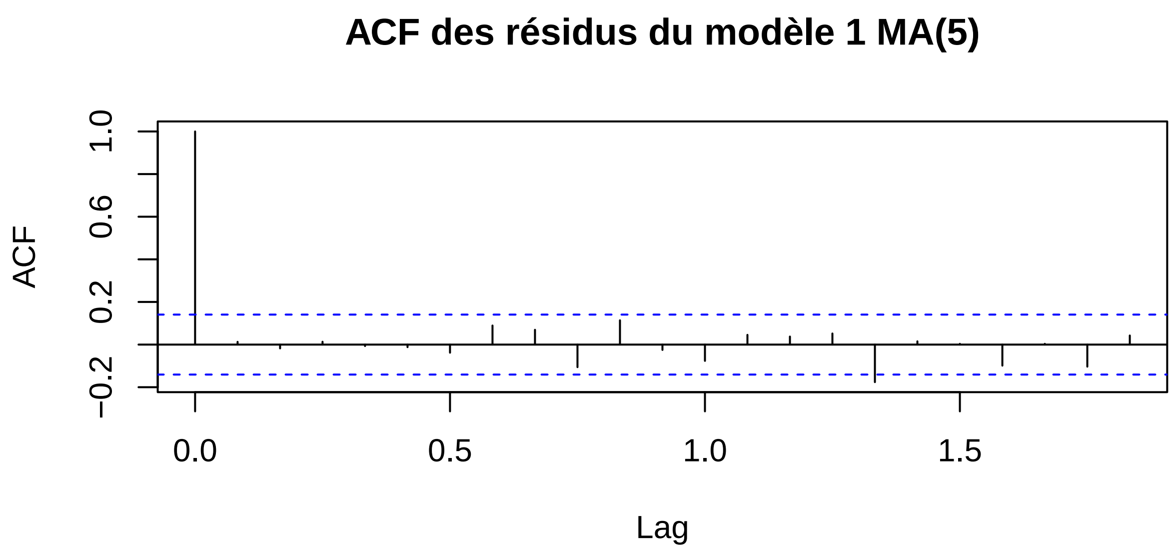
```
res <- ts(residus,frequency=12,start = c(2000,1),c(2016,7))
res_debut <- window(res,start=c(2000,1),end=c(2016,2))
res_vrai <- window(res,start=c(2016,3))
```

L'idée est de prédire les données de la deuxième partie grâce à ceux de la première partie.

D'après des résultats du cours, on sait d'une part qu'il n'est pas déraisonnable de proposer un modèle $MA(q)$ lorsque $q+1$ est le plus petit indice h pour lequel l'ACF $\rho(h) = 0$ et d'autre part qu'il n'est pas déraisonnable de proposer un modèle $AR(p)$ lorsque $p+1$ est le plus petit indice h pour lequel la PACF $\tau(h) = 0$.
Dans notre cas, on voit que l'ACF s'annule à partir de $h = 6$ on propose donc un modèle $MA(5)$, de plus on voit que la PACF s'annule à partir de $h = 6$ on propose donc un modèle $AR(5)$.
On va maintenant estimer les paramètres de ces deux modèles et décider sur la blancheur des résidus.

Modèle MA(5) :

```
modele1 <- arima(res_debut,order=c(0,0,5))
acf(modele1$residuals, main = 'ACF des résidus du modèle 1 MA(5)')
```



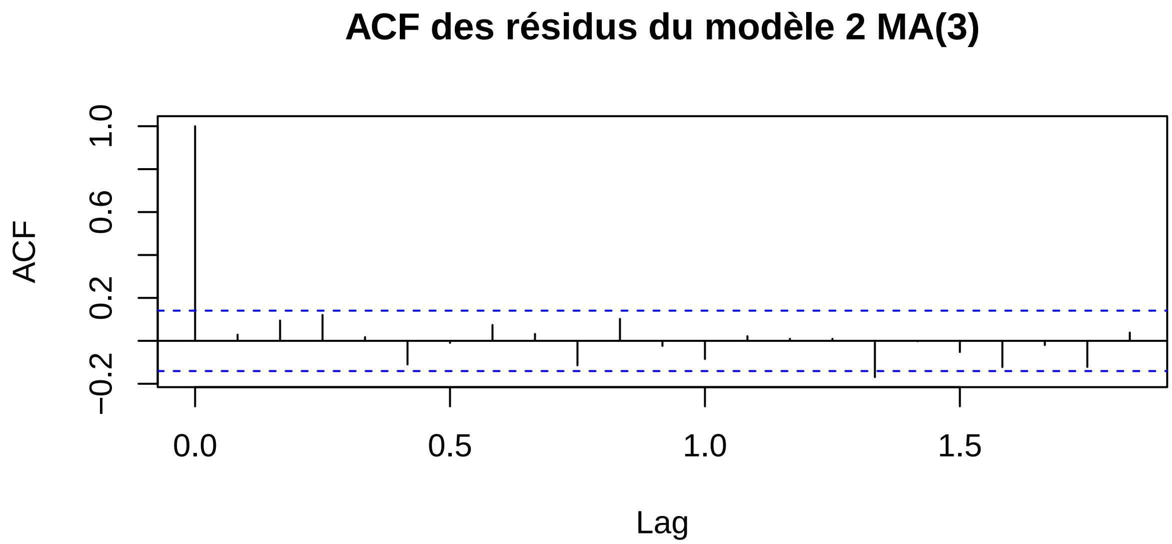
On décide d'accepter la blancheur des résidus car l'intervalle de confiance contient toutes les valeurs de ρ mise à part 0.
On regarde maintenant la significativité de chacun des coefficients avec la commande `confint`.

```
confint(modele1)
```

```
##                 2.5 %      97.5 %
## ma1       -0.39175709 -0.10920951
## ma2       -0.26637769  0.05242359
## ma3       -0.53649112 -0.20296306
## ma4       -0.22359958  0.09272858
## ma5       -0.34454758 -0.07020225
## intercept -0.06758928  0.09587679
```

On voit que deux intervalles de confiance contiennent la valeur 0 on propose alors un modèle $MA(3)$.

En appliquant le même raisonnement pour le modèle $MA(3)$ on obtient les résultats suivants :

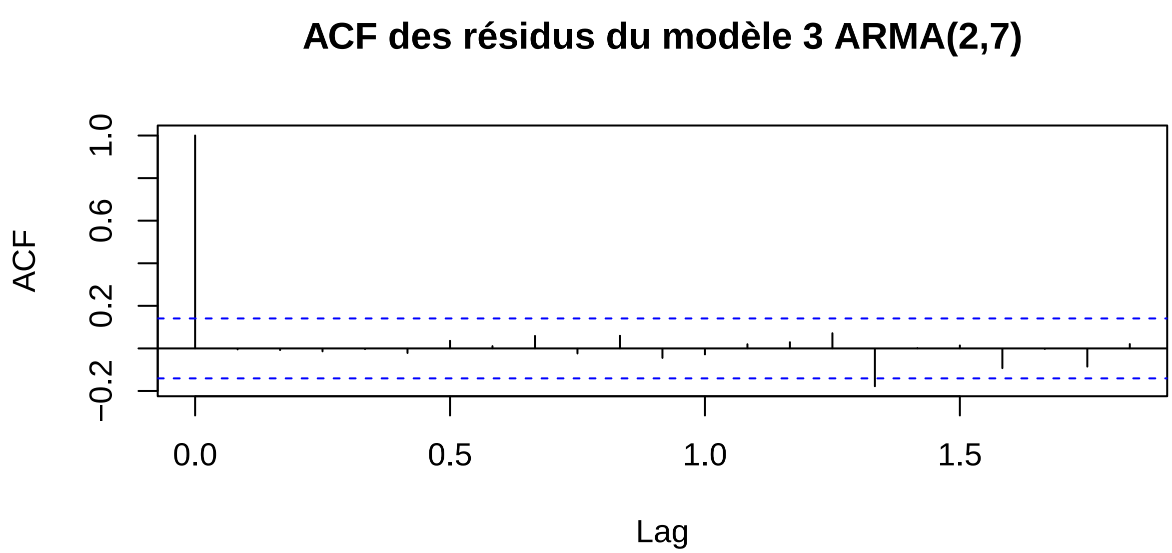



```
##                 2.5 %      97.5 %
## ma1       -0.3554098 -0.08660943
## ma2       -0.3626342 -0.10608924
## ma3       -0.6003659 -0.32190985
## intercept -0.1613714  0.16949728
```

De nouveau on accepte la blancheur des résidus et on remarque qu'aucun intervalles de confiance ne contient 0, on décide donc de séléctionner le modèle $MA(3)$ dans les modèles finaux pour nos prévisions.
Par ailleurs, après avoir effectué le même raisonnement pour le modèle $AR(5)$, on ne trouve pas de résultats concluants, donc on décide de ne pas retenir de modèles AR purs.

Modèle ARMA(2,7) :
En ce qui concerne les modèles $ARMA$ on effectue la méthode du plus compliqué au plus simple, et on trouve deux modèles interéssants : $ARMA(2,7)$ et $ARMA(2,5)$, on a :

```
modele3 = arima(res_debut,order = c(2,0,7))
acf(modele3$residuals, main = 'ACF des résidus du modèle 3 ARMA(2,7)')
```



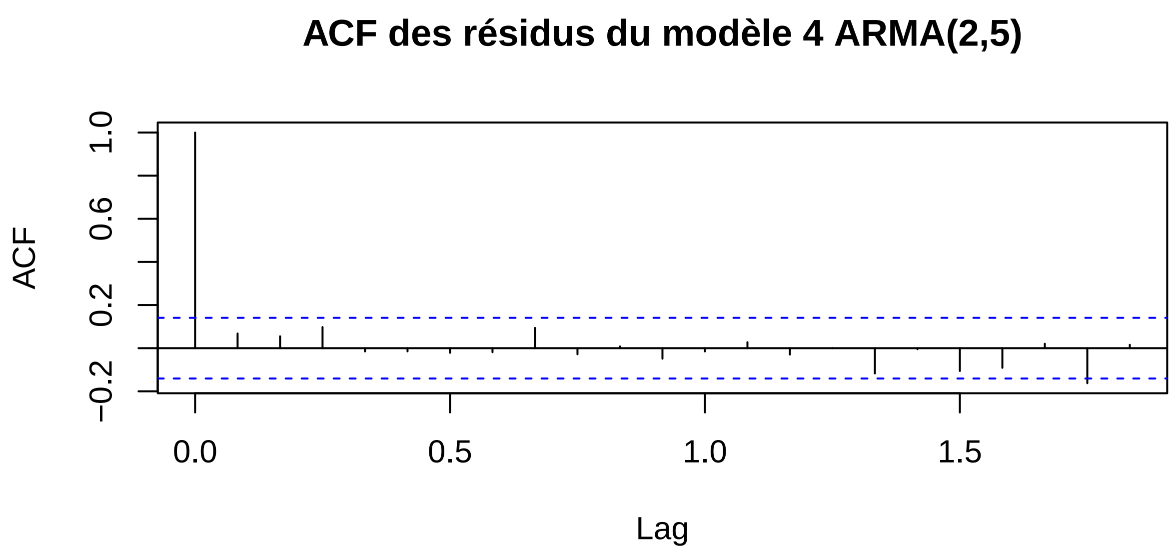
On accepte la blancheur des résidus pour ce modèle $ARMA(2,7)$.

```r
confint(modele3)
```

```
##                 2.5 %       97.5 %
## ar1       -0.88821809 -0.622222518
## ar2       -0.96032440 -0.712794663
## ma1        0.37192252  0.765135564
## ma2        0.42094388  0.817108836
## ma3       -0.84847076 -0.473280600
## ma4       -0.64047690 -0.272815118
## ma5       -0.77007674 -0.397406403
## ma6       -0.48986096 -0.162606429
## ma7       -0.32484861  0.004848883
## intercept -0.06795376  0.098949159
```

Aucun intervalles de confiance ne contient 0 (à l'exeption de ma7 mais la borne supérieure est très proche de 0 donc on décide de le conserver).
On décide donc de séléctionner le modèle $ARMA(2,7)$ dans les modèles finaux pour nos prévisions.

Modèle ARMA(2,5) :
On applique le même raisonnement pour le modèle $ARMA(2,5)$, on obtient les résultats suivants :




```
##                2.5 %     97.5 %
## ar1       -0.4755916 -0.1309671
## ar2       -1.0415213 -0.7258870
## ma1       -0.1434421  0.2190138
## ma2        0.5354507  0.8666031
## ma3       -0.9174151 -0.5584943
## ma4       -0.4062760 -0.1288182
## ma5       -0.6970475 -0.4278356
## intercept -0.1427827  0.1605247
```

On accepte la blancheur des résidus pour ce modèle $ARMA(2,5)$.
Aucun intervalles de confiance ne contient 0 (à l'exeption de ma1 mais on décide de le conserver car on n'a pas trouvé mieux en dessous).
On décide donc de séléctionner le modèle $ARMA(2,5)$ dans les modèles finaux pour nos prévisions.

On va maintenant comparer les modèles entre eux en affichant leur AIC et savoir lequel est le plus pertinent pour nos données.

```
aic <- c(modele2$aic,modele3$aic,modele4$aic)
names(aic) <- c("MA(3)","ARMA(2,7)","ARMA(2,5)")
aic
```

```
##     MA(3) ARMA(2,7) ARMA(2,5)
##  1525.784  1514.365  1518.738
```

On voit ici que l'AIC le plus petit est celui du modèle $ARMA(2, 7)$, le modèle $ARMA(2, 5)$ n'est pas très éloigné non plus.
Nous allons maintenant procéder à la prévison des 5 observations de `res_vrai` grâce à la commande `predict` et afficher les prédictions.

```
prev2=predict(modele2,n.ahead = 5, ci=T)
prev3=predict(modele3,n.ahead = 5, ci=T)
prev4=predict(modele4,n.ahead = 5, ci=T)
preds <- data.frame("MA(3)" = prev2$pred,
                    "ARMA(2,7)" = prev3$pred,
                    "ARMA(2,5)" = prev4$pred,
                    'vraies valeurs' = res_vrai)
preds
```

```
##          MA.3. ARMA.2.7. ARMA.2.5. vraies.valeurs
## 1  0.068788573  6.452747  9.189716      17.446394
## 2 -0.612024690  2.853540 -5.713271     -12.366960
## 3  2.590443429 -4.155928 -5.175197      -7.194178
## 4  0.004062932 -1.041883  2.268122      -3.402983
## 5  0.004062932  4.118962  7.024148      -3.131835
```

On peut comparer chaque prévision avec la vraie valeur en calculant leur erreur quadratique, on a :

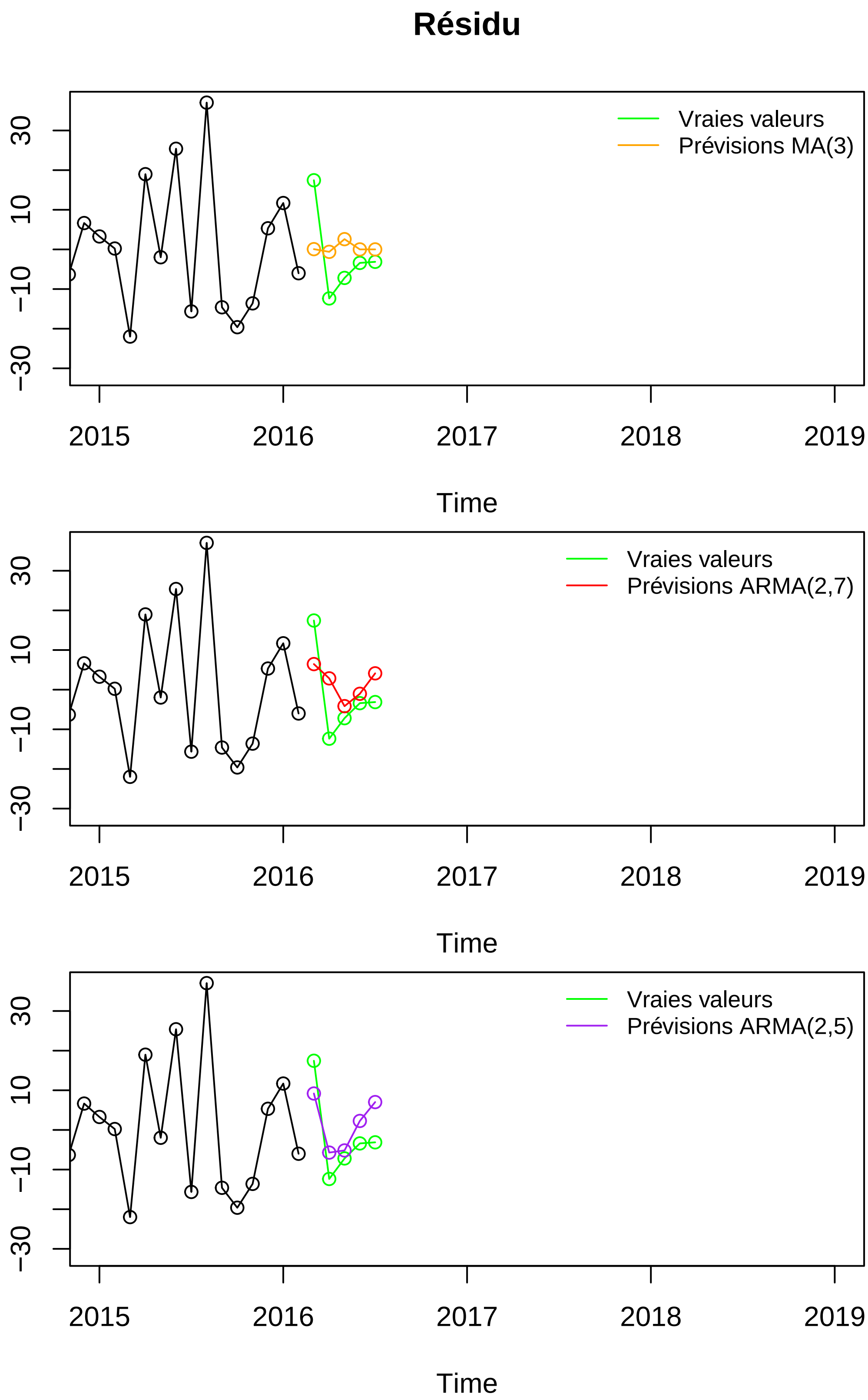
```
errquad <- c(sqrt(mean((prev2$pred-res_vrai)^2)),sqrt(mean((prev3$pred-res_vrai)^2)),
             sqrt(mean((prev4$pred-res_vrai)^2)))
names(errquad) <- c("MA(3)","ARMA(2,7)","ARMA(2,5)")
errquad
```

```
##     MA(3) ARMA(2,7) ARMA(2,5)
## 10.557843  9.164100  7.096844
```

On voit cette fois ci que l'erreur quadratique la plus petite est celle du modèle $ARMA(2, 5)$, au vu des AIC précédants, il semble que le modèle $ARMA(2, 5)$ soit le mieux adapté pour ces données.

Pour finir, on va afficher tous ces résultats.

```
plot.ts(res_debut,type='o',ylab='',main='Résidu',xlim=c(2015,2019))
lines(res_vrai,col='green',type='o')
lines(prev2$pred,col='orange',type='o') # prev3$pred, prev4$pred
legend(x='topright',legend=c("Vraies valeurs","Prévisions MA(3)"),col=c("green","orange"),
       bty = 'n',lty=1,cex=0.8)
```



Résidu
Vraies valeurs
Prévisions MA(3)
Vraies valeurs
Prévisions ARMA(2,7)
Vraies valeurs
Prévisions ARMA(2,5)
30
10
−10
−30
2015
2016
2017
2018
2019
Time

On peut revenir à la série initiale en ajoutant la tendance et la saisonnalité à la prédiction de notre résidu, on décide d'afficher les prévisions du modèle $ARMA(2,5)$, on a :

```
serie2 <- window(serie,start=c(2000,1),end=c(2016,2))
serie3 <- window(serie,start=c(2016,2),end=c(2016,7))
sais2 <- window(sais,start=c(2016,3),end=c(2016,7))
tend2 <-  window(tend,start=c(2016,3),end=c(2016,7))
serie4 <- prev4$pred  + sais2 + tend2
ts.plot(serie2,type='l',xlim=c(2010,2020.5),main = 'Série temporelle n°19')
lines(serie3,col="green",type='l')
lines(serie4,col="purple",type='o')
legend(x='topright',legend=c("Vraies valeurs","Prévisions ARMA(2,5)"),col=c("green","purple"),
       bty = 'n',lty=1,cex=0.8)
```