المندوبية السامية للتخطيط

ⵜⴰⵎⵓⵙⵙⵏⵜ ⵜⴰⵎⴰⵜⵜⴰⵢⵜ ⵏ ⵓⵙⵏⵓⵎⵍ

HAUT-COMMISSARIAT AU PLAN

INSEA

INSEA
Institut National des
Statistiques et d'Economie
Appliquée

Projet : Séries Chronologiques

Réalisé par : Lagzouli Riyad DS

Mohamed Nabil Ouaja DS

Description des Données :

La série chronologique représente le nombre trimestriel de faillites d'entreprises à Paris sur une période de 27 ans, allant du premier trimestre de 1995 (1995-T1) au quatrième trimestre de 2021 (2021-T4).

Caractéristiques principales :

Fréquence : Données trimestrielles (4 observations par an).

Nombre total d'observations : 108 (27 ans × 4 trimestres).

Unité de mesure : Nombre d'entreprises en faillite par trimestre.

Annee	T1	T2	T3	T4
1995	1731	1338	1190	1643
1996	1690	1542	1230	1598
1997	1696	1670	1294	1540
1998	1435	1168	1007	1385
1999	1451	1251	894	1333
2000	1222	886	814	1176
2001	635	1164	828	1439
2002	1161	1326	935	1318
2003	1499	1099	907	1367
2004	1301	1168	924	1274
2005	1357	1235	855	1189
2006	964	900	699	1063
2007	1195	1039	743	1153
2008	1050	1035	767	1347
2009	1136	1136	955	1243
2010	1162	1033	836	1006
2011	1039	884	761	1009
2012	946	1001	738	1041
2013	1078	1028	819	1188
2014	1106	918	861	992
2015	1138	1019	731	1011
2016	1087	1119	668	1087
2017	998	883	711	959
2018	880	841	634	1016
2019	869	868	640	907
2020	700	354	565	708
2021	618	491	393	563

Récupération du fichier des données :

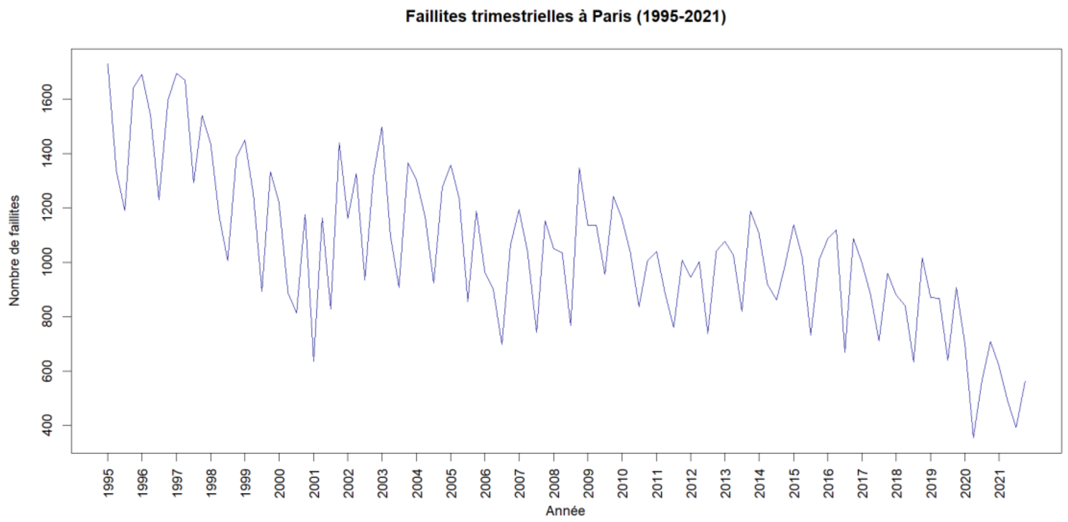
```
chemin <- "C:/Users/USER/OneDrive/Documents/S2/P2/Series chronologiques/defaillances_paris.txt"
data <- read.table(chemin, header = TRUE)
```

Division des données en un échantillon d'apprentissage (1995-2011) et un échantillon de validation (2012-2021) :

```
train <- window(ts_data, end = c(2011, 4))
test <- window(ts_data, start = c(2012, 1))
```

Conversion en série temporelle et représentation graphique de la série :

```
ts_data <- ts(data$Nombre_de_faillites,
              start = c(1995, 1),
              frequency = 4)

plot(ts_data,
     main = "Faillites trimestrielles à Paris (1995-2021)",
     ylab = "Nombre de faillites",
     xlab = "Année",
     col = "darkblue",
     xaxt = "n")


axis(1, at = seq(1995, 2021, by = 1), labels = seq(1995, 2021, by = 1), las = 2)
```



Interprétation :

La série montre une tendance décroissante et des pics qui révèlent une forte saisonnalité trimestrielle (justifiant l'utilisation d'un modèle SARIMA) et une non-stationnarité confirmant la présence d'une tendance.

Test de la stationnarité de la série :

```
adf.test(train)

kpss.test(train)
```

```
	Augmented Dickey-Fuller Test

data:  train
Dickey-Fuller = -2.3566, Lag order = 4, p-value = 0.4303
alternative hypothesis: stationary

	KPSS Test for Level Stationarity

data:  train
KPSS Level = 1.0997, Truncation lag parameter = 3, p-value = 0.01
```

On a p-value>0.05 (Test ADF) et p-value<0.05 (Test KPSS), donc on doit faire une différenciation pour rendre la série stationnaire et par la suite on reteste la stationnarité.

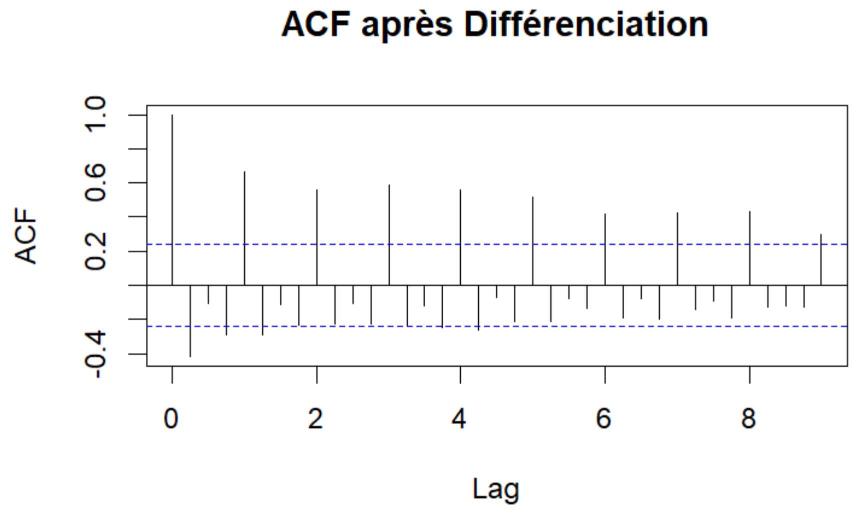
```
train_diff <- diff(train, differences = 1)
adf.test(train_diff)
kpss.test(train_diff)
```

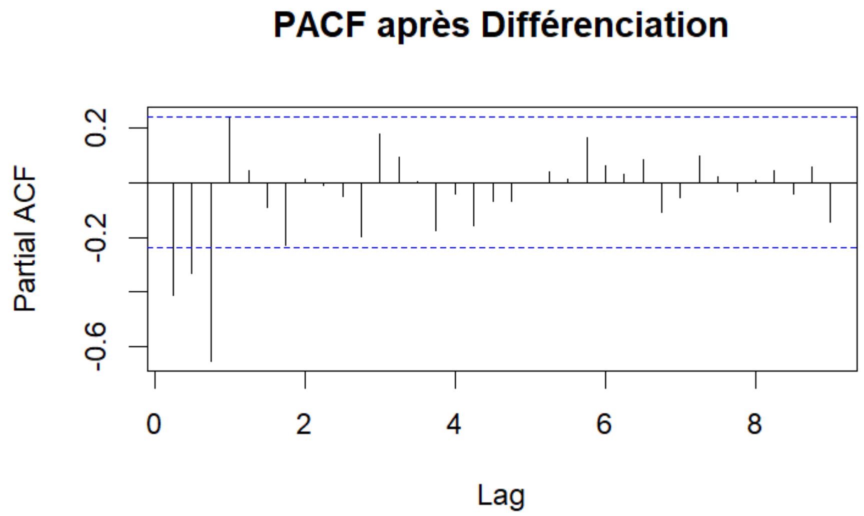
```
	Augmented Dickey-Fuller Test

data:  train_diff
Dickey-Fuller = -3.7934, Lag order = 4, p-value = 0.0243
alternative hypothesis: stationary

	KPSS Test for Level Stationarity

data:  train_diff
KPSS Level = 0.1294, Truncation lag parameter = 3, p-value = 0.1
```

On trouve p-value<0.05 (Test ADF) et p-value>0.05 (Test KPSS) d'où la série après différenciation (d=1) est **stationnaire** (les deux tests concordent).

Traçage et Analyse des corrélogrammes simple et partiel de la série :

```
acf(train_diff, lag.max = 36, main = "ACF après Différenciation")
pacf(train_diff, lag.max = 36, main = "PACF après Différenciation")
```





L'ACF montre des pics à des multiples de 4 (saisonnalité trimestrielle), et la PACF diminue rapidement. Cela indique la présence de composantes saisonnières, ce qui guide le choix du modèle SARIMA.

La méthode de Box et Jenkins :

Identification du processus et de son ordre :

```
auto.arima(train, d = 1, seasonal = TRUE)

 Series: train
 ARIMA(0,1,1)(2,1,0)[4]

 Coefficients:
           ma1     sar1     sar2
       -0.5172  -0.5532  -0.3612
 s.e.   0.1101   0.1176   0.1167

 sigma^2 = 29110:  log likelihood = -412.76
 AIC=833.51   AICc=834.2   BIC=842.09
```

On sait qu'on a une différenciation d=1 et la série représente une saisonnalité (D=1 et d+D $\leqslant$2) ce qui nous donne un processus SARIMA(0,1,1)(2,1,0).

Estimation des paramètres du modèle SARIMA(0,1,1)(2,1,0) :

```
# Estimation du modèle
model <- arima(train,
               order = c(0, 1, 1),
               seasonal = list(order = c(2, 1, 1), period = 4))

# Fonction pour afficher les résultats avec tests de Student
summary_sarima <- function(model) {
  # Calcul des statistiques
  estimates <- model$coef
  std_errors <- sqrt(diag(model$var.coef))
  t_values <- estimates/std_errors
  p_values <- 2*(1-pnorm(abs(t_values)))

  coef_table <- data.frame(
    Paramètre = names(estimates),
    Estimation = round(estimates, 4),
    Ecart_Type = round(std_errors, 4),
    t_value = round(t_values, 2),
    p_value = format.pval(p_values, eps = 0.0001),
    Signif = ifelse(p_values < 0.001, "***",
                    ifelse(p_values < 0.01, "**",
                           ifelse(p_values < 0.05, "*", "")))
  )

  # Affichage
  cat("=== Modèle SARIMA(0,1,1)(2,1,1)[4] ===\n\n")
  print(coef_table, row.names = FALSE)
  cat("\nSigma^2 =", model$sigma2,
      "\nLogLikelihood =", model$loglik,
      "\nAIC =", model$aic)
}

# Application
summary_sarima(model)
```

```
Paramètre Estimation Ecart_Type t_value p_value Signif
      ma1    -0.5260     0.1124   -4.68 < 1e-04    ***
     sar1    -0.0084     0.1300   -0.06 0.94861
     sar2    -0.1887     0.1285   -1.47 0.14208
     sma1    -1.0000     0.1406   -7.11 < 1e-04    ***
```

On a |t_value(ma1)| > 1.96, |t_value(sma2)| > 1.96 et |t_value(sar1)| < 1.96, |t_value(sma2)| < 1.96 d'où ma1 et sma2 sont significatifs contrairement à sar1 et sar2 (risque 5%).

```
model_simple <- arima(train,
                      order = c(0,1,1),
                      seasonal = list(order = c(0,1,1), period = 4))
```

Examen des résidus (Test de Box et Pierce) :

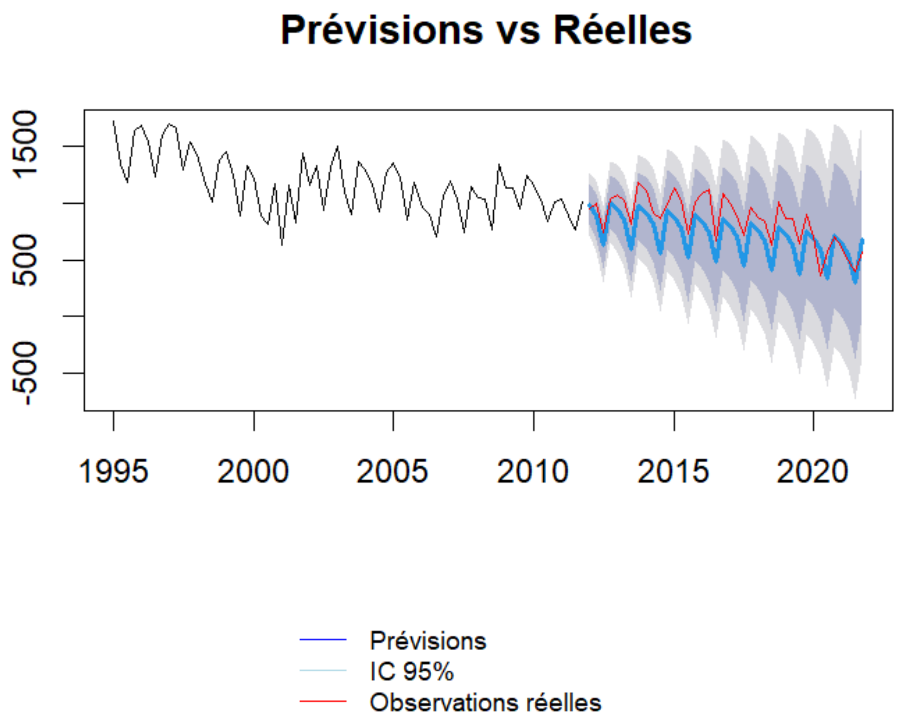
```
#Calcul des résidus
residus <- residuals(model_simple)

# Test de Box-Pierce
box_test <- Box.test(residus, type = "Box-Pierce")

# Affichage des résultats
cat("=== Test de Box-Pierce ===\n")
cat("Statistique X-squared:", box_test$statistic, "\n")
cat("p-value:", box_test$p.value, "\n")
```

```
=== Test de Box-Pierce ===
> cat("Statistique X-squared:", box_test$statistic, "\n")
Statistique X-squared: 0.1657632
> cat("p-value:", box_test$p.value, "\n")
p-value: 0.683905
```

On a p-value>0.05, donc les résidus sont non corrélés (bruit blanc) au seuil de 5%.

Prévision et représentation graphique de la série ainsi que la prévision :

```
# Préparation des prévisions
forecast_values <- forecast(model_simple, h = length(test))

plot(forecast_values, main = "Prévisions vs Réelles")
lines(test, col = "red")
legend("topleft",
       col = c("black", "blue", "red"), lty = 1)
```



Evaluation des prévisions obtenues et interprétation des résultats :

```
# Calcul des mesures d'erreur
mape=mean(abs(1-forecast_values$mean/test ))*100
mape
rmse=sqrt(mean((test - forecast_values$mean)^2))
rmse
```

```
> mape=mean(abs(1-forecast_values$mean/test ))*100
> mape
[1] 20.48444
> rmse=sqrt(mean((test - forecast_values$mean)^2))
> rmse
[1] 185.7862
```

On a les prévisions s'écartent de 20,48 % par rapport aux valeurs réelles d'où on peut accepter la prévision.

Test de normalité des erreurs :

```
# Test de normalité des erreurs
forecast_errors <- test - forecast_values$mean
shapiro.test(forecast_errors)
```

```
        Shapiro-Wilk normality test

data:  forecast_errors
W = 0.93677, p-value = 0.02695
```

On a p-value<0.05 d'où le test de normalité n'est pas vérifié.