Séries Temporelles 1 (2A) - ENSAI

TP 1 : Premiers pas et quelques expériences numériques

José G. GOMEZ-GARCIA

Octobre 2023

Exercice 1

On étudie la série temporelle `nottem` des températures mensuelles enregistrées à Nottingham Castle entre janvier 1920 et décembre 1939. Cette série est disponible sous R :

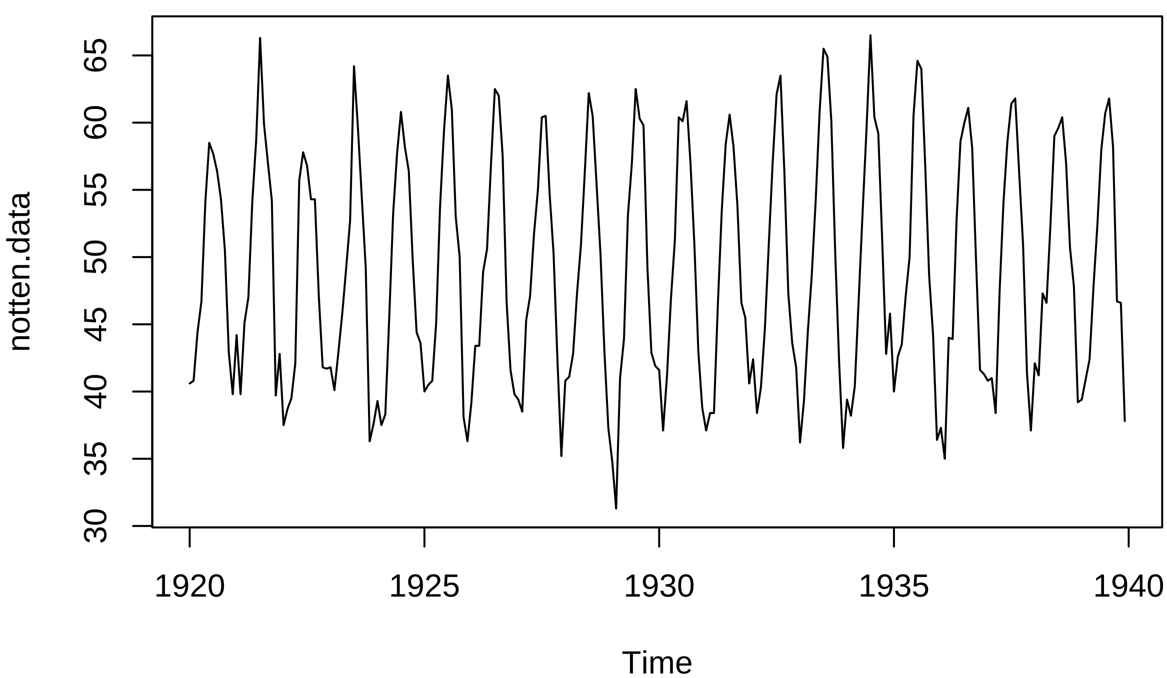
```
notten.data <- nottem
summary(notten.data)
```

```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  31.30   41.55   47.35   49.04   57.00   66.50 
```

1.- Regarder la structure de ces données, puis tracer la série à l'aide de la fonction `plot.ts`. Observe-t-on une tendance ? une saisonnalité ?

Voici la répresentation graphique en format de série temporelle `ts` :

```
plot.ts(notten.data)
```




```
# Tracez le même graphe avec plot(notten.data). Remarquez-vous une différence ?
```

A simple vue, on n'observe pas de tendance (globale) notable. On pourrait valider cela avec un test. Par exemple,

$$H_0 : \{X_t = at + b \ , \ a \neq 0\} \text{ (existence d'une tendance linéaire)}$$

vs

$$H_1 : \{X_t = b\} \text{ (pas de tendance linéaire)}$$

Pour cela, il suffit d'utiliser la commande `lm` *et voir le résultat du test donnée par la commande* `summary`.

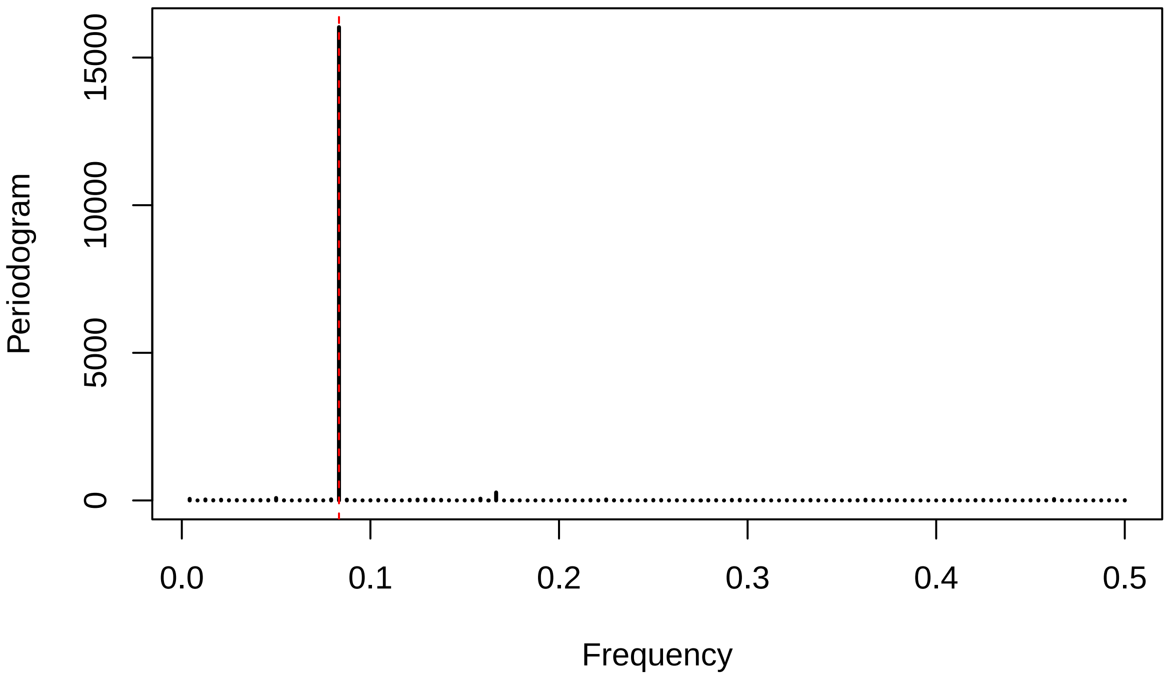
```
mois <- 1:length(notten.data)
mod0 <- lm(notten.data ~ mois)
summary(mod0)
```

```

Call:
lm(formula = notten.data ~ mois)

Residuals:
    Min      1Q  Median      3Q     Max
-17.675  -7.575  -1.620   8.028  17.882

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 48.301984   1.111112  43.472   <2e-16 ***
mois         0.006121   0.007994   0.766    0.445
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 8.58 on 238 degrees of freedom
Multiple R-squared:  0.002458,	Adjusted R-squared:  -0.001734
F-statistic: 0.5864 on 1 and 238 DF,  p-value: 0.4446
```

On ne peut pas rejeter H_0, on valide donc qu'on n'a pas de tendance statistiquement significative. Cependant, en regardant les données, on observe bien une saisonalité (qui n'est pas étonnant, vue la nature des données)

Remarque : Vous pouvez vérifier cela en utilisant le ***`periodogram`*** *et/ou un test de saisonalité (non vu en cours)*

```
TSA::periodogram(notten.data)
abline(v=1/12, col="red", lty=2)
```



Ce graphe montre bien que nous avous une période de $P = 12$, car $freq = 1/P$

2.- On tente de la désaisonnaliser par régression sur les fonctions trigonométriques, $t \to \cos\left(\frac{2\pi j}{12}t\right)$ pour $j = 1, 2, \ldots, 6$ et $t \to \sin\left(\frac{2\pi j}{12}t\right)$ pour $j = 1, \ldots, 5$. Pour cela, on crée une base trigonométrique à l'aide du code suivant :

```
freq <- matrix(1:6, nrow=1)/12
time <- matrix(1:length(notten.data), ncol = 1)
B <- cbind(cos(2*pi*time%*%freq), sin(2*pi*time%*%freq))[,-12]
B <- as.data.frame(B)
colnames(B) <- c('cos1','cos2','cos3','cos4','cos5','cos6','sin1','sin2','sin3',
                 'sin4','sin5')
library(knitr)
knitr::kable(head(B), digits = 4) #Si vous voulez regarder les premières 6 lignes du tableau B
```

cos1	cos2	cos3	cos4	cos5	cos6	sin1	sin2	sin3	sin4	sin5
0.866	0.5	0	-0.5	-0.866	-1	0.500	0.866	1	0.866	0.500
0.500	-0.5	-1	-0.5	0.500	1	0.866	0.866	0	-0.866	-0.866
0.000	-1.0	0	1.0	0.000	-1	1.000	0.000	-1	0.000	1.000
-0.500	-0.5	1	-0.5	-0.500	1	0.866	-0.866	0	0.866	-0.866
-0.866	0.5	0	-0.5	0.866	-1	0.500	-0.866	1	-0.866	0.500
-1.000	1.0	-1	1.0	-1.000	1	0.000	0.000	0	0.000	0.000

3.- Effectuer la régression sur la base créée en utilisant la fonction `lm` puis faire un résumé (`summary`) des résultats obtenus.

L'idée est de réaliser ici une régression (parfois appelé "régression harmonique") :

$$X_t = \mu + \sum_{j=1}^{6} \alpha_j \cos\left(\frac{2\pi j}{12}t\right) + \sum_{j=1}^{5} \beta_j \sin\left(\frac{2\pi j}{12}t\right) + \epsilon_t$$

afin d'obtenir une approximation de la partie saisonnière S_t de la decomposition additive de la série (X_t), i.e., $X_t = S_t + T_t + \epsilon_t$.

```
data_base <- data.frame(cbind(B,notten.data))
colnames(data_base)[12] <- 'temperature'
mod0 <- lm(temperature~. , data=data_base)
summary(mod0)
```

```

Call:
lm(formula = temperature ~ ., data = data_base)

Residuals:
   Min     1Q Median     3Q    Max
-7.890 -1.369  0.285  1.405  6.270

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 49.03958    0.14942 328.207  < 2e-16 ***
cos1        -9.24092    0.21131 -43.732  < 2e-16 ***
cos2        -0.08083    0.21131  -0.383   0.7024
cos3        -0.06417    0.21131  -0.304   0.7617
cos4         0.02167    0.21131   0.103   0.9184
cos5         0.05009    0.21131   0.237   0.8128
cos6        -0.19542    0.14942  -1.308   0.1922
sin1        -6.94091    0.21131 -32.848  < 2e-16 ***
sin2         1.49822    0.21131   7.090 1.66e-11 ***
sin3         0.34333    0.21131   1.625   0.1056
sin4         0.36517    0.21131   1.728   0.0853 .
sin5         0.14174    0.21131   0.671   0.5030
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.315 on 228 degrees of freedom
Multiple R-squared:  0.9304,    Adjusted R-squared:  0.9271
F-statistic: 277.3 on 11 and 228 DF,  p-value: < 2.2e-16
```

On observe que les éléments significatifs sont `cos1` (α_1), `sin1` (β_1), `sin2` (β_2) *et* l'intercept (μ).

4.- Recommencer en ne gardant que les éléments significatifs de la base.

```
mod1 <- lm(temperature~cos1+sin1+sin2, data=data_base)
summary(mod1)
```

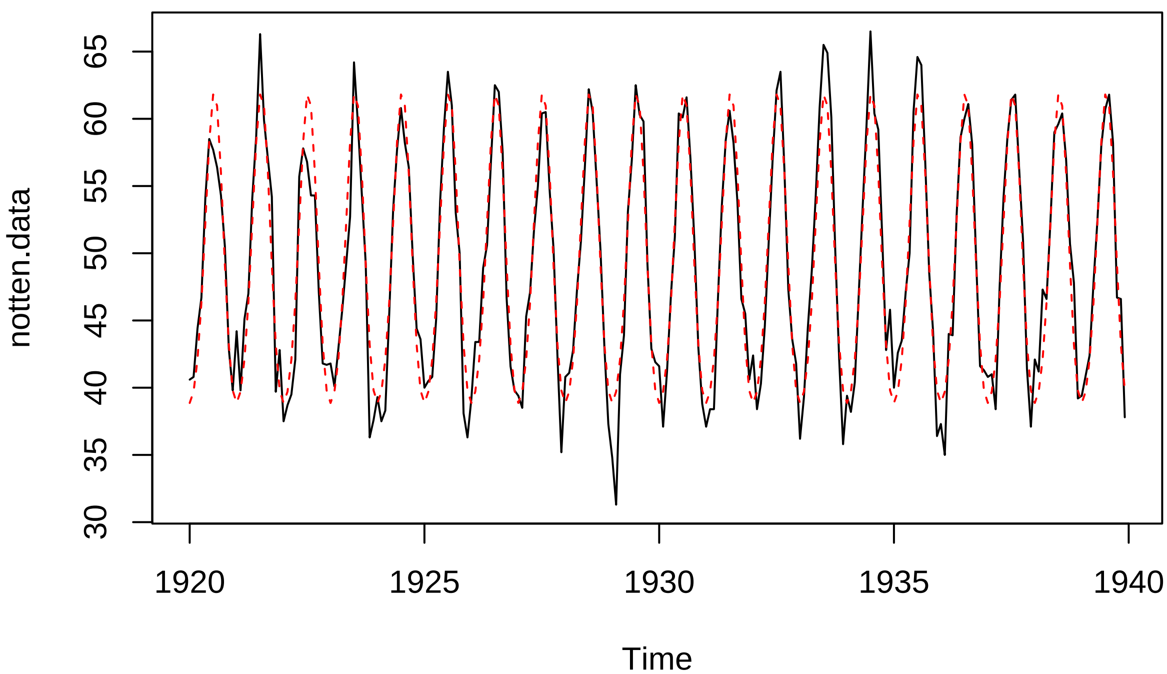
```

Call:
lm(formula = temperature ~ cos1 + sin1 + sin2, data = data_base)

Residuals:
    Min      1Q  Median      3Q     Max
-8.4056 -1.4098  0.3104  1.3961  6.0013

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  49.0396     0.1494 328.143  < 2e-16 ***
```

```
cos1         -9.2409     0.2113 -43.724  < 2e-16 ***
sin1         -6.9409     0.2113 -32.841  < 2e-16 ***
sin2          1.4982     0.2113   7.089 1.56e-11 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.315 on 236 degrees of freedom
Multiple R-squared:  0.928, Adjusted R-squared:  0.9271
F-statistic:  1014 on 3 and 236 DF,  p-value: < 2.2e-16
```

5.- Créer une série temporelle contenant la partie saisonnière obtenue sur cette base puis une contenant les résidus de la régression. On fera appel aux commandes `$fitted` et `$residuals`.

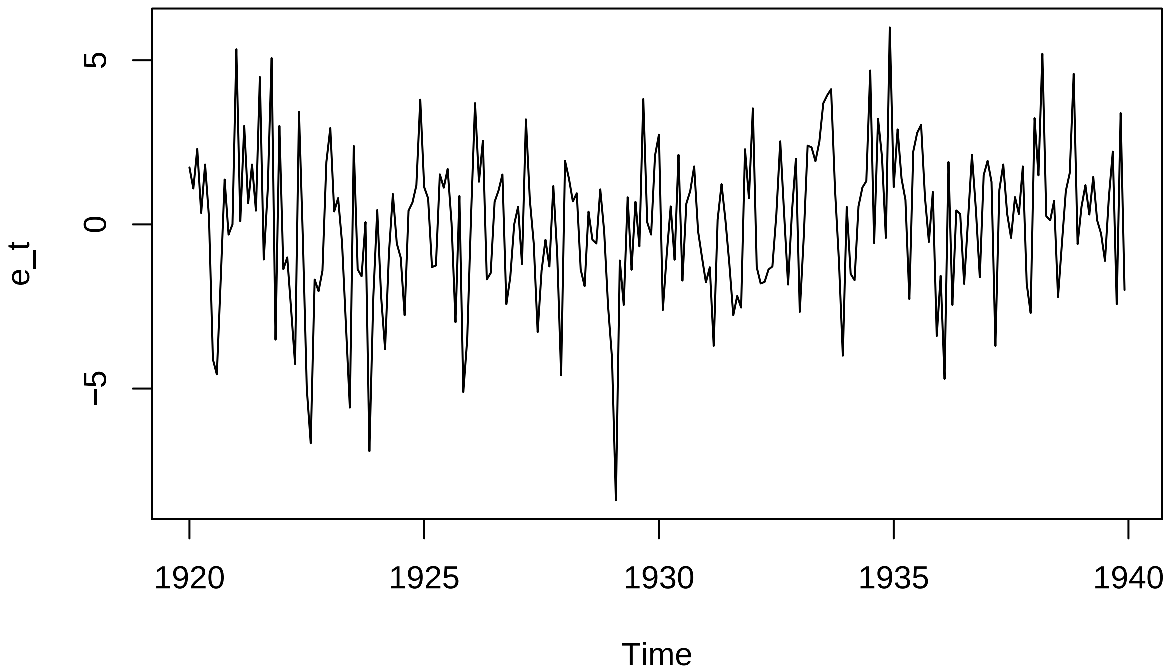
- *Voici la série de saisonalité S_t :*

```
#On utilise la commande $fitted :
S_t <- mod1$fitted.values
s_t <- ts(S_t, frequency = 12, start = c(1920,1)) #Pour obtenir la série placée
#sur les dates des observations notten.data

plot.ts(notten.data) #La série originale
lines(s_t, col="red", lty=2) #La série S_t est la rouge pointillée
```



- *Voici la série des résidus ϵ_t :*

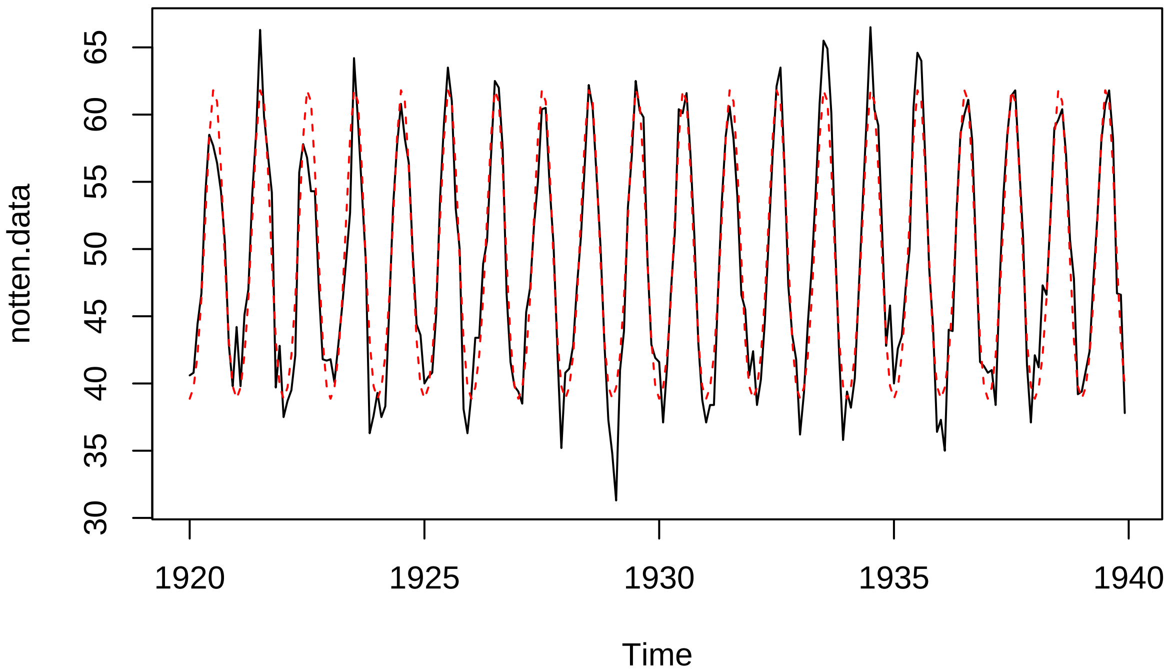
```
E_t <- mod1$residuals
e_t <- ts(E_t, frequency = 12, start = c(1920,1)) #Pour obtenir la série placée
#sur les dates des observations notten.data

plot(e_t)
```



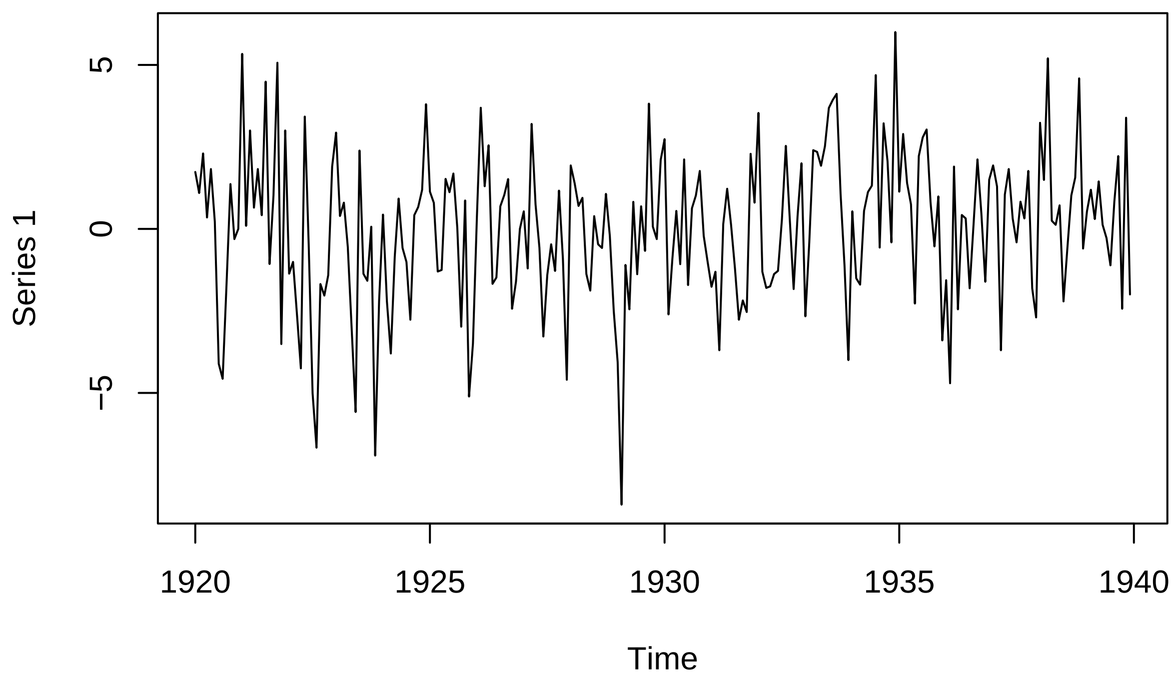
Bonus : vous pouvez vérifier si ces résidus estimés sont par exemple centrés, gaussiens, etc. Pour cela, vous pouvez tracer des histogrammes, qq-plots, etc.

Vous pourriez également trouver la partie saisonnière et les résidus d'une manière "manuelle", c'est-à-dire, sans utiliser les commandes `$fittel.values` *et* `$residuals` :

```
## Solution manuelle

#partie saisonniere :
S_t2 <- matrix(c(rep(1, length(time)),cos(pi*time/6), sin(pi*time/6), sin(pi*time/3)),
               ncol=4, byrow = FALSE)%*%as.matrix(mod1$coefficients)
s_t2 <- ts(S_t2, frequency = 12, start = c(1920,1))
plot.ts(notten.data)
lines(s_t2, col="red", lty=2)
```



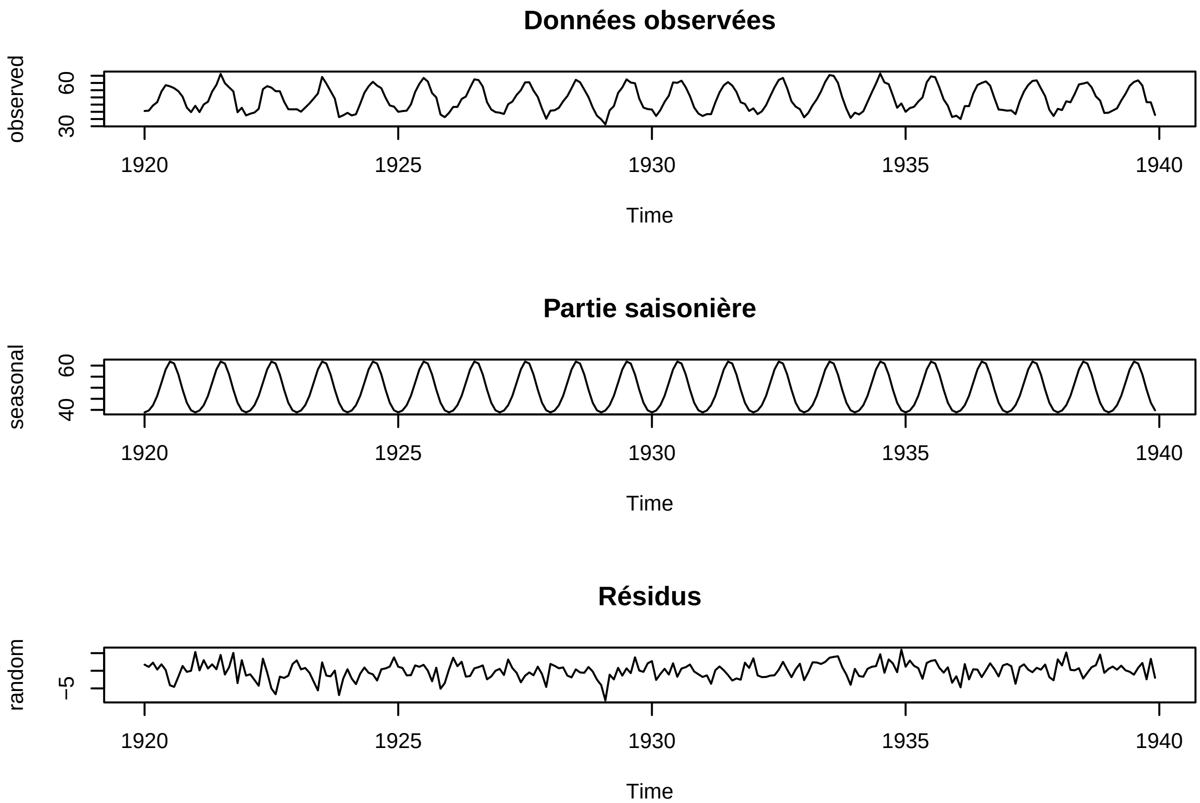

```r
#serie des residus :
E_t2 <- notten.data - S_t2
e_t2 <- ts(E_t2, frequency = 12, start = c(1920,1))
plot.ts(e_t2)
```



6.- Tracer les trois séries dans une même fenêtre.

```r
par(mfrow=c(3,1))
plot.ts(notten.data, main="Données observées", ylab = "observed")
plot.ts(s_t, main="Partie saisonière", ylab="seasonal")
```

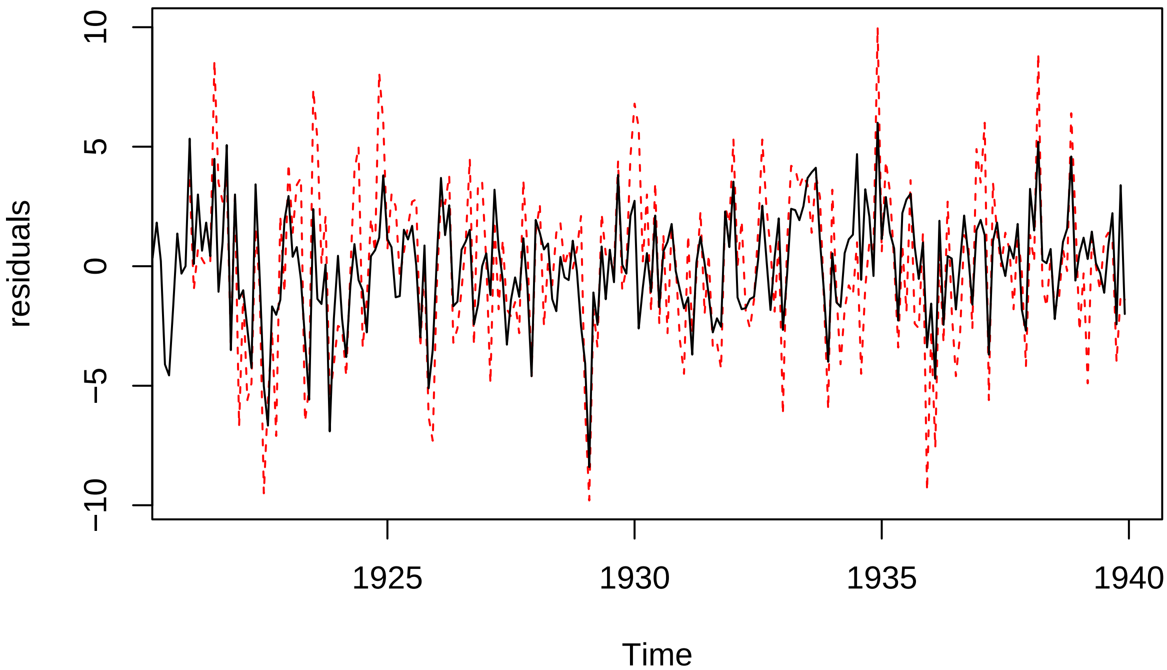
```r
plot.ts(e_t, main="Résidus", ylab="random")
```



7.- Filtrer la série de départ pour éliminer la saisonnalité en utilisant le filtre de différenciation saisonnière via la fonction **diff**. Superposer la série ainsi obtenue avec les résidus précédents. Commenter.

On a vu en cours que le filtre $\Delta_p = (I - B^p)$ *absorbe les saisonnalités de période* p*. Sachant que notre série a une composante saisonnière de période* $p = 12$ *on applique la commande suitante :*

```r
D_12 <- diff(notten.data, lag=12) #lag=p (la période de la série)

# Essayez de tester plusieurs lags pour observer si la série résultante conserve
# ou non une saisonnalité.

par(mfrow=c(1,1))
plot(D_12, col="red", lty=2, ylab="residuals")
lines(e_t)
```

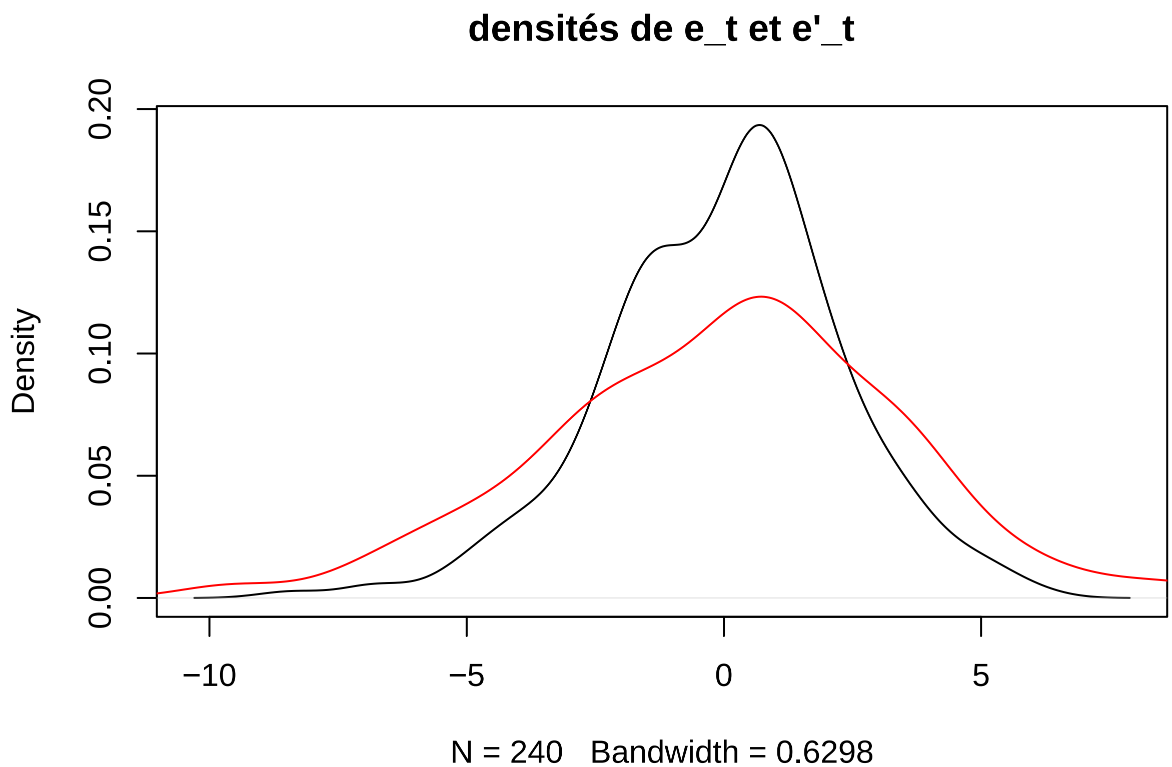


Sur le graphe, la trajectoire rouge correspond à la série filtrée et la trajectoire noire à la série des résidus estimés précedement. On peut constater une variance plus élévée dans le cas de la série filtrée et c'est dû à l'effet du filtre. En effet, si

$$X_t = m + S_t + \epsilon_t \Longrightarrow \epsilon'_t := \Delta_{12}(X_t) = \epsilon_t - \epsilon_{12}.$$

On peut observer cette dispersion en regardant les densités (empiriques) :

```
#hist(e_t, probability = TRUE, breaks = 12)
plot(density(e_t), col="black", main = "densités de e_t et e'_t")
lines(density(D_12), col="red")
```

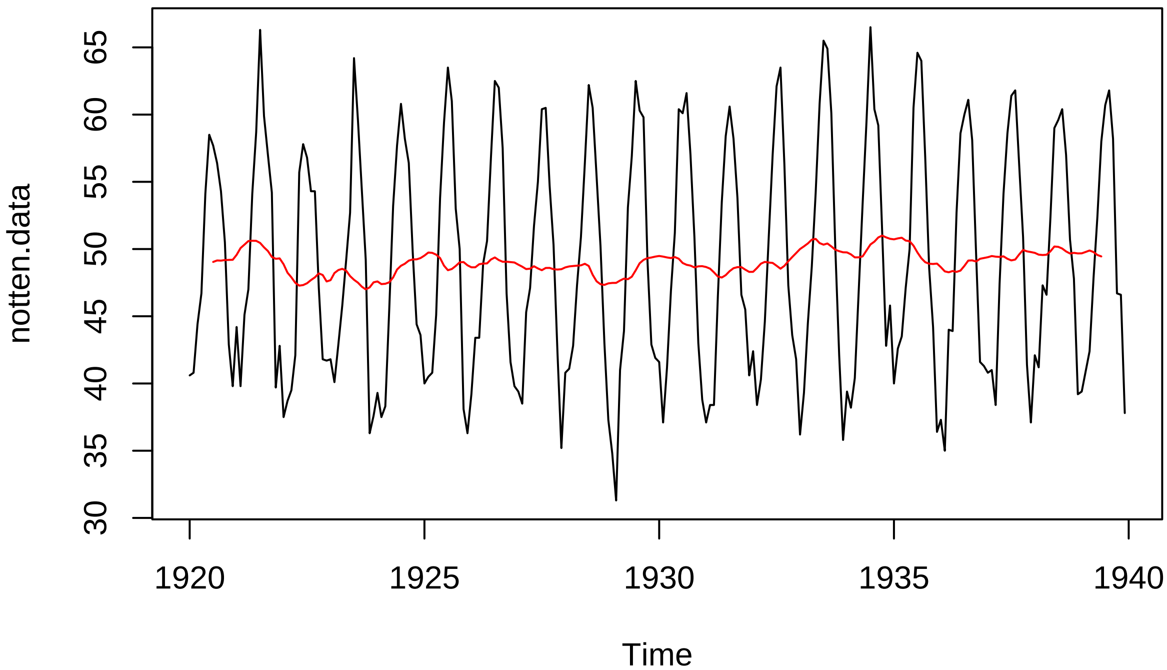


8.- Même question en utilisant cette fois la moyenne mobile arithmétique d'ordre 13 modifiée $M_6^*(B)$. On utilisera la fonction `filter` qui prend en argument la série à filter et un vecteur contenant les coefficients θ_i de la moyenne mobile à appliquer.

On sait que $M_m^*(B) = \frac{1}{2m}\sum_{k=-(m-1)}^{m-1} B^{-k} + \frac{1}{4m}(B^{-m} + B^m)$ *est un filtre qui absorbe la saisonnalité de période* $p = 2m$*. Ici,* $p = 12$*. Les coefficients de* $M_6^*(B)$ *sont donc dans le vecteur de dimension 13 :*

$$(\theta_{-6}, \theta_{-5}, \dots, \theta_5, \theta_6) = \left(\frac{1}{24}, \frac{1}{12}, \dots, \frac{1}{12}, \frac{1}{24}\right)$$

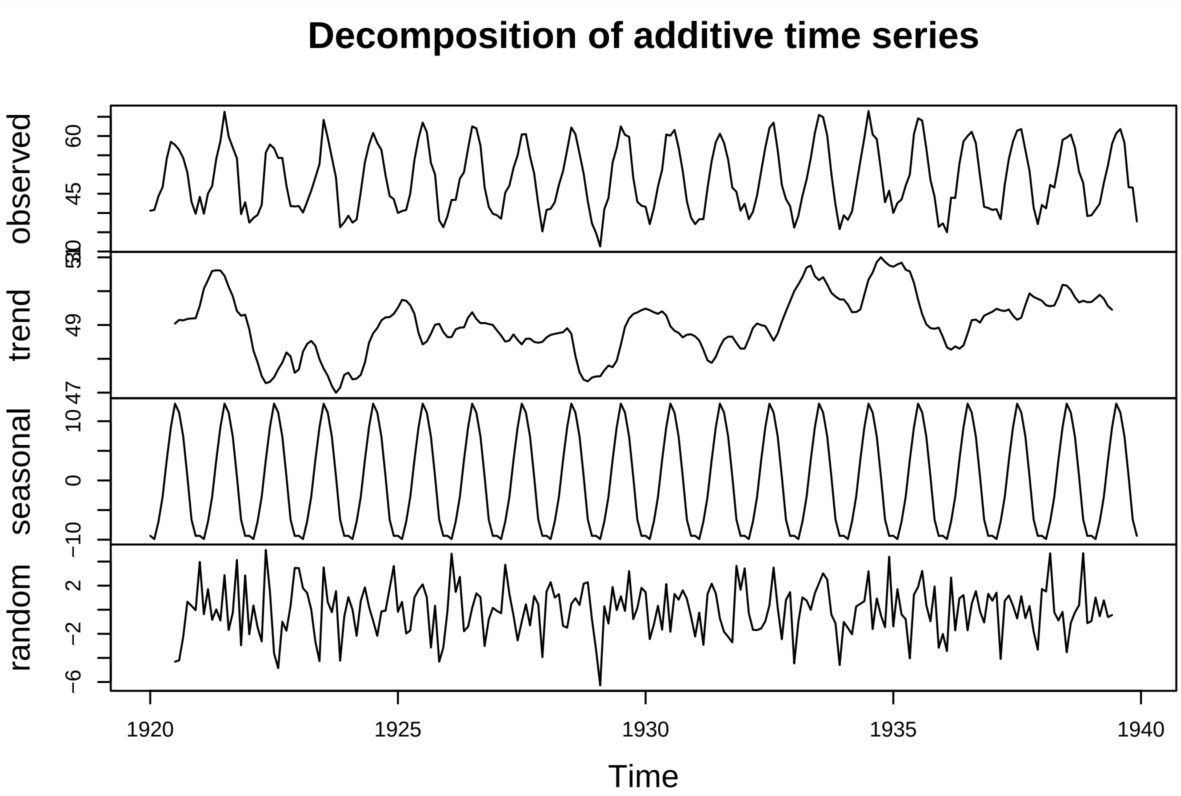
```
M_6_modif <- filter(notten.data,
                    filter = c(1/24, rep(1/12, 11), 1/24) #les coefficients du filtre
                    )
#plot(M_6_modif, col="red", lty=1)
plot.ts(notten.data)
lines(M_6_modif, col="red", lty=1)
```



On voit bien qu'il n'y a plus de saisonnalité. À différence du filtre Δ_{12}, ici, la série filtré est centrée autour de la moyenne globale. (Exercice, Pourquoi ?)

9.- Pour terminer, utiliser la fonction `decompose`. Que fait-elle? Faire un graphe de sa sortie. Commenter.

```
decomposition <- decompose(notten.data, type="additive")
plot(decomposition)
```



Regardez le suivant :

```
sum(na.omit(decomposition$trend - M_6_modif))
```

```
[1] 0
```

Cela montre que la série "trend" de `decompose` *est* $M_6^*(B)X_t$.

- *Comment sont-elles calculés les séries* `seasonal` et `random` (ou `residuals`) *dans la fonction* `decompose`? *(Discuté en TP)*

Exercice 2

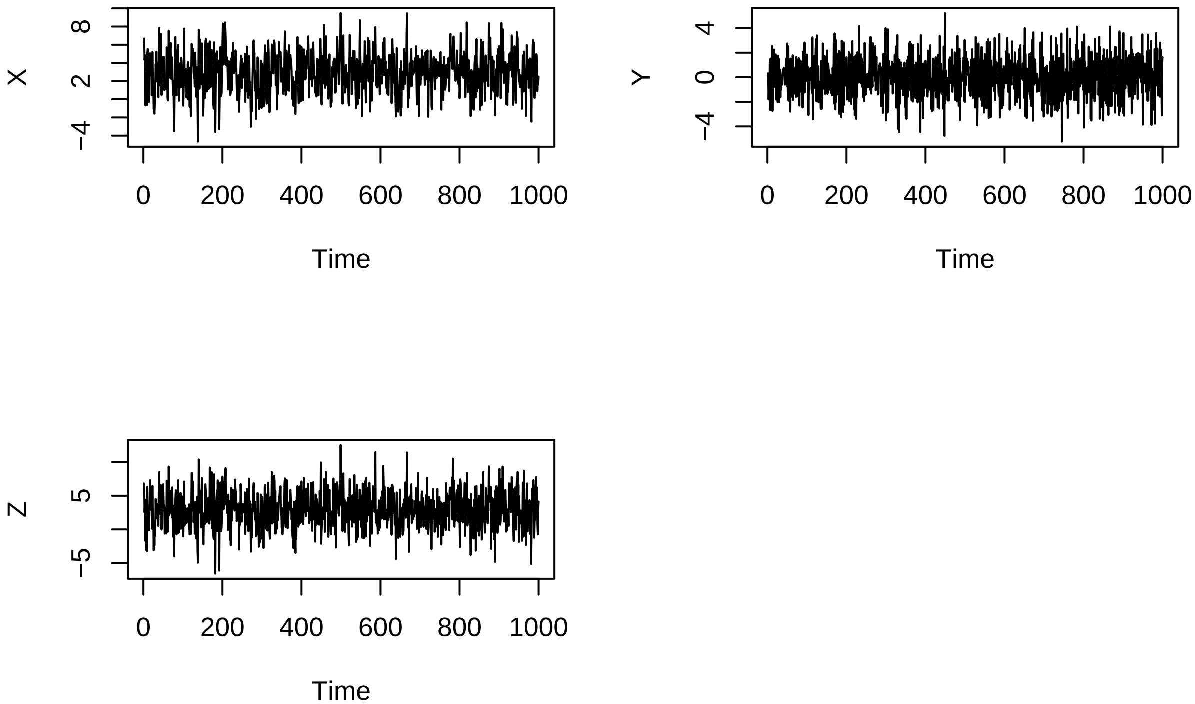
Soient $(X_t)_t$ et $(Y_t)_t$ deux processus MA définis pour tout $t \in \mathbb{Z}$ par

$$X_t = 3 + \epsilon_t + 2\epsilon_{t-1} \qquad et \qquad Y_t = \eta_t - \frac{1}{2}\eta_{t-1} + \frac{1}{3}\eta_{t-2}$$

où $(\epsilon_t)_t$ et $(\eta_t)_t$ sont deux bruits blancs indépendants de variance respective $\sigma_\epsilon^2 = 1$ et $\sigma_\eta^2 = 2$. On pose $Z_t = X_t + Y_t$.

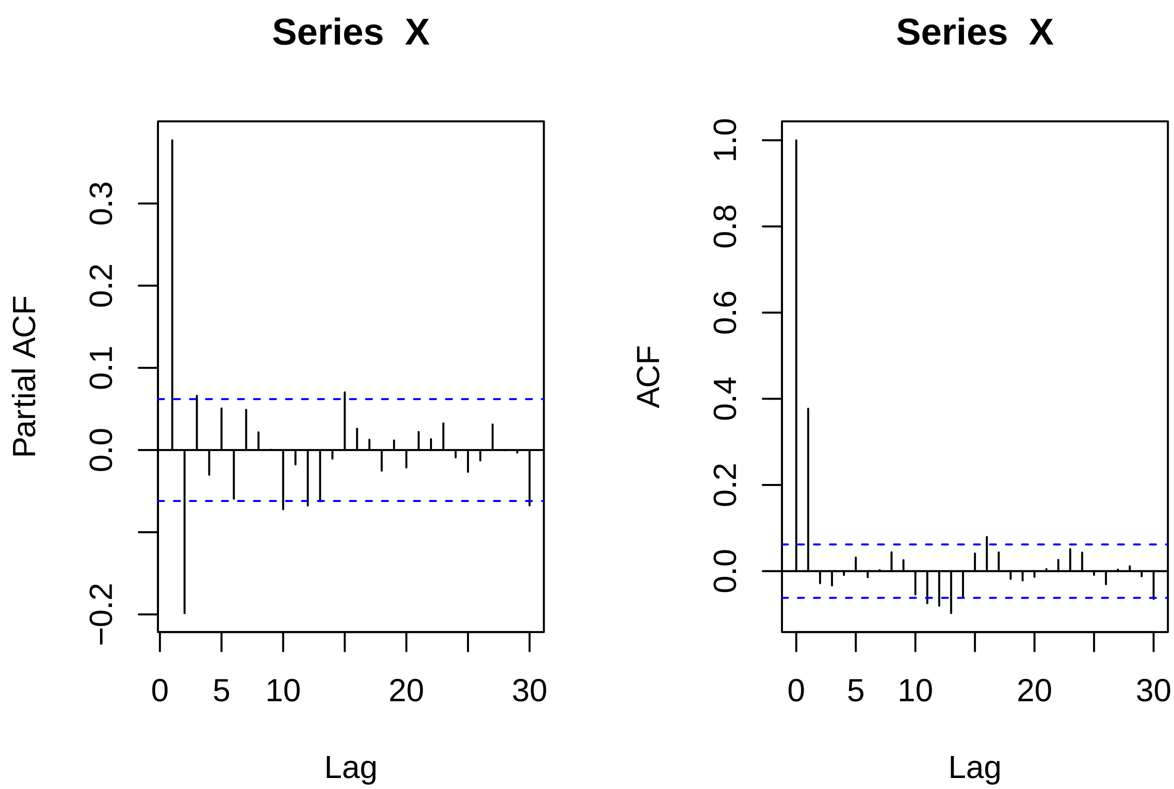
1.- Simuler une trajectoire de X, de Y et de Z, de taille $n = 1000$.

```
set.seed(666)
n <- 1000
m <- n + 2
epsilon <- rnorm(m, sd=1)
eta <- rnorm(m, sd=sqrt(2))
X <- 3 + epsilon[-1] + 2*epsilon[-m]
X <- X[1:n] #MA(1)
Y <- eta[-c(1,2)]-1/2*eta[-c(1,m)]+1/3*eta[-c(m-1,m)]
Y <- Y[1:n] #MA(2)
Z <- X + Y

par(mfrow=c(2,2))
plot.ts(X)
plot.ts(Y)
plot.ts(Z)
```

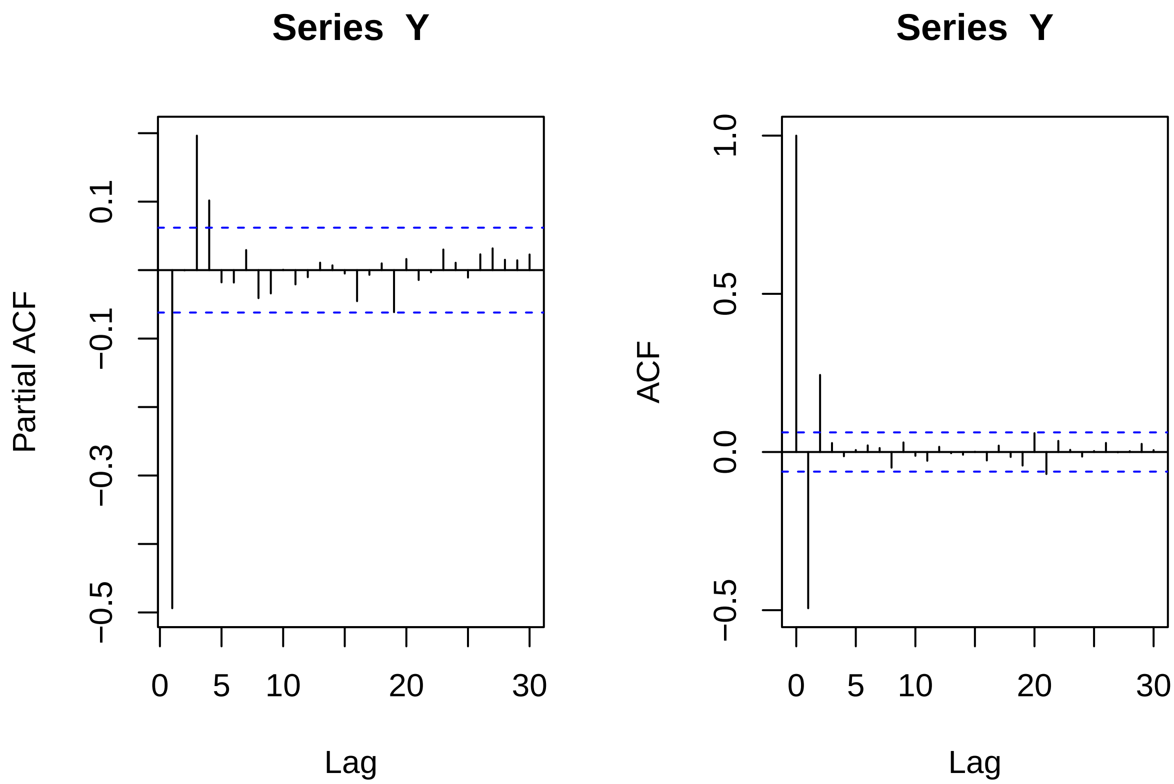


2.- A l'aide des fonctions `acf` et `pacf`, tracer les fonctions d'autocorrélation et d'autocorrélation partielle empirique de chacun des trois processus.

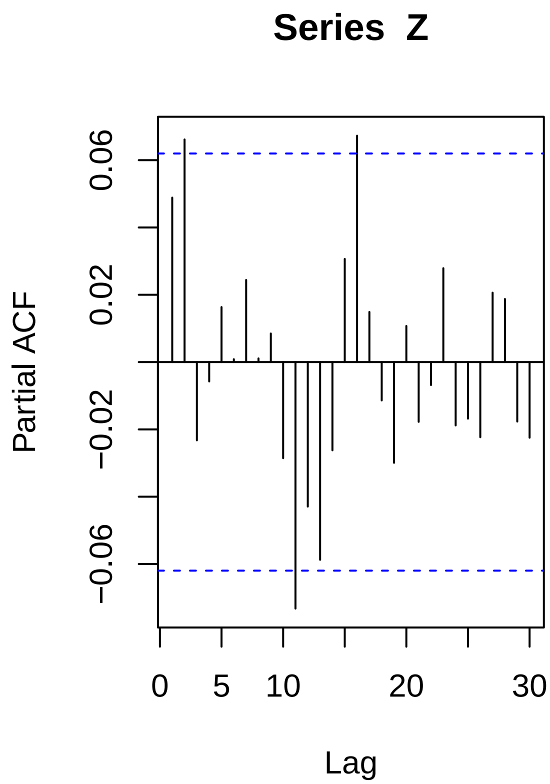
```
par(mfrow=c(1,2))
pacf(X)
acf(X)
```

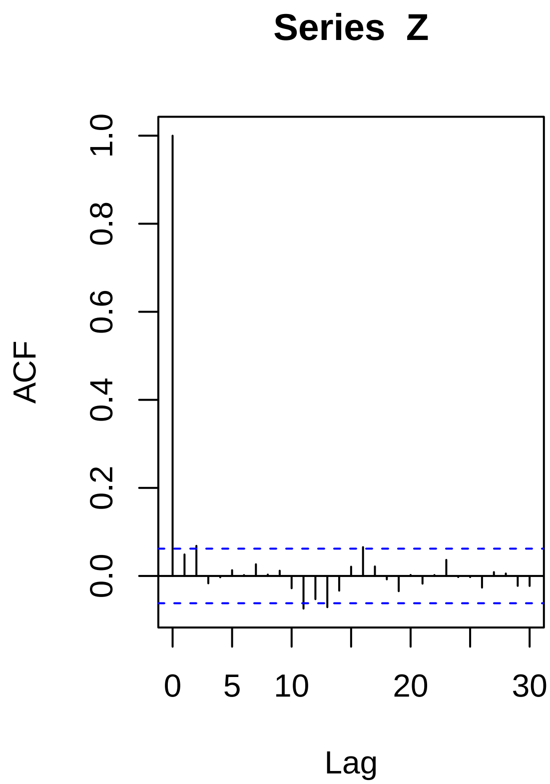


```r
par(mfrow=c(1,2))
pacf(Y)
acf(Y)
```




```r
par(mfrow=c(1,2))
pacf(Z)
acf(Z)
```





Exercice 3

Soit le processus AR(1) défini, pour tout $t \in \mathbb{Z}$, par

$$X_t = \mu + \phi X_{t-1} + \epsilon_t$$

où $(\epsilon_t)_t$ est un brut blanc de variance σ^2.

1.- Simuler une trajectoire de ce processus de taille $n = 1000$ pour $\mu = 1$, $\sigma^2 = 0.5$ et $\phi = 0.45$, puis pour $\phi = 0.95$ et enfin pour $\phi = 1$.

Pour que cela soit automatique, on construit d'abord la function suivante :

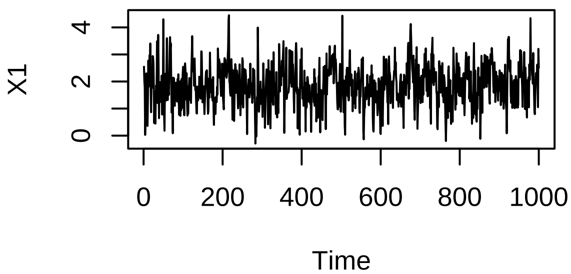
```
AR1 <- function(mu,phi,epsilon,size){
  X <-c(epsilon[1]) + mu
  for (t in 2:length(epsilon)){
    X[t] <- mu + phi*X[t-1] + epsilon[t]
  }
  return(X[1:size])
}
```

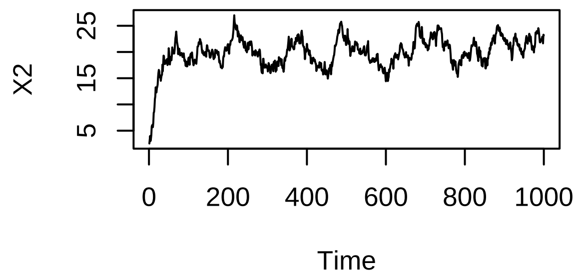
et on utilise cette fonction pour simuler chaque série temporelle :

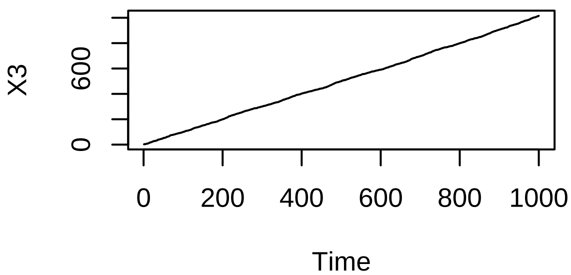
```
set.seed(1001)
epsilon1 <- rnorm(1010, sd=sqrt(0.5))
X1 <- AR1(mu=1, phi=0.45, epsilon = epsilon1, size=1000)
X2 <- AR1(mu=1, phi=0.95, epsilon = epsilon1, size=1000)
X3 <- AR1(mu=1, phi=1, epsilon = epsilon1, size=1000)
```

2.- Pour chaque valeur de ϕ représenter graphiquement la trajectoire simulée et commentez-les.

```r
par(mfrow=c(2,2))
plot.ts(X1)
plot.ts(X2)
plot.ts(X3)
```







La première série semble être stationnaire. La troisième série est clairement non-stationnaire. La deuxième série semble être entre les deux.

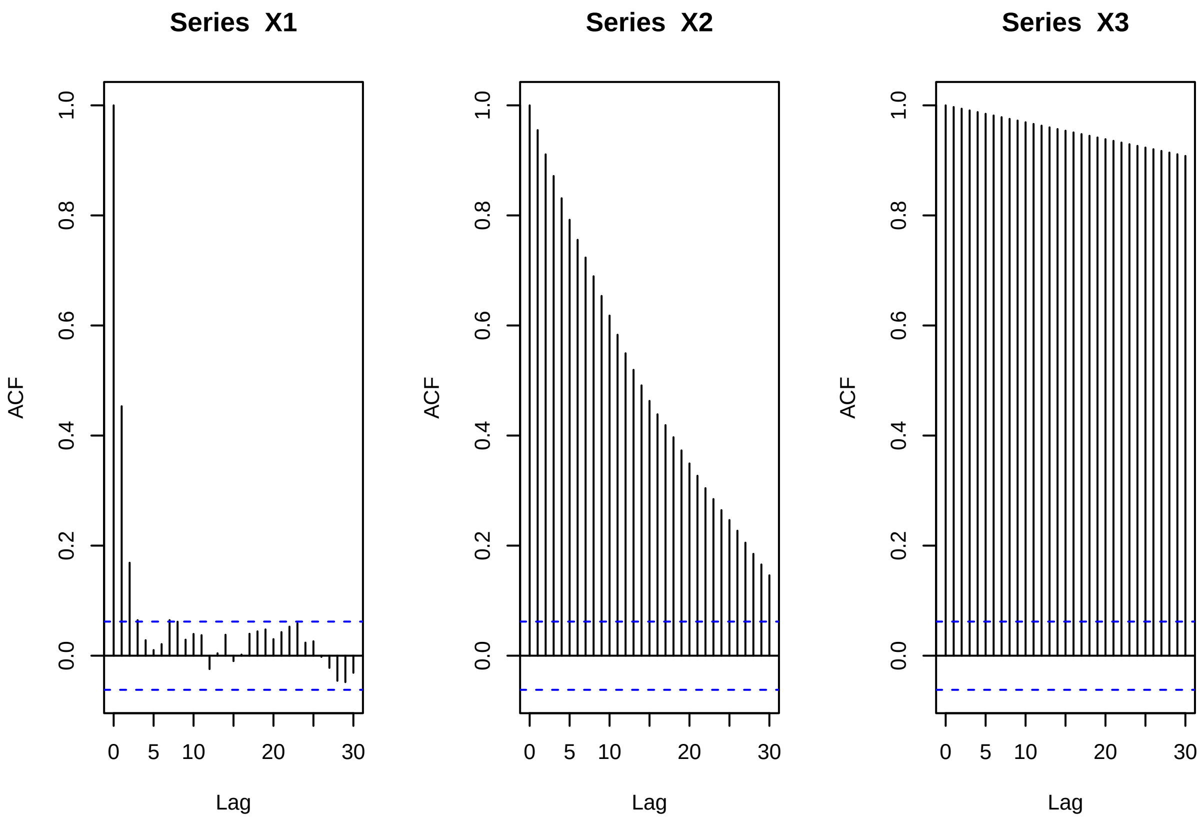
Remarque : Dans la théorie, la deuxième série est encore stationnaire. En effet, la série

$$X_t = \mu + \phi X_{t-1} + \epsilon_t$$

est stationnaire si $|\phi| < 1$. *La valeur* $\phi = 0.95$ *est presque à la frontière de la non-stationnarité.*

3.- A l'aide de la fonction `acf`, tracer la fonction d'autocorrelation empirique du processus. Dans quels cas est-il stationnaire ?

```r
par(mfrow=c(1,3))
acf(X1)
acf(X2)
acf(X3)
```



Ces graphiques confirment ce qu'on a dit prècedement. C'est-à-dire, que les séries 1 et 2 sont stationnaires et la 3eme non-stationnaire.

4.- Dans les cas stationnaires, estimer pour chaque $k \in \{10, \ldots, n\}$, μ, ϕ et σ^2 à partir de la trajectoire $x_1, \ldots, x_k$ (on utilisera la moyenne empirique et les équations de Yule-walker). Comparer graphiquement les estimations avec les vraies valeurs des paramètres en fonction de la valeur de k.

Ici, les équations de Yule-Walker sont en effet une seule équation :

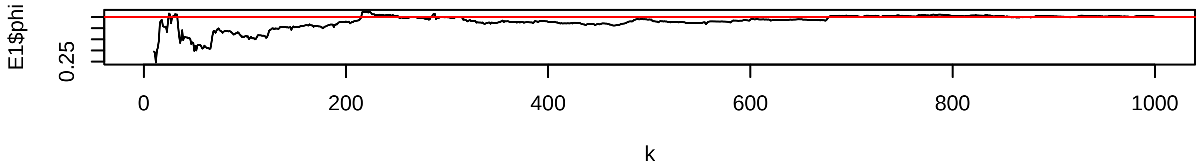
$$\phi = \rho(0)^{-1}\rho(1) = \rho(1)$$

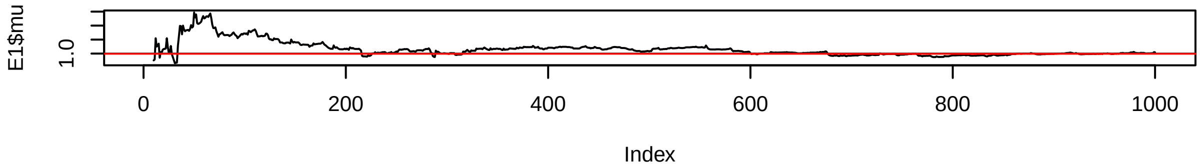
De cette manière, nous pouvons étudier le comporement des estimateurs de μ, ϕ et σ^2 ($\hat{\mu}_k$, $\hat{\phi}_k$, $\hat{\sigma}_k$) via la fonction suivante :

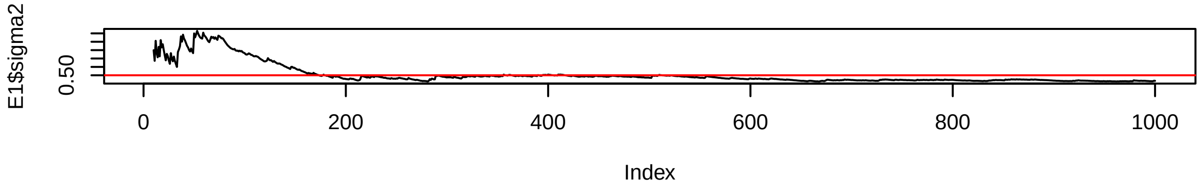
```
behavior.estimators.AR1 <- function(X){
  n <- length(X)
  hat.phi <-c(NA)
  hat.mu <- c(NA)
  hat.sigma2 <- c(NA)
  for (k in 10:n){
    aux <- acf(X[1:k], lag.max = 2, plot = FALSE)
    hat.phi[k] <- aux$acf[2] #Si l'on simplifie les equations de YW pour AR(1)
    hat.mu[k] <- (1-hat.phi[k])*mean(X[1:k])
    hat.sigma2[k] <- (1-hat.phi[k]^2)*var(X[1:k])
  }
  Parameters <- as.data.frame(cbind(hat.phi, hat.mu, hat.sigma2))
  names(Parameters) <- c("phi", "mu", "sigma2")
  return(Parameters)
}
```

- *Pour la première série :*

```
E1 <- behavior.estimators.AR1(X1)
par(mfrow=c(3,1))
plot(E1$phi, type="l", xlab="k")
abline(h=0.45, col="red")

plot(E1$mu, type="l")
abline(h=1, col="red")

plot(E1$sigma2, type = "l")
abline(h=0.5, col="red")
```

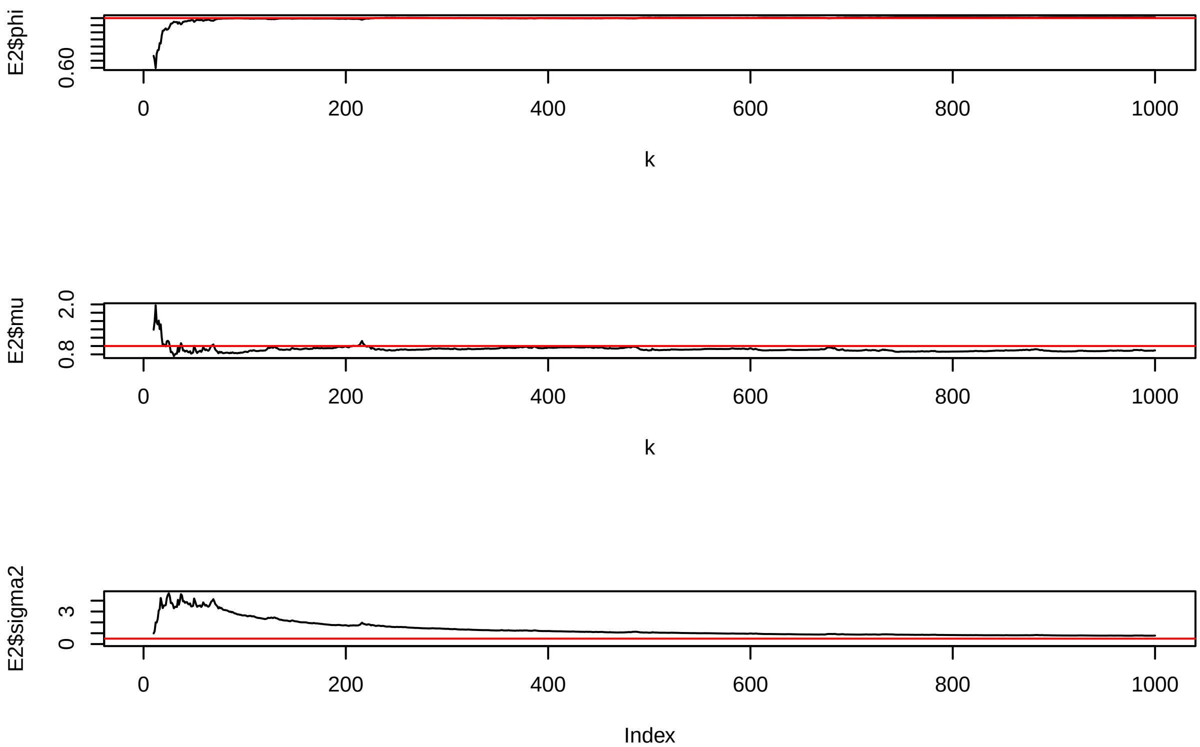






- *Pour la deuxième série :*

```
E2 <- behavior.estimators.AR1(X2)
par(mfrow=c(3,1))
plot(E2$phi, type="l", xlab="k")
abline(h=0.95, col="red")

plot(E2$mu, type="l", xlab="k")
abline(h=1, col="red")

plot(E2$sigma2, type = "l", ylim = c(0, max(na.omit(E2$sigma2))))
abline(h=0.5, col="red")
```

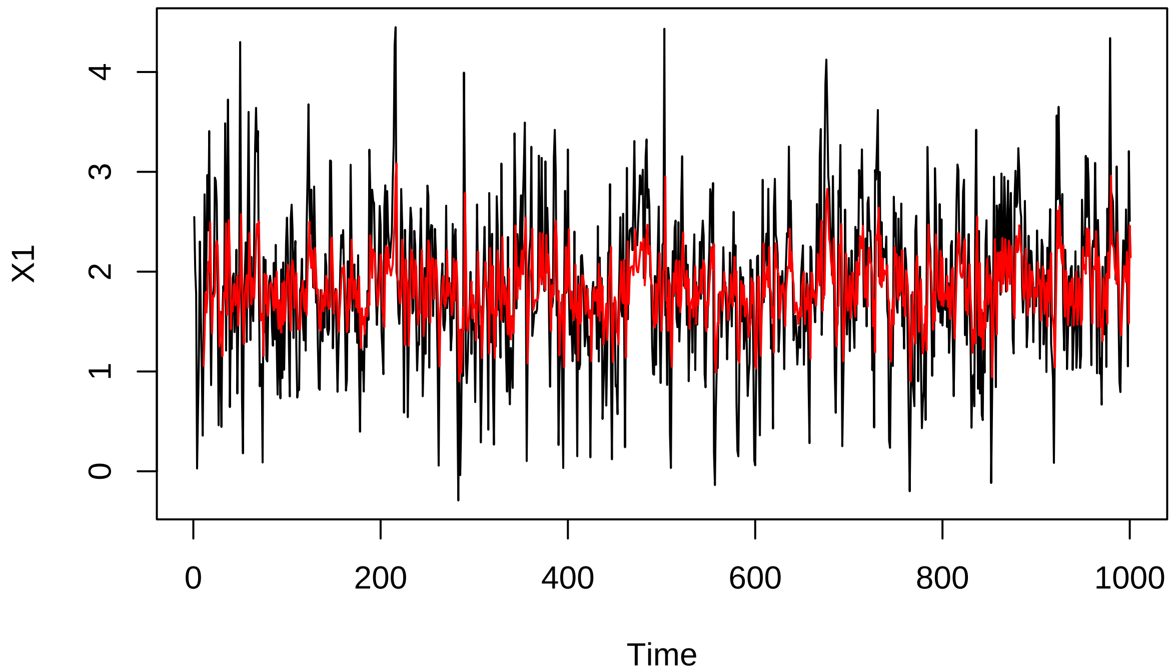


5.- Pour chaque $k \in \{10, \dots, n-1\}$, proposer un estimateur de la prévision de X_{k+1} à partir de l'observation de la trajectoire $x_1, \dots, x_k$. Superposer la vraie trajectoire et sa prévision en fonction de k. Que constatez-vous selon la valeur de ϕ ?

```
Prevision.AR1_1 <- function(X){
  previ <- c(NA)
  n <- length(X)
  for (k in 10:n){
    aux <- acf(X[1:k], lag.max = 2, plot = FALSE)
    hat.phi <- aux$acf[2]
    hat.mu <- (1-hat.phi)*mean(X[1:k])
    previ[k+1] <- hat.mu + hat.phi*X[k] #Voir la fin du Chapitre 3
  }
  return(previ)
}
```

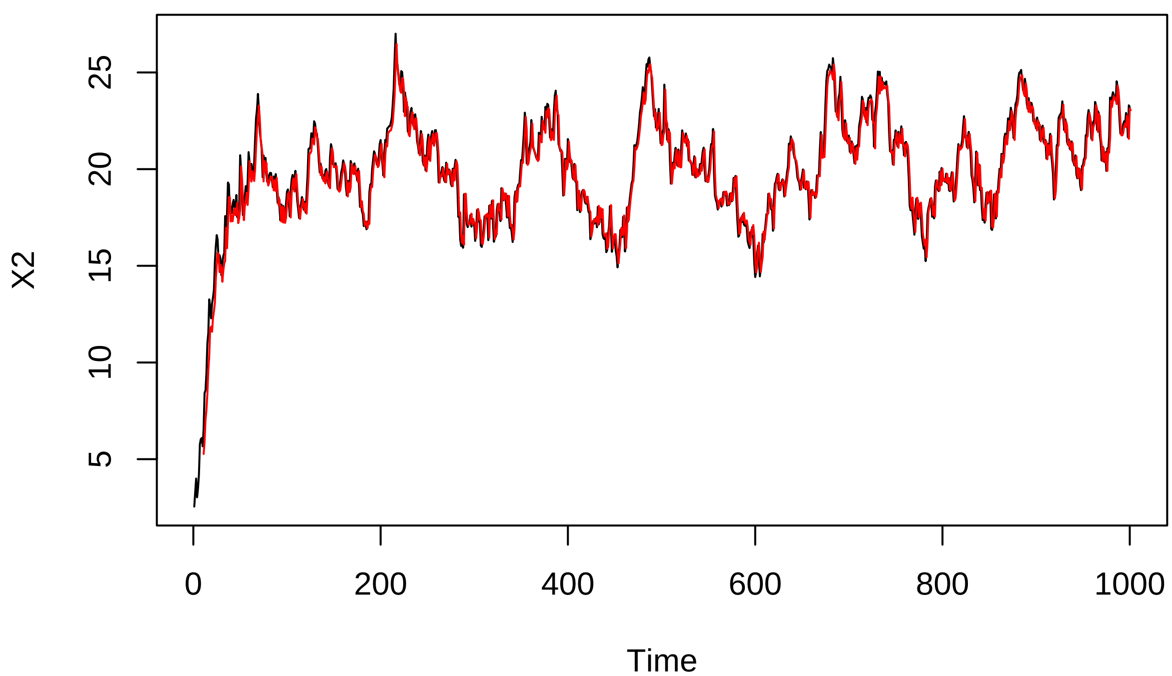
- *Prévisions $\hat{X}_k(1)$ pour la première série :*

```
Prev.X1 <- Prevision.AR1_1(X1)
plot.ts(X1)
lines(Prev.X1, col="red", lty=1)
```



- *Prévisions $\hat{X}_k(1)$ pour la deuxième série :*

```
Prev.X2 <- Prevision.AR1_1(X2)
plot.ts(X2)
lines(Prev.X2, col="red")
```

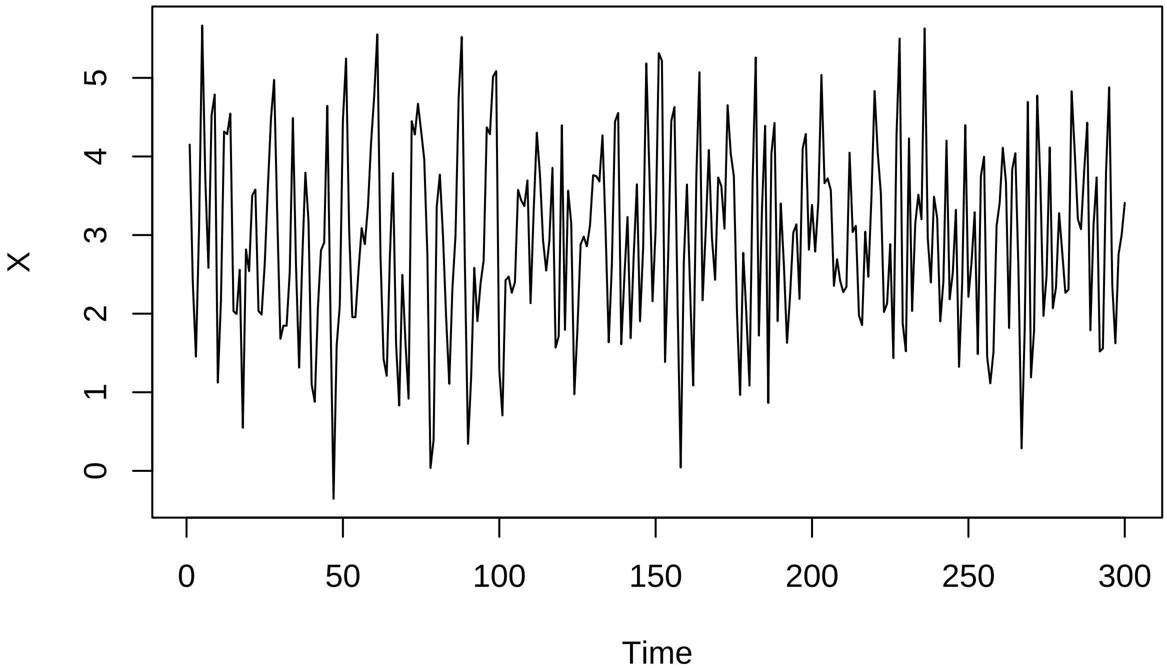


Exercice 4

Soit $(X_t)_t$ un processus MA(2) de paramètres $m = 3$, $\theta_1 = \frac{1}{2}$ et $\theta_2 = -\frac{1}{3}$.

1.- Simuler une trajectoire pour $t = 1, \dots, 300$.

```r
set.seed(134)
mu = 3
theta1 = 1/2
theta2 = -1/3
n = 300
bruit <- rnorm(n+2)
N = length(bruit)
X <- mu + bruit[-c(1,2)] + theta1*bruit[-c(1,N)] + theta2*bruit[-c(N-1,N)]
X <- X[1:n]
plot.ts(X)
```



2.- Utiliser la fonction `arima` pour estimer les paramètres m, θ_1 et θ_2 par maximum de vraisemblance à partir des 200 premières observations.

```r
ma2 <- arima(X[1:200], order = c(0,0,2))
ma2$coef
```

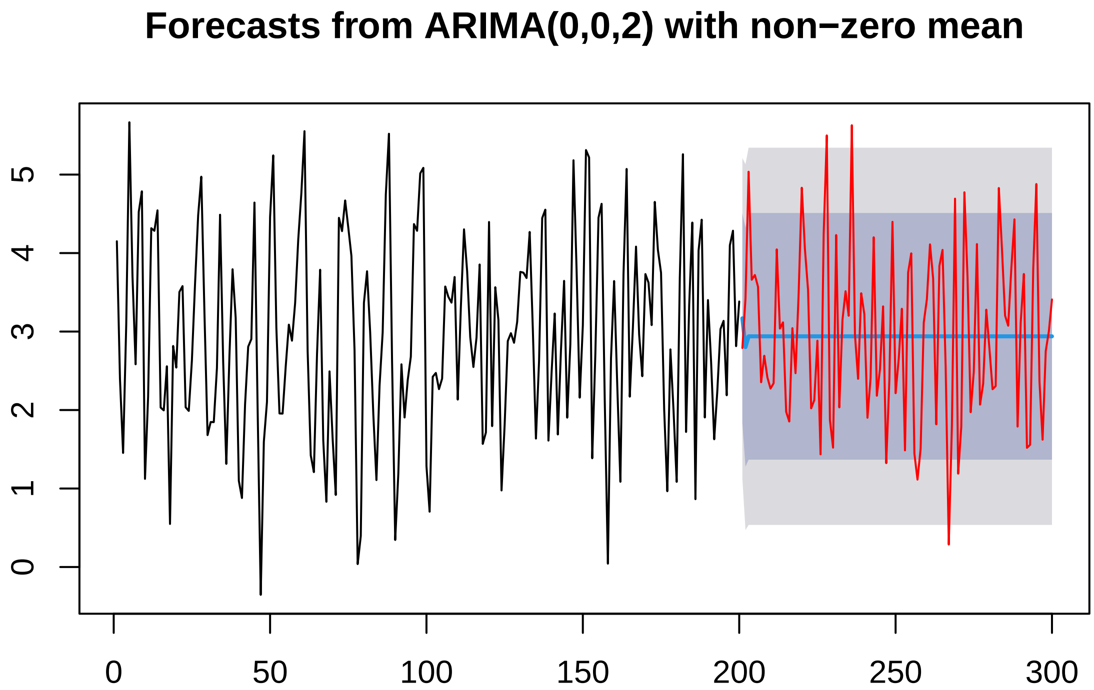
```
      ma1        ma2  intercept
 0.5493713 -0.2881389  2.9393657
```

3.- Ecrire un algorithme pour prédire les valeurs de X_t pour $t = 200, 201, \dots, 300$ et les comparer graphiquement avec les vraies valeurs observées.

Il faut calculer d'abord $P_{F_T}(X_{T+h})$, où $T = 200$ et $h = 1, 2, ..., 100$. $F_T = \bar{\text{Vect}}(X_T, X_{T+1}, ...)$. *Calcul fait (ou à faire) en TP. Le reste est facile à coder*

Sinon, on peut réaliser le calcul avec la librairie `forecast` :

```r
library(forecast)
prevision_h <- forecast(ma2, h=100)
plot(prevision_h)
lines(ts(X[201:300], start = 201, frequency=1), col="red")
```



Pas très intéressant comme prévision. Exercice : bricolez une solution plus réaliste !