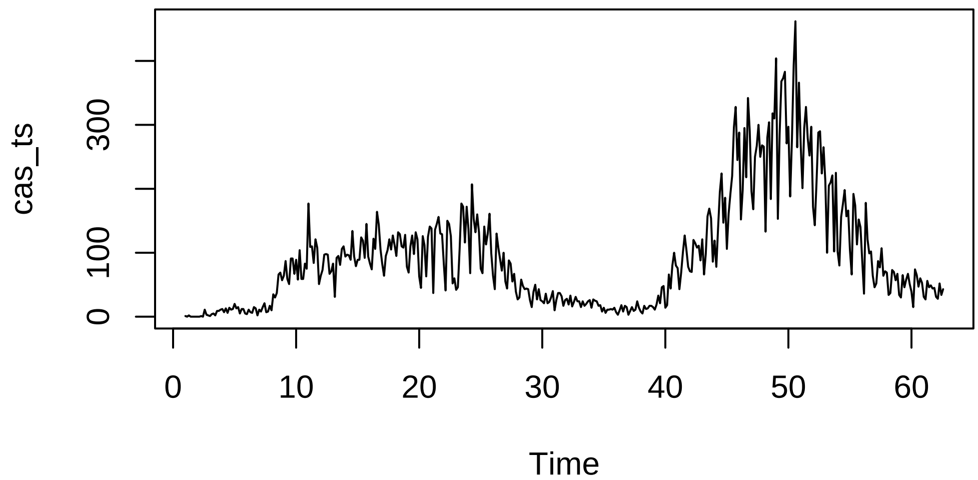
Projet Series Temporelles M1-SID 2023-2024

Abdoulaye SALL Code étudiant 1819037

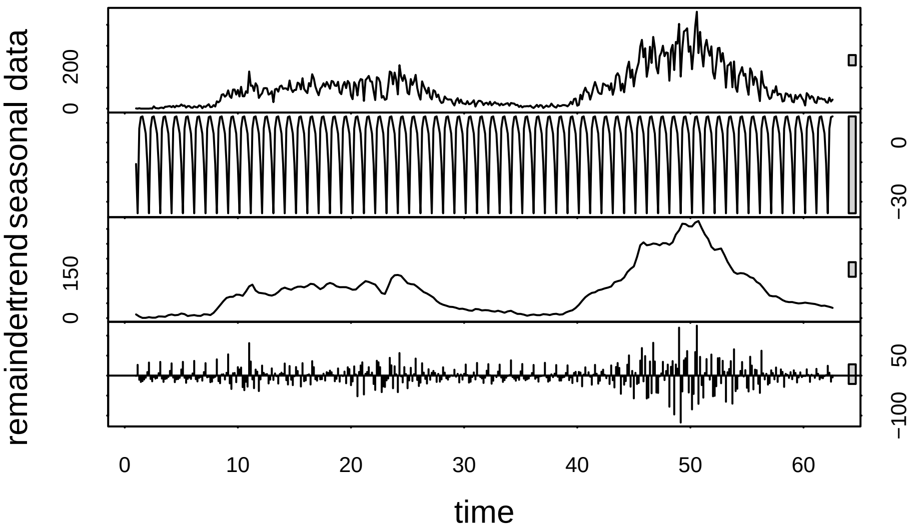
1. Importation et préparation des données

```r
# Import des packages nécessaires
library(forecast)
library(ggplot2)
library(tseries)
library(tidyr)

# Import des données
data <- read.csv("base_covid19.csv", header=TRUE, sep=";")

# Création de la série temporelle
cas_ts <- ts(data$CasPositif, frequency=7)

# Visualisation de la serie

plot(cas_ts)
```



2. Décomposition et analyse de la tendance

```
# Décomposer la série avec une saisonnalité hebdomadaire (fréquenc =7)

decomposition <- stl(cas_ts, s.window="periodic")
plot(decomposition)
```

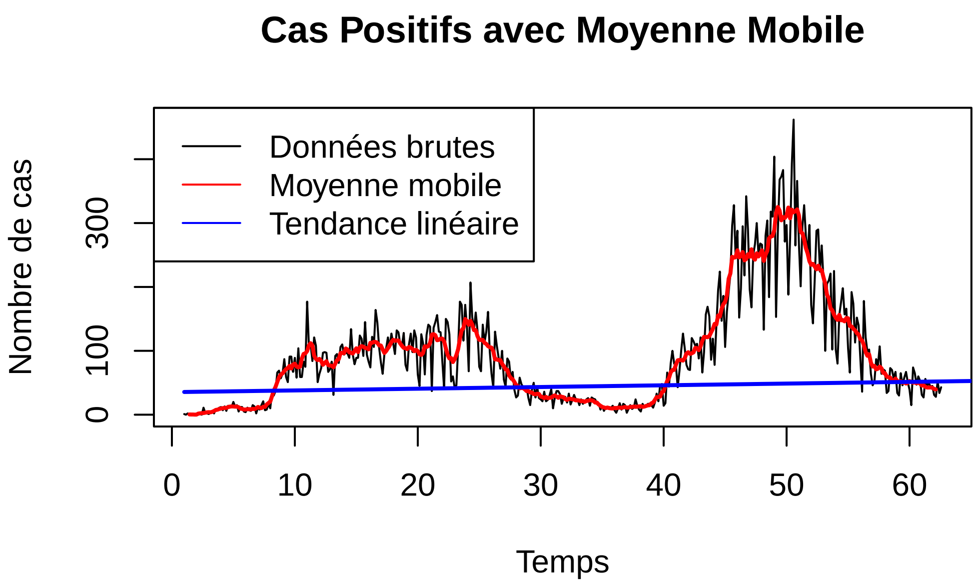


Interprétation

- **Tendance** : La tendance représente la direction générale de l'évolution des cas positifs. Elle montre une montée et une baisse au fil du temps, permettant d'identifier les phases critiques de la pandémie.
- **Saisonnalité** : La saisonnalité détecte des fluctuations répétitives hebdomadaires, probablement liées à des variations des tests effectués selon les jours de la semaine (moins de tests le week-end, par exemple).
- **Résidus** : Les résidus représentent les variations non expliquées par la tendance et la saisonnalité. Ils devraient idéalement être aléatoires (bruit blanc), mais des résidus importants peuvent indiquer des événements imprévus ou des changements de comportement non capturés par les autres composantes.

3. Moyenne mobile et tendance linéaire

```
# Calcul de la moyenne mobile
mm_7 <- ma(cas_ts, order=7)

# Visualisation avec moyenne mobile
plot(cas_ts, main="Cas Positifs avec Moyenne Mobile",
     ylab="Nombre de cas", xlab="Temps")
lines(mm_7, col="red", lwd=2)

# Ajout de la tendance linéaire
temps <- 1:length(cas_ts)
modele_lineaire <- lm(as.vector(cas_ts) ~ temps)
lines(temps, predict(modele_lineaire), col="blue", lwd=2)
legend("topleft", legend=c("Données brutes", "Moyenne mobile", "Tendance linéaire"),
       col=c("black", "red", "blue"), lty=1)
```



Interprétation

La courbe de la moyenne mobile sur 7 jours lisse les fluctuations quotidiennes et montre une tendance plus claire. On voit ici que, malgré les fluctuations journalières, la tendance générale peut être identifiée à travers cette moyenne mobile.

4. Modélisation Box-Jenkins

Tests de stationnarité

```r
# Test sur série originale
adf_test <- adf.test(cas_ts)
print("Test ADF sur série originale :")
```

```
[1] "Test ADF sur série originale :"
```

```r
print(adf_test)
```

```

	Augmented Dickey-Fuller Test

data:  cas_ts
Dickey-Fuller = -0.70951, Lag order = 7, p-value = 0.9694
alternative hypothesis: stationary
```

```r
# Test sur série différenciée
cas_diff <- diff(cas_ts)
adf_test_diff <- adf.test(cas_diff)
print("Test ADF sur série différenciée :")
```

```
[1] "Test ADF sur série différenciée :"
```

```r
print(adf_test_diff)
```

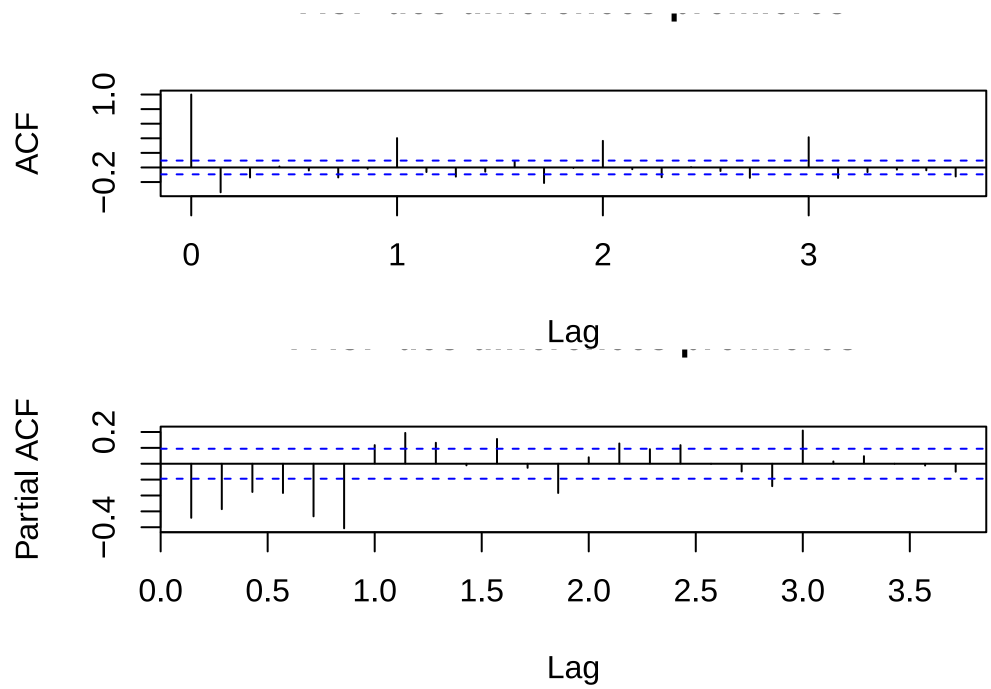
```

	Augmented Dickey-Fuller Test

data:  cas_diff
Dickey-Fuller = -9.0383, Lag order = 7, p-value = 0.01
alternative hypothesis: stationary
```

Interprétation

- Sur la série originale, p-value $= 0.97 > 0.05$, série non stationnaire
- Sur la série différenciée, p-value $= 0.01 < 0.05$, série stationnaire

5. Analyse ACF et PACF

```r
par(mfrow=c(2,1))
par(mar=c(4,4,2,2))
acf(cas_diff, main="ACF des différences premières")
pacf(cas_diff, main="PACF des différences premières")
```



Interprétation

Les fonctions d'autocorrélation (ACF) et d'autocorrélation partielle (PACF) permettent de déterminer les paramètres ppp et qqq du modèle ARIMA. Elles montrent les corrélations des données avec elles-mêmes à différents lags (décalages), aidant à comprendre si la série suit un modèle autoregressif (AR) ou un modèle de moyenne mobile (MA)

6. Modélisation ARIMA

```
modele_auto <- auto.arima(cas_diff, seasonal=TRUE)
summary(modele_auto)
```

```
Series: cas_diff
ARIMA(0,0,1)(1,0,1)[7] with zero mean

Coefficients:
          ma1    sar1     sma1
      -0.7391  0.8982  -0.6020
s.e.   0.0321  0.0284   0.0529

sigma^2 = 962.1:  log likelihood = -2092.59
AIC=4193.18   AICc=4193.27   BIC=4209.44

Training set error measures:
                      ME     RMSE      MAE MPE MAPE      MASE         ACF1
Training set -0.05961689 30.90945 20.50475 NaN  Inf 0.6203363 -0.006537767
```

Interprétation

Le modèle ARIMA sélectionné automatiquement est **ARIMA(0,0,1)(1,0,1)[7]** avec zéro moyenne. Cela signifie :

- p = 0 p = 0 p = 0 : pas de termes autoregressifs.
- d = 0 d = 0 d = 0 : pas de différenciation.
- q = 1 q = 1 q = 1 : un terme de moyenne mobile.
- Une composante saisonnière avec P=1 P= 1 P= 1, Q= 1 Q =1 Q= 1 et une fréquence de 7 (hebdomadaire).

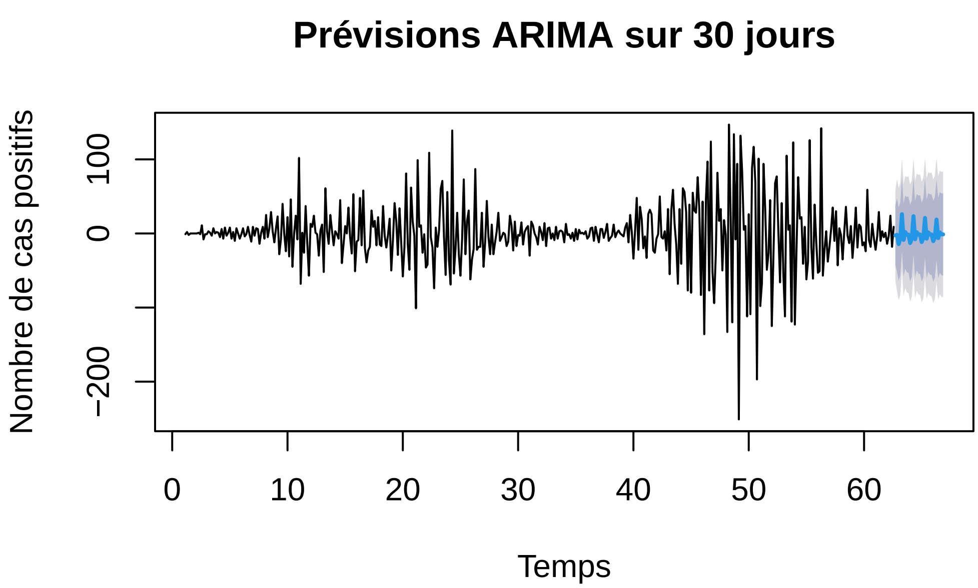
Les coefficients du modèle montrent des relations significatives, et l'erreur quadratique moyenne (RMSE) est de **30.91** sur l'ensemble d'entraînement.

---

7. Prévisions

7.1 Prévisions ARIMA sur 30 jours

```r
# Prévisions ARIMA
prev_arima <- forecast(modele_auto, h=30)
plot(prev_arima, main="Prévisions ARIMA sur 30 jours",
     xlab="Temps", ylab="Nombre de cas positifs")
```

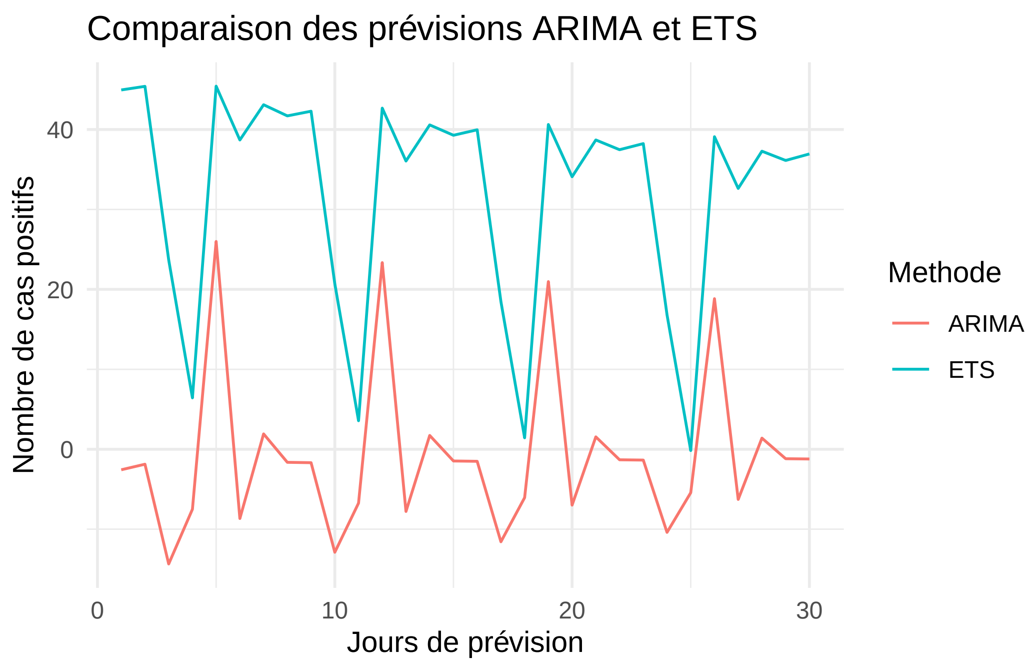


Interprétation

Les prévisions ARIMA montrent l'évolution des cas positifs sur les 30 jours suivants, avec des bandes de confiance. Ces bandes deviennent plus larges au fil du temps, indiquant une incertitude croissante dans les prévisions à mesure que l'horizon de prévision s'allonge.

7.2 Prévisions par LISSAGE EXPONENTIEL ETS (Error, Trend, Seasonality) sur 30 jours

```r
# Modèle ETS
modele_ets <- ets(cas_ts)
prev_ets <- forecast(modele_ets, h=30)

# Préparation des données pour la comparaison
df_comparaison <- data.frame(
    Date = 1:30,
    ARIMA = as.numeric(prev_arima$mean),
    ETS = as.numeric(prev_ets$mean)
)

df_long <- pivot_longer(df_comparaison,
                       cols = c("ARIMA", "ETS"),
                       names_to = "Methode",
                       values_to = "Prevision")

# Visualisation comparative
ggplot(df_long, aes(x = Date, y = Prevision, color = Methode)) +
    geom_line() +
    labs(title = "Comparaison des prévisions ARIMA et ETS",
         x = "Jours de prévision",
         y = "Nombre de cas positifs") +
    theme_minimal()
```



Interprétation

- Le modèle ETS, qui prend en compte les composantes d'erreur, de tendance et de saisonnalité, fournit également des prévisions sur 30 jours. Il est souvent plus flexible pour des données saisonnières ou avec une tendance non linéaire.
- La comparaison graphique des prévisions ARIMA et ETS montre des différences potentielles dans la manière dont les deux modèles anticipent l'évolution des cas. Il est important de noter que chaque modèle peut mieux capturer certains aspects des données (ARIMA pour la dynamique temporelle autoregressive et ETS pour la détection des tendances/saisons complexes).

8. Évaluation de la précision des modèles

```r
print("Précision du modèle ARIMA :")
```

```
[1] "Précision du modèle ARIMA :"
```

```r
accuracy(prev_arima)
```

```
                      ME     RMSE      MAE MPE MAPE      MASE         ACF1
Training set -0.05961689 30.90945 20.50475 NaN  Inf 0.6203363 -0.006537767
```

```r
print("Précision du modèle ETS :")
```

```
[1] "Précision du modèle ETS :"
```

```r
accuracy(prev_ets)
```

```
                    ME     RMSE      MAE MPE MAPE      MASE       ACF1
Training set 0.2317715 31.35175 21.72128 NaN  Inf 0.8129221 0.03148064
```

Interprétation

- **ARIMA** : Le RMSE (Root Mean Square Error) est de **30.91** et le MAE (Mean Absolute Error) de **20.50**. Cela signifie que le modèle ARIMA a une erreur moyenne relativement faible.
- **ETS** : Le RMSE est de **31.35** et le MAE de **21.72**. Légèrement plus élevé que celui du modèle ARIMA, indiquant que le modèle ETS est un peu moins précis que l'ARIMA pour cette série temporelle spécifique.
- **Performance Comparée** : Dans l'ensemble, le modèle **ARIMA** semble avoir une meilleure précision que le modèle **ETS** en raison de son RMSE et MAE inférieurs, ce qui suggère qu'il pourrait être préférable pour ces données spécifiques.