Séries temporelles univariées

M1 ECAP – TD4 – Année 2024/2025

TD4 : Analyse de la non-stationarité et racines unitaires

Responsable d'enseignement : Benoît SÉVI
benoit.sevi@univ-nantes.fr

HOUSSAIS Rémi, QUINTIN DE KERCADIO Pierre

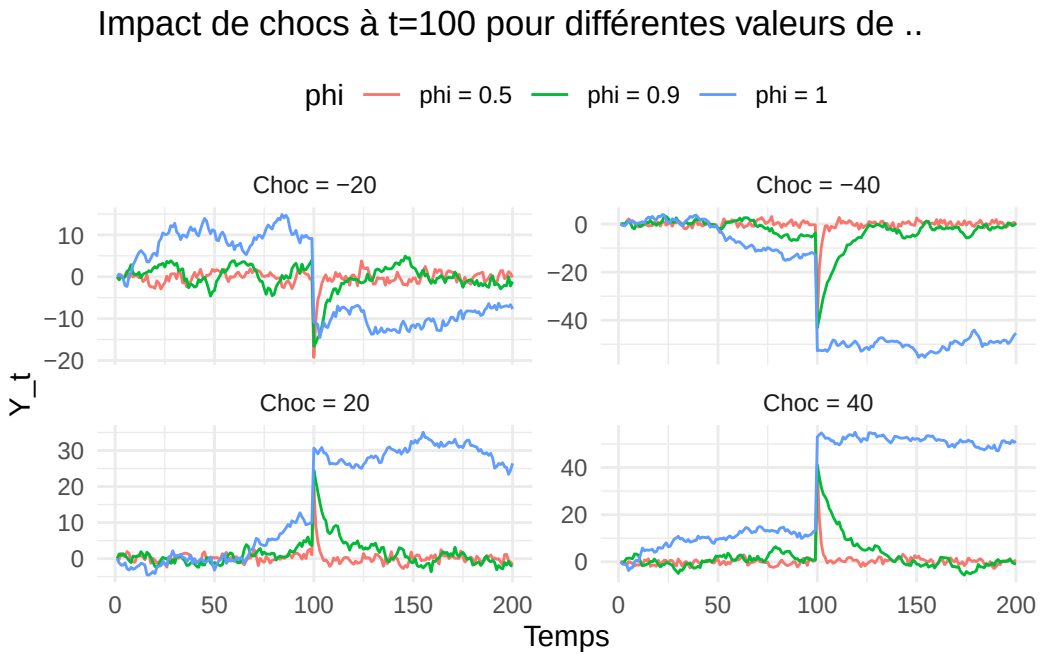
April 1, 2025

Table of contents

1 Chocs permanents vs. choc transitoires: une analyse exploratoire

```r
library(stats)
library(ggplot2)


simulate_ar1 <- function(phi, choc_value, choc_time = 100, n = 200) {
  Y <- numeric(n)
  eps <- rnorm(n, mean = 0, sd = 1)

  for (t in 2:n) {
    Y[t] <- phi * Y[t - 1] + eps[t]
    if (t == choc_time) {
      Y[t] <- Y[t] + choc_value
    }
  }
  return(Y)
}

set.seed(123)


phi_vals <- c(0.5, 0.9, 1.0)
chocs <- c(20, 40, -20, -40)

# Stocker les résultats dans une data.frame
results <- data.frame()

for (choc in chocs) {
  for (phi in phi_vals) {
    serie <- simulate_ar1(phi, choc)
    df <- data.frame(
      t = 1:200,
      Y = serie,
      phi = paste0("phi = ", phi),
      choc = paste0("Choc = ", choc)
    )
    results <- rbind(results, df)
  }
}
ggplot(results, aes(x = t, y = Y, color = phi)) +
  geom_line() +
  facet_wrap(~choc, scales = "free_y") +
  labs(title = "Impact de chocs à t=100 pour différentes valeurs de  ",
       x = "Temps", y = "Y_t") +
```

```
  theme_minimal() +
  theme(legend.position = "top")
```




```
library(ggplot2)
simulate_ar1 <- function(phi, choc_value, choc_time = 100, n = 200) {
  Y <- numeric(n)
  eps <- rnorm(n, mean = 0, sd = 1)

  for (t in 2:n) {
    Y[t] <- phi * Y[t - 1] + eps[t]
    if (t == choc_time) {
      Y[t] <- Y[t] + choc_value
    }
  }
  return(Y)
}
```

Interprétation

Lorsque ϕ=0.5 et ϕ=0.5 :

Un choc de +20 ou -20 entraîne une perturbation temporaire, mais la série retrouve rapidement son niveau d'avant-choc, en environ 10 unités de temps. Cela s'explique par le fait qu'un faible signifie une mémoire courte : les valeurs passées influencent peu les valeurs futures, et l'effet du choc s'estompe rapidement.

Lorsque ϕ=0.9 ϕ=0.9 :

Pour un même choc, le retour à l'état initial est bien plus lent, prenant environ 25 unités de temps. Avec une valeur plus élevée de , la série possède une mémoire plus longue, ce qui signifie que les effets du choc persistent plus longtemps avant de s'atténuer.

Lorsque ϕ=1.0 ϕ=1.0 (marche aléatoire) :

La série ne revient pas à une moyenne stable après le choc. Une fois l'impact survenu, la trajectoire du processus est modifiée de façon durable. Cela illustre bien le fait qu'un AR(1) avec =1 =1 est non stationnaire : il ne possède pas de force de rappel vers une moyenne fixe, et l'effet du choc se propage indéfiniment avec des fluctuations autour du nouveau niveau atteint.

Enfin, lorsque l'intensité du choc passe de 20 à 40 (ou de -20 à -40), les écarts par rapport à l'état initial sont plus marqués, et le temps nécessaire pour s'en rapprocher à nouveau augmente. Cela confirme que plus le choc est fort, plus le processus aura du mal à retrouver son équilibre, surtout pour des valeurs élevées de .

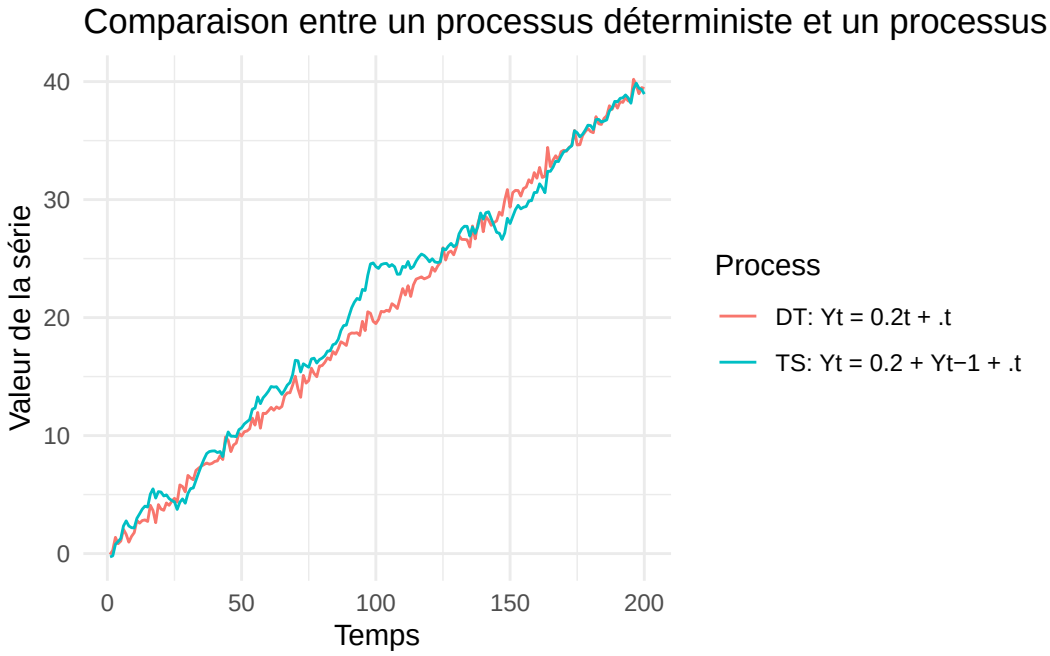
2 TS vs DS : simulation de processus

2.1 Loi $N(0,1/4)$

2.1.1 Seed(123)

```r
library(ggplot2)
set.seed(123)

n <- 200
sigma2 <- 1/4

epsilon <- rnorm(n, mean = 0, sd = sqrt(sigma2))  # Génération du bruit blanc


Y_DT <- numeric(n)
Y_TS <- numeric(n)

for (t in 1:n) {
  Y_DT[t] <- 0.2 * t + epsilon[t]
  if (t == 1) {
    Y_TS[t] <- epsilon[t]
  } else {
    Y_TS[t] <- 0.2 + Y_TS[t - 1] + epsilon[t]
  }
}

# Création du DataFrame
data <- data.frame(
  Time = rep(1:n, 2),
  Value = c(Y_DT, Y_TS),
  Process = rep(c("DT: Yt = 0.2t +  t", "TS: Yt = 0.2 + Yt-1 +  t"), each = n)
)
```

```r
# Graphique
ggplot(data, aes(x = Time, y = Value, color = Process)) +
  geom_line() +
  labs(title = "Comparaison entre un processus déterministe et un processus stochasti
       x = "Temps",
       y = "Valeur de la série") +
  theme_minimal()
```



Interprétation

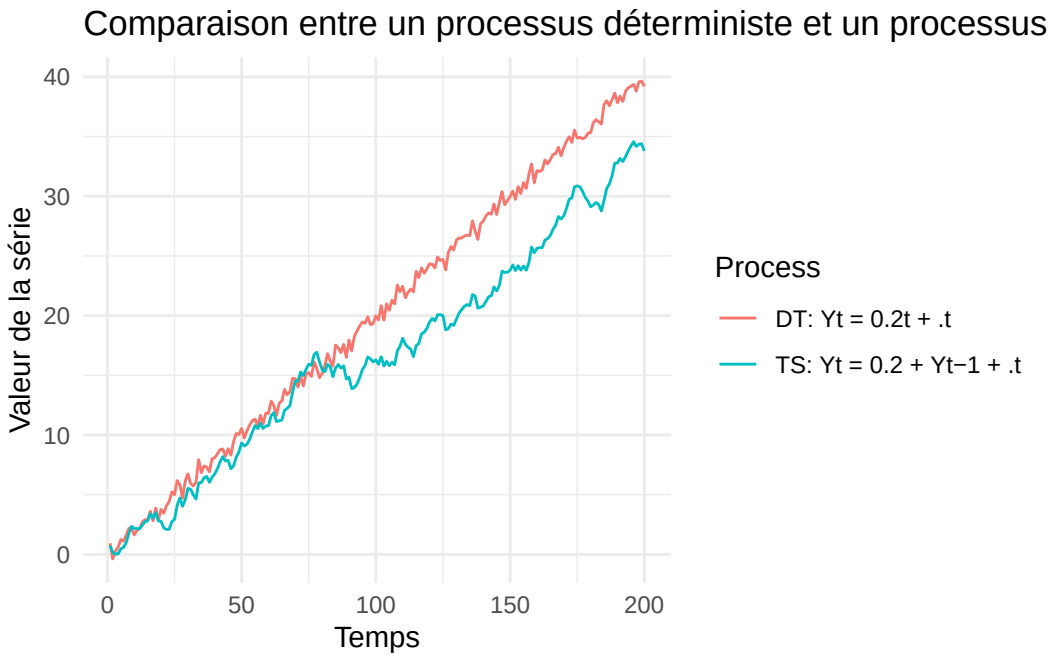
Le graphique compare deux séries soumises à un même bruit N(0,1/4): un processus déterministe ($Y_t = 0.2t + \varepsilon_t$), et un processus stochastique ($Y_t = 0.2 + Y_{t-1} + \varepsilon_t$).

La série déterministe suit une tendance linéaire régulière, perturbée temporairement par de faibles chocs : elle est stationnaire autour d'une tendance. En revanche, la série stochastique accumule les chocs dans le temps, ce qui entraîne des écarts de plus en plus marqués : elle est non stationnaire avec des effets de chocs permanents.

2.1.2 Seed(287)

```r
set.seed(287)

n <- 200
sigma2 <- 1/4

epsilon <- rnorm(n, mean = 0, sd = sqrt(sigma2))  # Génération du bruit blanc


Y_DT <- numeric(n)
```

```r
Y_TS <- numeric(n)

for (t in 1:n) {
  Y_DT[t] <- 0.2 * t + epsilon[t]
  if (t == 1) {
    Y_TS[t] <- epsilon[t]
  } else {
    Y_TS[t] <- 0.2 + Y_TS[t - 1] + epsilon[t]
  }
}

# Création du DataFrame
data <- data.frame(
  Time = rep(1:n, 2),
  Value = c(Y_DT, Y_TS),
  Process = rep(c("DT: Yt = 0.2t +  t", "TS: Yt = 0.2 + Yt-1 +  t"), each = n)
)

# Graphique
ggplot(data, aes(x = Time, y = Value, color = Process)) +
  geom_line() +
  labs(title = "Comparaison entre un processus déterministe et un processus stochasti
       x = "Temps",
       y = "Valeur de la série") +
  theme_minimal()
```



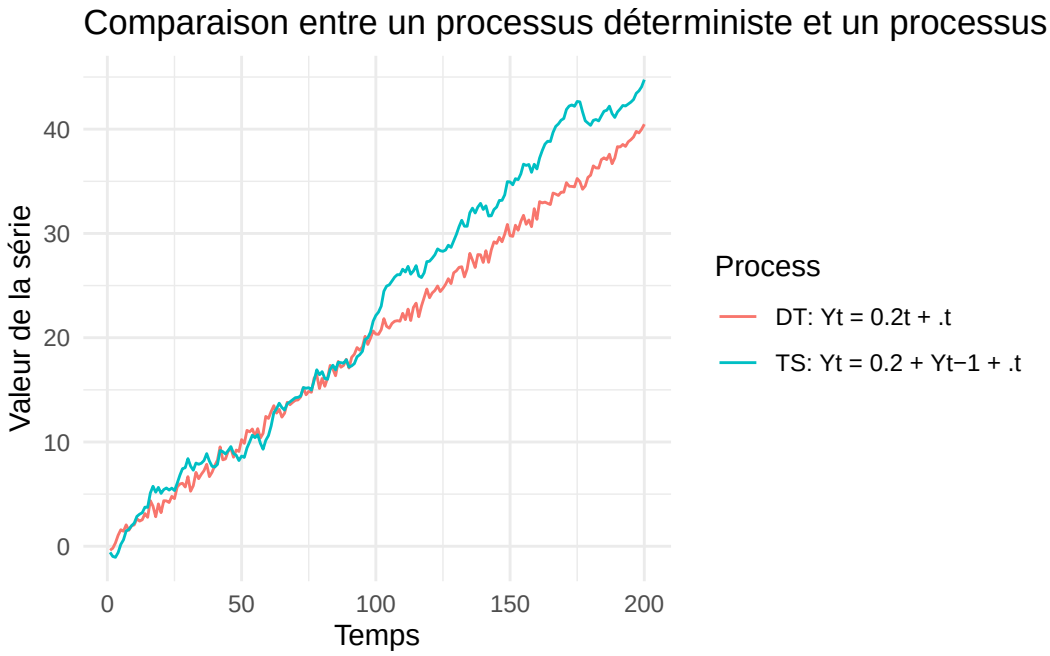
Interprétation

Avec ce nouveau tirage aléatoire, on retrouve la même dynamique générale : le processus déterministe suit une tendance régulière, alors que le processus stochastique évolue de

manière plus irrégulière. Cette fois, le DT reste au-dessus du TS sur presque toute la période, ce qui montre que la trajectoire du processus stochastique dépend fortement des chocs initiaux. Le fond reste inchangé : le DT absorbe les chocs, le TS les cumule dans le temps.

2.1.3 Seed(986)

```r
set.seed(986)

n <- 200
sigma2 <- 1/4

epsilon <- rnorm(n, mean = 0, sd = sqrt(sigma2))  # Génération du bruit blanc


Y_DT <- numeric(n)
Y_TS <- numeric(n)

for (t in 1:n) {
  Y_DT[t] <- 0.2 * t + epsilon[t]
  if (t == 1) {
    Y_TS[t] <- epsilon[t]
  } else {
    Y_TS[t] <- 0.2 + Y_TS[t - 1] + epsilon[t]
  }
}

# Création du DataFrame
data <- data.frame(
  Time = rep(1:n, 2),
  Value = c(Y_DT, Y_TS),
  Process = rep(c("DT: Yt = 0.2t +  t", "TS: Yt = 0.2 + Yt-1 +  t"), each = n)
)

# Graphique
ggplot(data, aes(x = Time, y = Value, color = Process)) +
  geom_line() +
  labs(title = "Comparaison entre un processus déterministe et un processus stochasti
       x = "Temps",
       y = "Valeur de la série") +
  theme_minimal()
```



Interprétation

Avec ce nouveau tirage aléatoire, on retrouve la même dynamique générale: le processus déterministe (rouge) suit une tendance régulière, alors que le processus stochastique (bleu) évolue de manière plus irrégulière. Cette fois, le TS dépasse nettement le DT à partir du milieu de la période, ce qui montre une nouvelle fois que la trajectoire du processus stochastique dépend fortement des chocs accumulés. Le fond reste inchangé : le DT absorbe les chocs, le TS les cumule dans le temps.

2.1.4 Conclusion

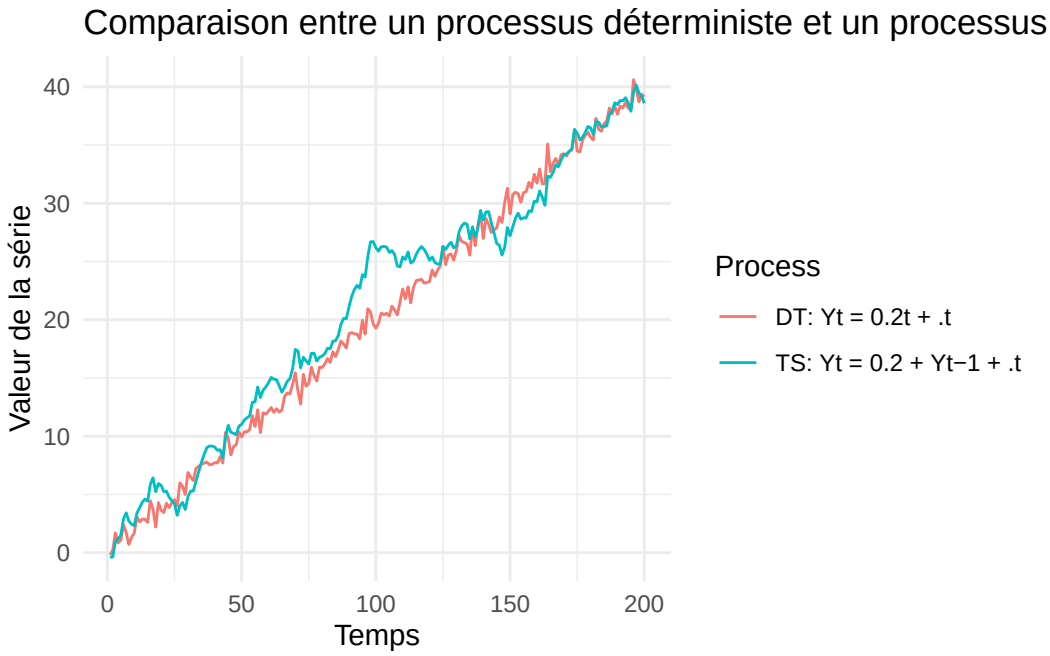
La dynamique globale reste constante : le processus déterministe suit toujours une tendance régulière perturbée temporairement par le bruit, tandis que le processus stochastique affiche une trajectoire plus irrégulière, qui s'écarte progressivement en fonction des chocs accumulés. Dans certaines simulations, le TS reste en dessous du DT, dans d'autres il le dépasse, ce qui illustre parfaitement la forte sensibilité du processus stochastique à la réalisation des chocs initiaux. Ces trois graphiques confirment que le DT est stable et prévisible, alors que le TS est instable et imprévisible à long terme, malgré des conditions de départ identiques

2.2 Loi $N(0,1/2)$

2.2.1 Seed(123)

```r
set.seed(123)

# Paramètres
n <- 200  # Nombre d'observations
```

```r
sigma2 <- 1/2  # Variance du bruit

epsilon <- rnorm(n, mean = 0, sd = sqrt(sigma2))  # Génération du bruit blanc

# Simulation des processus
Y_DT <- numeric(n)
Y_TS <- numeric(n)

for (t in 1:n) {
  Y_DT[t] <- 0.2 * t + epsilon[t]
  if (t == 1) {
    Y_TS[t] <- epsilon[t]
  } else {
    Y_TS[t] <- 0.2 + Y_TS[t - 1] + epsilon[t]
  }
}

# Création du DataFrame
data <- data.frame(
  Time = rep(1:n, 2),
  Value = c(Y_DT, Y_TS),
  Process = rep(c("DT: Yt = 0.2t +  t", "TS: Yt = 0.2 + Yt-1 +  t"), each = n)
)

# Graphique
ggplot(data, aes(x = Time, y = Value, color = Process)) +
  geom_line() +
  labs(title = "Comparaison entre un processus déterministe et un processus stochasti
       x = "Temps",
       y = "Valeur de la série") +
  theme_minimal()
```



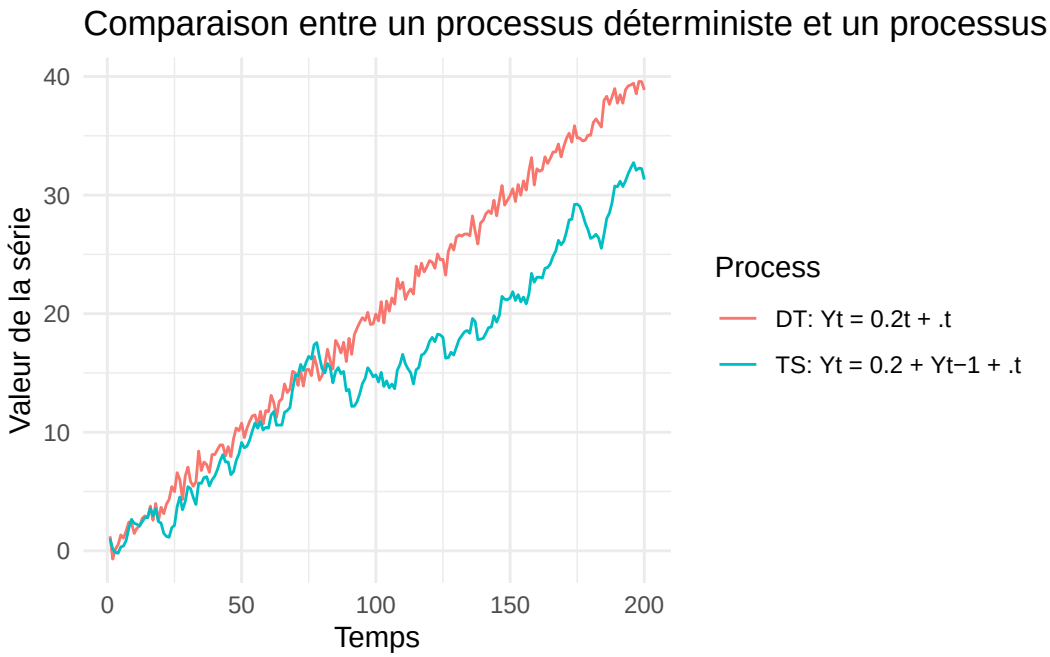
Interprétation

Avec ce nouveau tirage aléatoire et une variance du bruit plus élevée, on observe une dynamique toujours similaire : le processus déterministe progresse régulièrement selon sa tendance linéaire, tandis que le processus stochastique présente une trajectoire plus irrégulière. Ici, les deux séries restent proches sur l'ensemble de l'échantillon, mais on note que le TS dépasse légèrement le DT en fin de période. Cela montre que l'augmentation de la variance amplifie la variabilité des deux séries, mais affecte davantage le TS qui accumule les chocs. Le DT, lui, reste globalement stable autour de sa pente.

2.2.2 Seed(287)

```r
set.seed(287)

# Paramètres
n <- 200  # Nombre d'observations
sigma2 <- 1/2  # Variance du bruit

epsilon <- rnorm(n, mean = 0, sd = sqrt(sigma2))  # Génération du bruit blanc

# Simulation des processus
Y_DT <- numeric(n)
Y_TS <- numeric(n)

for (t in 1:n) {
  Y_DT[t] <- 0.2 * t + epsilon[t]
  if (t == 1) {
    Y_TS[t] <- epsilon[t]
  } else {
```

```r
    Y_TS[t] <- 0.2 + Y_TS[t - 1] + epsilon[t]
  }
}

# Création du DataFrame
data <- data.frame(
  Time = rep(1:n, 2),
  Value = c(Y_DT, Y_TS),
  Process = rep(c("DT: Yt = 0.2t +  t", "TS: Yt = 0.2 + Yt-1 +  t"), each = n)
)

# Graphique
ggplot(data, aes(x = Time, y = Value, color = Process)) +
  geom_line() +
  labs(title = "Comparaison entre un processus déterministe et un processus stochasti
       x = "Temps",
       y = "Valeur de la série") +
  theme_minimal()
```

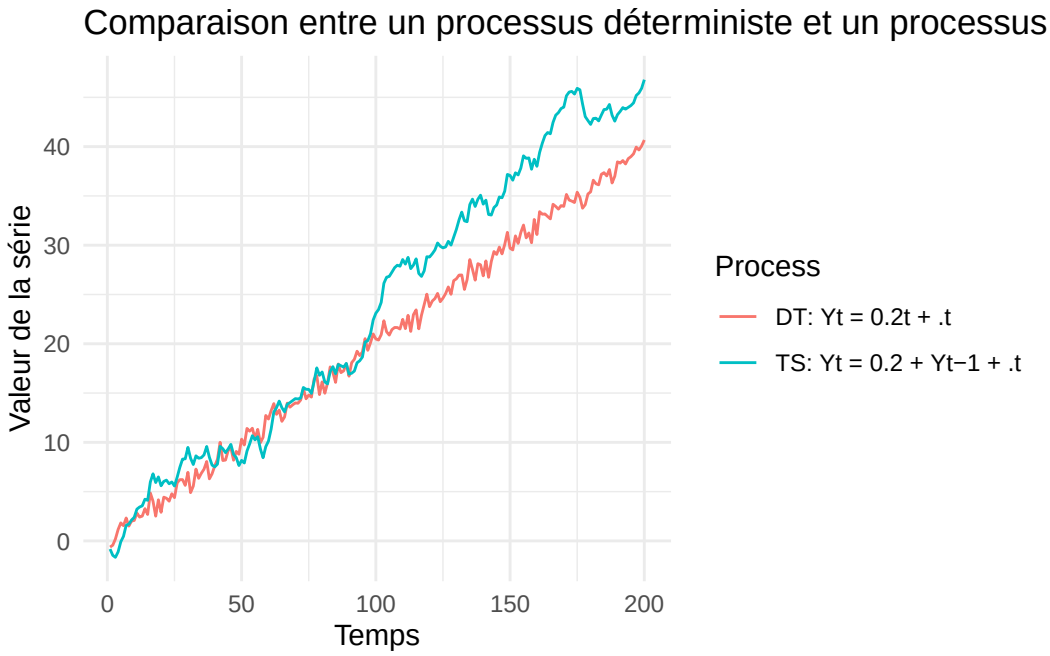


Interprétation

Avec ce nouveau tirage, on observe une configuration différente : le processus déterministe dépasse clairement le processus stochastique sur presque toute la période. Cela illustre à nouveau que, bien que les deux séries partagent la même pente moyenne (0.2), le comportement du TS est fortement influencé par les chocs initiaux. Ici, ces chocs ont freiné la progression de la série stochastique, tandis que la série déterministe, insensible à la mémoire, poursuit une trajectoire plus régulière. Le contraste est net en fin de période, où le DT atteint des niveaux beaucoup plus élevés que le TS.

2.2.3 Seed(986)

```r
set.seed(986)

# Paramètres
n <- 200  # Nombre d'observations
sigma2 <- 1/2  # Variance du bruit

epsilon <- rnorm(n, mean = 0, sd = sqrt(sigma2))  # Génération du bruit blanc

# Simulation des processus
Y_DT <- numeric(n)
Y_TS <- numeric(n)

for (t in 1:n) {
  Y_DT[t] <- 0.2 * t + epsilon[t]
  if (t == 1) {
    Y_TS[t] <- epsilon[t]
  } else {
    Y_TS[t] <- 0.2 + Y_TS[t - 1] + epsilon[t]
  }
}

# Création du DataFrame
data <- data.frame(
  Time = rep(1:n, 2),
  Value = c(Y_DT, Y_TS),
  Process = rep(c("DT: Yt = 0.2t +  t", "TS: Yt = 0.2 + Yt-1 +  t"), each = n)
)

# Graphique
ggplot(data, aes(x = Time, y = Value, color = Process)) +
  geom_line() +
  labs(title = "Comparaison entre un processus déterministe et un processus stochasti
       x = "Temps",
       y = "Valeur de la série") +
  theme_minimal()
```



Interprétation

Avec ce troisième tirage, le processus stochastique est nettement au dessus sur le déterministe dès la moitié de la période. On observe ici une forte montée du TS, liée à une succession de chocs positifs qui, une fois cumulés, entraînent un écart important par rapport au DT. À l'inverse, le processus déterministe reste fidèle à sa tendance linéaire, avec des fluctuations modérées. Cette configuration montre à quel point le processus stochastique peut diverger rapidement selon les chocs, même si les deux séries partagent la même pente moyenne.

2.2.4 Conclusion

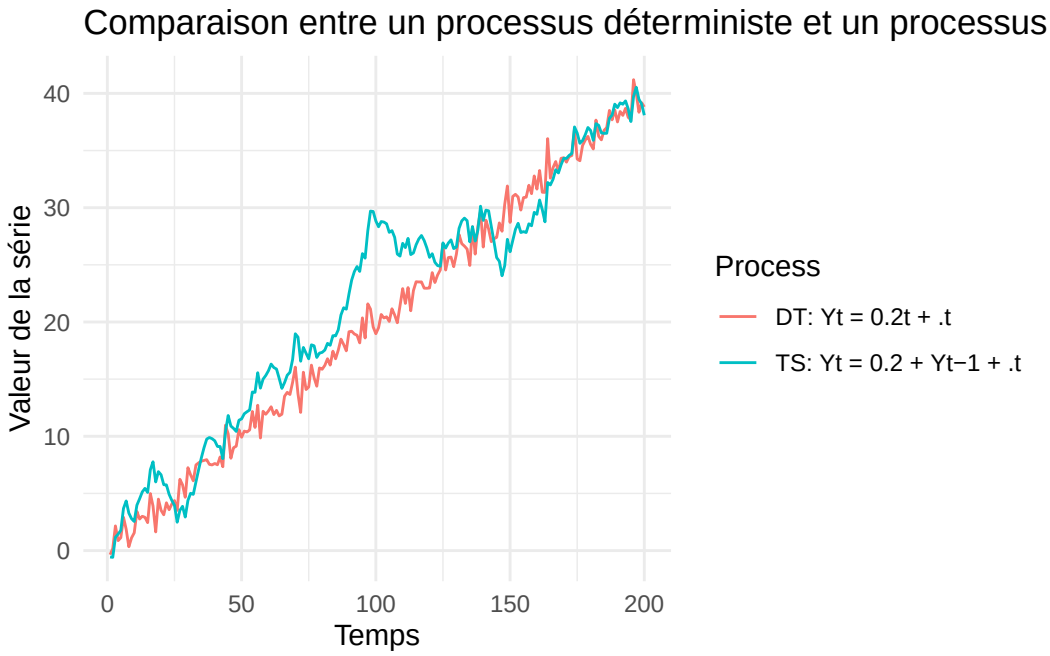
Pour une loi $N(0,1/2)$ nous pouvons remarquer que la dynamique reste constante comme sur la loi $N(0,1/4)$. En effet, le processus déterministe suit toujours une tendance régulière, seulement perturbée de manière temporaire par un bruit modéré. En revanche, le processus stochastique présente une trajectoire plus irrégulière, qui s'écarte progressivement à mesure que les chocs s'accumulent dans le temps. Dans certaines simulations, le TS reste en dessous du DT, dans d'autres il le dépasse nettement, ce qui illustre parfaitement la forte sensibilité du processus stochastique à la réalisation des chocs aléatoires. Ces trois graphiques confirment que le DT est stable, tandis que le TS reste instable.

2.3 Loi $N(0,1)$

2.3.1 Seed(123)

```
set.seed(123)

# Paramètres
```

```r
n <- 200  # Nombre d'observations
sigma2 <- 1  # Variance du bruit

epsilon <- rnorm(n, mean = 0, sd = sqrt(sigma2))  # Génération du bruit blanc

# Simulation des processus
Y_DT <- numeric(n)
Y_TS <- numeric(n)

for (t in 1:n) {
  Y_DT[t] <- 0.2 * t + epsilon[t]
  if (t == 1) {
    Y_TS[t] <- epsilon[t]
  } else {
    Y_TS[t] <- 0.2 + Y_TS[t - 1] + epsilon[t]
  }
}

# Création du DataFrame
data <- data.frame(
  Time = rep(1:n, 2),
  Value = c(Y_DT, Y_TS),
  Process = rep(c("DT: Yt = 0.2t +  t", "TS: Yt = 0.2 + Yt-1 +  t"), each = n)
)

# Graphique
ggplot(data, aes(x = Time, y = Value, color = Process)) +
  geom_line() +
  labs(title = "Comparaison entre un processus déterministe et un processus stochasti
       x = "Temps",
       y = "Valeur de la série") +
  theme_minimal()
```



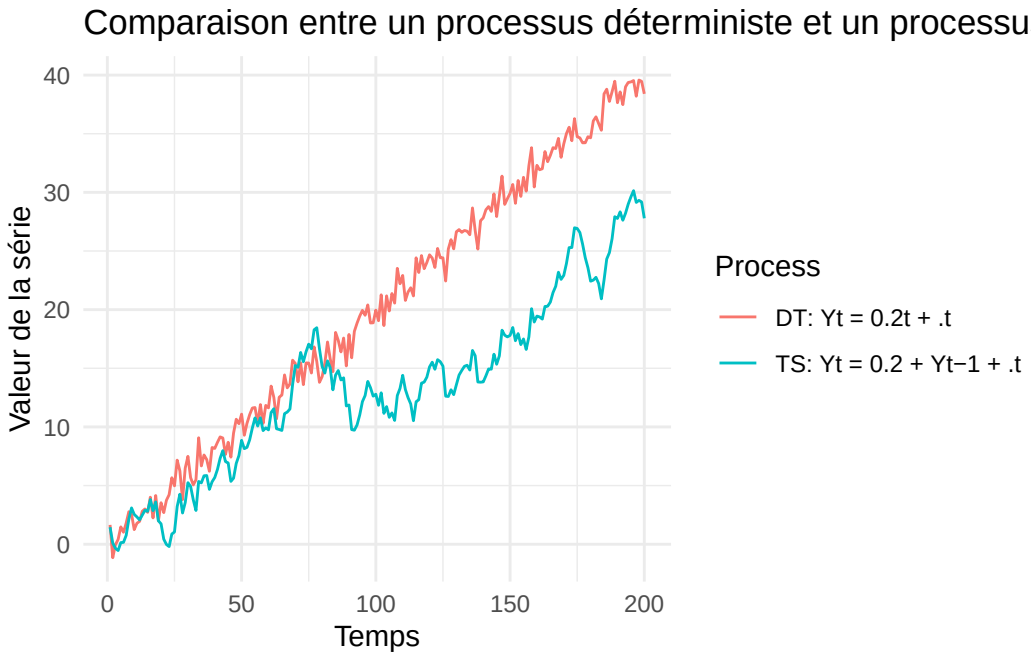
Interprétation

Pour cette loi $N(0,1)$, la dynamique reste la même observée que les deux autres lois. En effet,le processus déterministe suit une trajectoire croissante régulière, alors que le processus stochastique fluctue davantage et s'en écarte par moments. Ici, le TS dépasse nettement le DT autour de t=100. avant de redescendre et de se rapprocher en fin de période.

2.3.2 Seed(287)

```r
set.seed(287)

# Paramètres
n <- 200  # Nombre d'observations
sigma2 <- 1  # Variance du bruit

epsilon <- rnorm(n, mean = 0, sd = sqrt(sigma2))  # Génération du bruit blanc

# Simulation des processus
Y_DT <- numeric(n)
Y_TS <- numeric(n)

for (t in 1:n) {
  Y_DT[t] <- 0.2 * t + epsilon[t]
  if (t == 1) {
    Y_TS[t] <- epsilon[t]
  } else {
    Y_TS[t] <- 0.2 + Y_TS[t - 1] + epsilon[t]
  }
```

```r
}

# Création du DataFrame
data <- data.frame(
  Time = rep(1:n, 2),
  Value = c(Y_DT, Y_TS),
  Process = rep(c("DT: Yt = 0.2t +  t", "TS: Yt = 0.2 + Yt-1 +  t"), each = n)
)

# Graphique
ggplot(data, aes(x = Time, y = Value, color = Process)) +
  geom_line() +
  labs(title = "Comparaison entre un processus déterministe et un processus stochasti
       x = "Temps",
       y = "Valeur de la série") +
  theme_minimal()
```



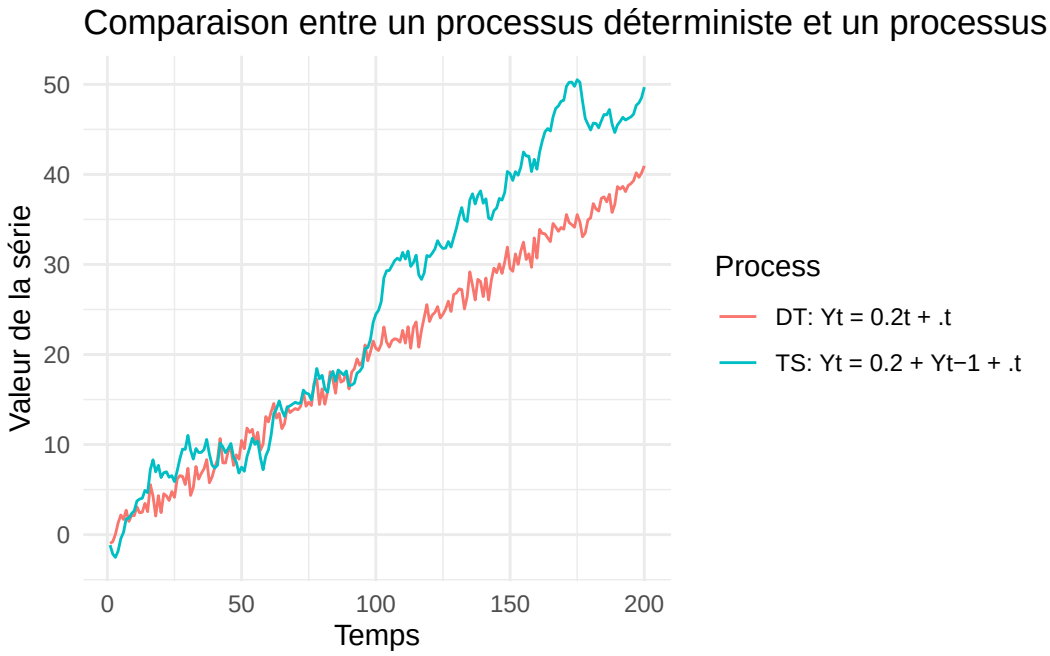
Interprétation

Avec ce nouveau tirage aléatoire, ,nous pouvons remarquer le processus déterministe est toujours stable dans le temps et est au dessus du stochastique sur l'ensemble de la période. Nous pouvons remarquer, que à partiur de T=90, TS chute brutalement et reste en dessous du processus déterministe.

2.3.3 Seed(986)

```r
set.seed(986)

# Paramètres
```

```r
n <- 200  # Nombre d'observations
sigma2 <- 1  # Variance du bruit

epsilon <- rnorm(n, mean = 0, sd = sqrt(sigma2))  # Génération du bruit blanc

# Simulation des processus
Y_DT <- numeric(n)
Y_TS <- numeric(n)

for (t in 1:n) {
  Y_DT[t] <- 0.2 * t + epsilon[t]
  if (t == 1) {
    Y_TS[t] <- epsilon[t]
  } else {
    Y_TS[t] <- 0.2 + Y_TS[t - 1] + epsilon[t]
  }
}

# Création du DataFrame
data <- data.frame(
  Time = rep(1:n, 2),
  Value = c(Y_DT, Y_TS),
  Process = rep(c("DT: Yt = 0.2t +  t", "TS: Yt = 0.2 + Yt-1 +  t"), each = n)
)

# Graphique
ggplot(data, aes(x = Time, y = Value, color = Process)) +
  geom_line() +
  labs(title = "Comparaison entre un processus déterministe et un processus stochasti
       x = "Temps",
       y = "Valeur de la série") +
  theme_minimal()
```



Interprétation

Avec ce nouveau tirage aléatoire, ,nous pouvons remarquer le processus déterministe est toujours stable dans le temps et est en dessous du stochastique sur l'ensemble de la période. Nous pouvons remarquer, que à partiur de T=90, TS monte brutalement et reste au dessus du processus déterministe.

2.3.4 Conclusion

Pour une loi $N(0,1)$, la dynamique globale reste constante : le processus déterministe suit toujours une tendance régulière perturbée temporairement par le bruit, tandis que le processus stochastique affiche une trajectoire plus irrégulière, qui s'écarte progressivement en fonction des chocs accumulés. Avec une variance plus élevée, ces écarts deviennent encore plus marqués. Le TS peut rester en retrait comme dans le deuxième graphique, coller au DT comme dans le premier, ou bien le dépasser largement comme dans le troisième. Cette variabilité illustre parfaitement la forte sensibilité du processus stochastique à la réalisation des chocs, qui s'amplifie avec la taille des perturbations.

2.4 Conlusion générale

Pour conclure sur cette partie, nous pouvons remarquer que pour chaque série c'était plus ou moins la m^meme tendance. En effet, le processus déterministe a toujours suivi la tendnace quelque soit la loi N ou le seed, et le processus stochastique a toujours été plus sensible aux fluctuations.

3 Régressions fallacieuses

3.1 $n = 200$ et $N = 5000$

```
set.seed(123)


n <- 200
N <- 5000
rejets <- numeric(N)

for (i in 1:N) {

  X <- cumsum(rnorm(n))
  Y <- cumsum(rnorm(n))


  model <- lm(Y ~ X)
  p_value <- summary(model)$coefficients[2, 4]


  rejets[i] <- ifelse(p_value < 0.05, 1, 0)
}

pourcentage_rejets <- mean(rejets) * 100
cat("Pourcentage de rejets de H0 ( 1 = 0) au seuil de 5% :", round(pourcentage_rejets,
```

```
Pourcentage de rejets de H0 ( 1 = 0) au seuil de 5% : 83.52 %
```

Interprétation Avec ce premeir test avec n=200 et N=500, nous obtenons un taux de rejet de HO de 83.52%, par conséquent même si les séries sont indépendantes, nous rejetons HO très souvent. Cela s'explique par le fait que les marches aléatoires sont non stationnaires, en raison de la présence d'une racine unitaire dans leur dynamique.

Pour pouvoir observer les differnces nousn allons varier n et N

3.2 Variations du nombre d'observations,n, et du nombre de séries générée, N.

3.2.1 Augmentation et diminution de n

3.2.1.1 $n = 50$ et $N = 5000$

```
set.seed(238)
```

```r
n <- 50
N <- 5000
rejets <- numeric(N)

for (i in 1:N) {

  X <- cumsum(rnorm(n))
  Y <- cumsum(rnorm(n))


  model <- lm(Y ~ X)
  p_value <- summary(model)$coefficients[2, 4]


  rejets[i] <- ifelse(p_value < 0.05, 1, 0)
}

pourcentage_rejets <- mean(rejets) * 100
cat("Pourcentage de rejets de H0 ( 1 = 0) au seuil de 5% :", round(pourcentage_rejets,
```

```
Pourcentage de rejets de H0 ( 1 = 0) au seuil de 5% : 66.14 %
```

Interprétation

Lorsqu'on diminue le nombre d'observation n à 50, nous pouvons remarquer que nous obtenons un taux de rejet de HO de 66.14%. Ce qui est inférieur à celui précédent. Par conséquent, quand le nombre d'observation diminue alors on rejete moins à tort.

3.2.1.2 $n = 500$ et $N = 5000$

```r
set.seed(238)


n <- 500
N <- 5000
rejets <- numeric(N)

for (i in 1:N) {

  X <- cumsum(rnorm(n))
  Y <- cumsum(rnorm(n))


  model <- lm(Y ~ X)
  p_value <- summary(model)$coefficients[2, 4]
```

```
  rejets[i] <- ifelse(p_value < 0.05, 1, 0)
}

pourcentage_rejets <- mean(rejets) * 100
cat("Pourcentage de rejets de H0 ( 1 = 0) au seuil de 5% :", round(pourcentage_rejets,
```

```
Pourcentage de rejets de H0 ( 1 = 0) au seuil de 5% : 89.2 %
```

Interprétation

Lorsqu'on augmente le nombre d'observation n à 50°, nous pouvons remarquer que nous obtenons un taux de rejet de HO de 89.2%. Ce qui est supérieur aux deux précédents. Par conséquent, quand le nombre d'observation augmente alors on rejete plus à tort.

3.2.2 Augmentation et diminution de N

3.2.2.1 $n = 200$ et $N = 2000$

```
set.seed(238)


n <- 200
N <- 2000
rejets <- numeric(N)

for (i in 1:N) {

  X <- cumsum(rnorm(n))
  Y <- cumsum(rnorm(n))


  model <- lm(Y ~ X)
  p_value <- summary(model)$coefficients[2, 4]


  rejets[i] <- ifelse(p_value < 0.05, 1, 0)
}

pourcentage_rejets <- mean(rejets) * 100
cat("Pourcentage de rejets de H0 ( 1 = 0) au seuil de 5% :", round(pourcentage_rejets,
```

```
Pourcentage de rejets de H0 ( 1 = 0) au seuil de 5% : 83.6 %
```

Interprétation

Lorsqu'on diminue le nombre de séries générées N à 2000, nous pouvons remarquer que nous obtenons un taux de rejet de HO de 83.6%. Ce qui est légérement au dessus du modèle initial.

3.2.2.2 $n = 200$ et $N = 10000$

```r
set.seed(238)


n <- 200
N <- 10000
rejets <- numeric(N)

for (i in 1:N) {

  X <- cumsum(rnorm(n))
  Y <- cumsum(rnorm(n))


  model <- lm(Y ~ X)
  p_value <- summary(model)$coefficients[2, 4]


  rejets[i] <- ifelse(p_value < 0.05, 1, 0)
}

pourcentage_rejets <- mean(rejets) * 100
cat("Pourcentage de rejets de H0 ( 1 = 0) au seuil de 5% :", round(pourcentage_rejets,
```

```
Pourcentage de rejets de H0 ( 1 = 0) au seuil de 5% : 83.49 %
```

Interprétation

Lorsqu'on augmente le nombre de séries générées N à 10000, nous pouvons remarquer que nous obtenons un taux de rejet de HO de 83.49%. Ce qui est légérement en dessous du modèle initial.

3.3 Conclusion générale

Par conséquent, le fait de changer le nombre de séries générées ne varie pas énormément le taux de rejet de H0. Alors que, le fait de changer à lla baisse ou à la hausse le nombre d'observations fait varier énormément le taux de rejet de H0

4 Distribution de la statistique de test de Dickey-Fuller pour le modèle sans constante ni tendance via la méthode de Monte Carlo

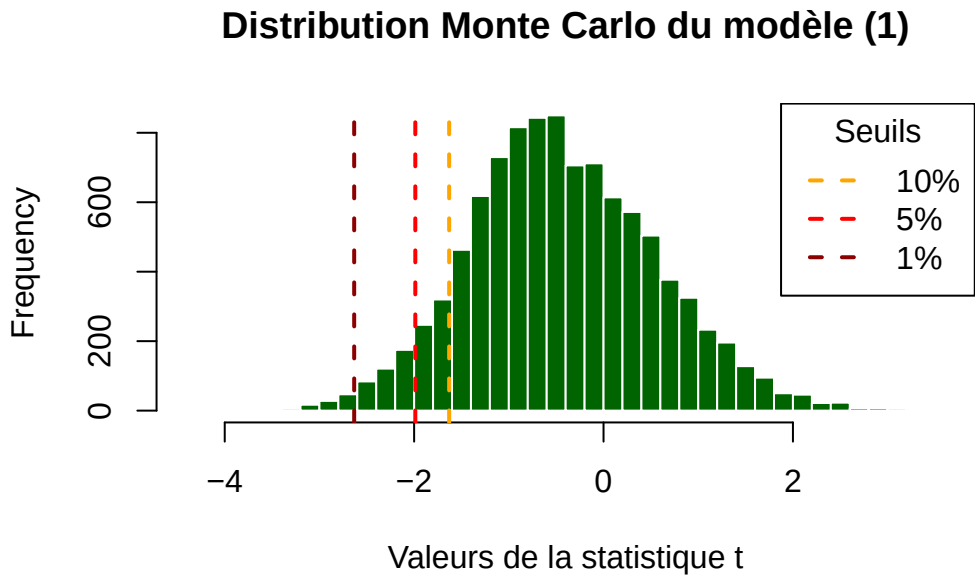
4.1 Distribution du modèle (1) sans constante ni tendance $Y_t = Y_{t-1} + \varepsilon_t$

```
set.seed(123)

# Paramètres
n <- 100          # Longueur de la série
N <- 10000        # Nombre de simulations
t_stats <- numeric(N)

for (i in 1:N) {
  eps <- rnorm(n)
  Y <- cumsum(eps)  # Y_t = Y_{t-1} +  _t

  dY <- diff(Y)             # ΔY_t
  Y_lag <- Y[-n]            # Y_{t-1}

  model <- lm(dY ~ 0 + Y_lag)  # régression sans constante
  t_stats[i] <- summary(model)$coefficients[1, 3]  # t-statistique
}

# Valeurs critiques empiriques
quantiles <- quantile(t_stats, probs = c(0.10, 0.05, 0.01))
print(round(quantiles, 3))
```

```
   10%     5%     1%
-1.630 -1.988 -2.633
```

```
# Histogramme
hist(t_stats, breaks = 50, col = "darkgreen", border = "white",
     main = "Distribution Monte Carlo du modèle (1)",
     xlab = "Valeurs de la statistique t")

# Ajout des lignes verticales pour les quantiles critiques
abline(v = quantiles[1], col = "orange", lwd = 2, lty = 2)
abline(v = quantiles[2], col = "red", lwd = 2, lty = 2)
abline(v = quantiles[3], col = "darkred", lwd = 2, lty = 2)

# Ajout d'une légende
legend("topright", legend = c("10%", "5%", "1%"),
```

```
       col = c("orange", "red", "darkred"),
       lty = 2, lwd = 2, title = "Seuils")
```



Interprétation

Pour ce modèle, sans constante ni tendance nous pouvons observer plusieurs indications. En effet, les valeurs critiques issues de la simulation du modèle (1) sont très proches de celles de la table théorique de Dickey-Fuller :

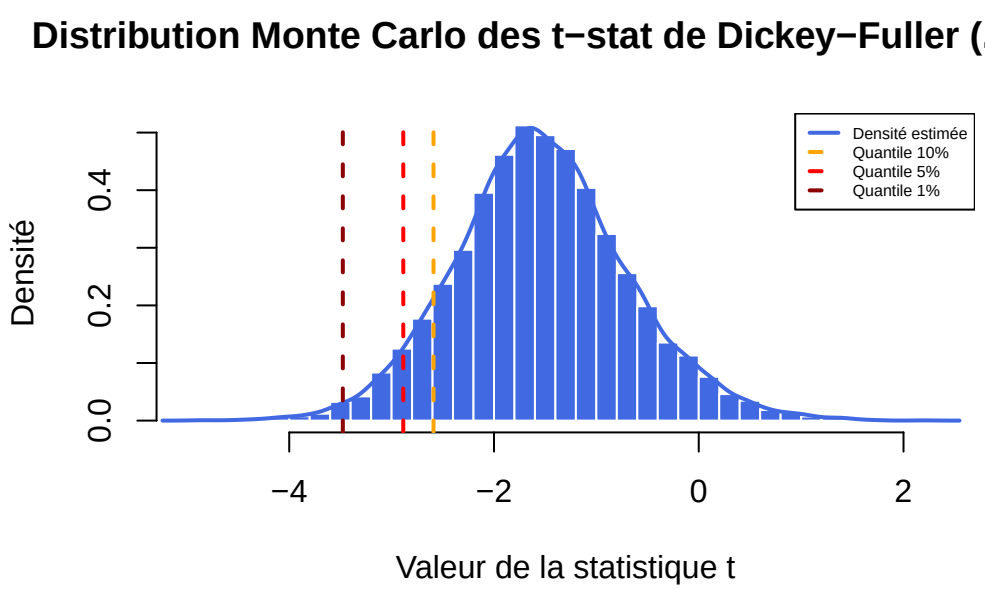
Seuil	Monte Carlo	Table DF (n = 100)
10%	-1.630	-1.61
5%	-1.988	-1.95
1%	-2.633	-2.60

En simulant 10 000 séries de 100 observations d'un processus à racine unitaire sans constante ni tendance, nous obtenons une distribution empirique de la statistique t du test de Dickey-Fuller. Les seuils de rejet à 10 %, 5 % et 1 % obtenus sont respectivement de -1.630, -1.988 et -2.633. Et lorsque l'on compare ces valeurs à celles issues de la table théorique de Dickey-Fuller, on observe une très bonne similitude avec des valeurs respectives de -1.61, -1.95 et -2.60 pour les mêmes seuils de rejet.

4.2 Distribution du modèle (2) avec $\delta_0 = 5$

```
set.seed(123)

n_simulations <- 10000
n_obs <- 100
```

```r
# Fonction pour générer une marche aléatoire avec constante
generate_random_walk_avec_const <- function(n, delta0) {
  Yt <- numeric(n)
  Yt[1] <- delta0  # Initialisation avec la constante
  for (t in 2:n) {
    Yt[t] <- Yt[t-1] + rnorm(1)
  }
  return(Yt)
}

# Fonction pour exécuter la simulation et afficher les résultats
simulation_cst <- function(delta0) {
  set.seed(123)  # Réinitialisation de la graine pour reproductibilité

  t_stats <- numeric(n_simulations)  # Réinitialisation à chaque appel

  for (i in 1:n_simulations) {
    Yt <- generate_random_walk_avec_const(n_obs, delta0)

    dYt <- diff(Yt)
    Yt_lag <- Yt[-n_obs]

    model <- lm(dYt ~ Yt_lag)  # Régression avec constante par défaut
    t_stats[i] <- summary(model)$coefficients[2, 3]  # Récupération du t-stat
  }

  # Calcul des quantiles (valeurs critiques)
  quantiles <- quantile(t_stats, c(0.10, 0.05, 0.01))

  # Affichage graphique
  hist(t_stats, breaks = 50, probability = TRUE, col = "royalblue", border = "white",
       main = paste("Distribution Monte Carlo des t-stat de Dickey-Fuller ( 0 =", delt
       xlab = "Valeur de la statistique t", ylab = "Densité")
  lines(density(t_stats), col = "royalblue", lwd = 2)
  abline(v = quantiles, col = c("orange", "red", "darkred"), lwd = 2, lty = 2)
  legend("topright", legend = c("Densité estimée", "Quantile 10%", "Quantile 5%", "Qu
         col = c("royalblue", "orange", "red", "darkred"), lwd = 2, cex = 0.5, lty =

  # Affichage des valeurs critiques
  cat("Valeurs critiques obtenues pour  0 =", delta0, ":\n")
  print(quantiles)
}


simulation_cst(delta0 = 5)
```




```
Valeurs critiques obtenues pour  0 = 5 :
      10%        5%        1%
-2.590458 -2.885931 -3.476393
```

```
#simulation_cst(delta0 = 10)
```

Interprétation

En introduisant une constante $\delta_0 = 5$ dans le processus simulé, nous observons que la distribution empirique de la statistique t est nettement déplacée vers la gauche par rapport à la distribution théorique attendue pour le test de Dickey-Fuller avec constante.

Seuil	Monte Carlo	Table DF (n = 100)
10%	-2.590	-1.61
5%	-2.885	-1.95
1%	-3.476	-2.60

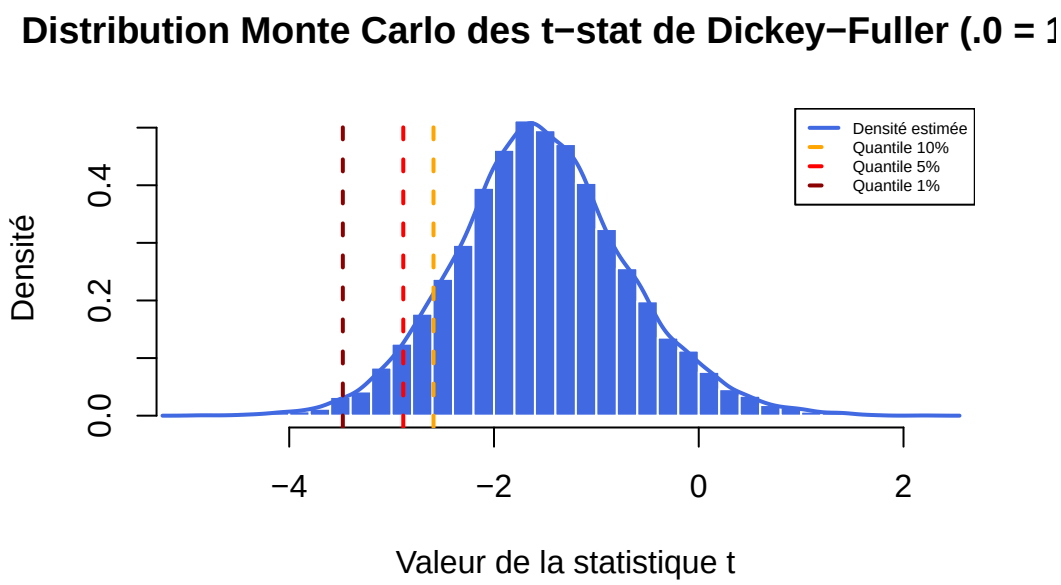
En effet, nous pouvons observer que pour $\delta_0 = 5$ les seuils de rejet s'éloignent de ceux de la table de Dickey-Fuller.

Par conséquent, d'après cette première analyse nous pouvons observer que l'instauration d'un delta est importante dans les simulations. Nous allons le vérifier avec l'instauration de delta à 10.

4.3 Distribution du modèle (3) avec une constante $\delta_0 = 10$

```r
set.seed(123)

n_simulations <- 10000
n_obs <- 100

# Fonction pour générer une marche aléatoire avec constante
generate_random_walk_avec_const <- function(n, delta0) {
  Yt <- numeric(n)
  Yt[1] <- delta0  # Initialisation avec la constante
  for (t in 2:n) {
    Yt[t] <- Yt[t-1] + rnorm(1)
  }
  return(Yt)
}

# Fonction pour exécuter la simulation et afficher les résultats
simulation_cst <- function(delta0) {
  set.seed(123)  # Réinitialisation de la graine pour reproductibilité

  t_stats <- numeric(n_simulations)  # Réinitialisation à chaque appel

  for (i in 1:n_simulations) {
    Yt <- generate_random_walk_avec_const(n_obs, delta0)

    dYt <- diff(Yt)
    Yt_lag <- Yt[-n_obs]

    model <- lm(dYt ~ Yt_lag)  # Régression avec constante par défaut
    t_stats[i] <- summary(model)$coefficients[2, 3]  # Récupération du t-stat
  }

  # Calcul des quantiles (valeurs critiques)
  quantiles <- quantile(t_stats, c(0.10, 0.05, 0.01))

  # Affichage graphique
  hist(t_stats, breaks = 50, probability = TRUE, col = "royalblue", border = "white",
       main = paste("Distribution Monte Carlo des t-stat de Dickey-Fuller ( 0 =", delt
       xlab = "Valeur de la statistique t", ylab = "Densité")
  lines(density(t_stats), col = "royalblue", lwd = 2)
  abline(v = quantiles, col = c("orange", "red", "darkred"), lwd = 2, lty = 2)
  legend("topright", legend = c("Densité estimée", "Quantile 10%", "Quantile 5%", "Qu
         col = c("royalblue", "orange", "red", "darkred"), lwd = 2, cex = 0.5, lty =

  # Affichage des valeurs critiques
  cat("Valeurs critiques obtenues pour  0 =", delta0, ":\n")
  print(quantiles)
}
```

```r
#simulation_cst(delta0 = 5)
simulation_cst(delta0 = 10)
```




```
Valeurs critiques obtenues pour  0 = 10 :
      10%        5%        1%
-2.590458 -2.885931 -3.476393
```

Interprétation

En introduisant une constante $\delta_0 = 10$ dans le processus simulé, nous observons que la distribution empirique de la statistique t est nettement déplacée vers la gauche par rapport à la distribution théorique attendue pour le test de Dickey-Fuller avec constante.

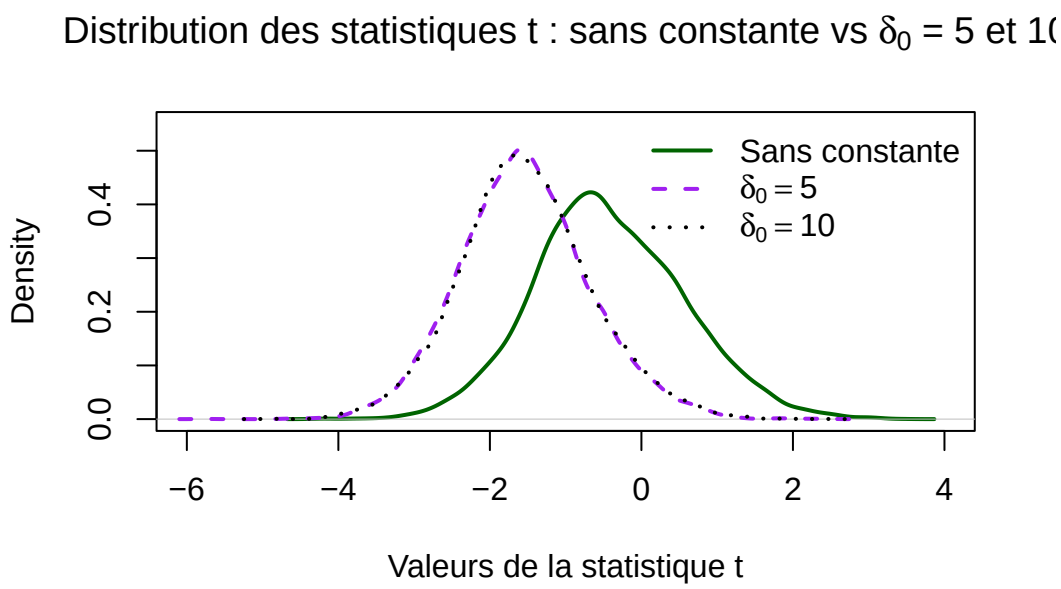
Seuil	Monte Carlo	Table DF (n = 100)
10%	-2.590	-1.61
5%	-2.885	-1.95
1%	-3.476	-2.60

Pour cette secponde simulation avec constante, nous pouvons également remarquer que le fait d'introduire une constante influence grandement grandement les résultats de la simulation.

4.4 Conclusion générale

```r
set.seed(123)

n <- 100
N <- 10000

# Initialisation
t_none <- numeric(N)
t_d5 <- numeric(N)
t_d10 <- numeric(N)

# Simulation : modèle sans constante
for (i in 1:N) {
  eps <- rnorm(n)
  Y <- cumsum(eps)
  dY <- diff(Y)
  Y_lag <- Y[-n]
  model <- lm(dY ~ 0 + Y_lag)
  t_none[i] <- summary(model)$coefficients[1, 3]
}

# Simulation : modèle avec constante   = 5
for (i in 1:N) {
  eps <- rnorm(n)
  Y <- numeric(n)
  Y[1] <- 5
  for (t in 2:n) {
    Y[t] <- Y[t - 1] + eps[t]
  }
  dY <- diff(Y)
  Y_lag <- Y[-n]
  model <- lm(dY ~ Y_lag)
  t_d5[i] <- summary(model)$coefficients[2, 3]
}

# Simulation : modèle avec constante   = 10
for (i in 1:N) {
  eps <- rnorm(n)
  Y <- numeric(n)
  Y[1] <- 10
  for (t in 2:n) {
    Y[t] <- Y[t - 1] + eps[t]
  }
  dY <- diff(Y)
  Y_lag <- Y[-n]
  model <- lm(dY ~ Y_lag)
  t_d10[i] <- summary(model)$coefficients[2, 3]
}
```

```r
# Densités
dens_none <- density(t_none)
dens_5 <- density(t_d5)
dens_10 <- density(t_d10)

# Graphique final
plot(dens_none, col = "darkgreen", lwd = 2,
     main = expression("Distribution des statistiques t : sans constante vs " * delta
     xlab = "Valeurs de la statistique t", ylab = "Density",
     xlim = c(-6, 4), ylim = c(0, 0.55))
lines(dens_5, col = "purple", lwd = 2, lty = 2)
lines(dens_10, col = "black", lwd = 2, lty = 3)

legend("topright",
       legend = c("Sans constante", expression(delta[0] == 5), expression(delta[0] ==
       col = c("darkgreen", "purple", "black"),
       lty = c(1, 2, 3), lwd = 2, bty = "n")
```



Interprétation

Concernant le graphqiue ci dessus, nous observer comme indiqué plus haut, une certaine différence entre le fait de rajouter une constante et de pas en avoir. En effet, avoir une constance entraine que la courbe se décalle vers la gauche.

Néanmoins, concernnat la différence de $\delta_0 = 5$ et $\delta_0 = 10$, il n'existe pas dedifférence visuelle. Les deux courbes se suivent.

Tableau de comparaison

Seuil	Sans constante	$\delta_0 = 5$	$\delta_0 = 10$	Table DF (n = 100)
10%	-1.630	-2.590	-2.590	-1.61
5%	-1.988	-2.885	-2.885	-1.95
1%	-2.633	-3.476	-3.476	-2.60

Comme nous pouvons l'observer avec ce tableau, il existe une réélle différence entre le fait d'avoir une constant et non. En effet, avec une constante on s'éloigne des résultats de la table de Dickey-Fuller.

Néanmoins, entre $\delta_0 = 5$ et $\delta_0 = 10$ il n'existe pas de différence donc la différence se fait suelement sur le fait d'ajouter une constante ou non.

5 Analyse de la série temporelle du PIB des USA sur la période 1990-2023

5.1 Représentation graphique du PIB US avec les zones ombrées pour les récessions

```
library(ggplot2)
library(readr)
library(dplyr)
```

```
Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
library(tibbletime)
```

```
Attaching package: 'tibbletime'

The following object is masked from 'package:stats':

    filter
```

```r
library(scales)
```

```
Attaching package: 'scales'

The following object is masked from 'package:readr':

    col_factor
```

```r
library(lubridate)
```

```
Attaching package: 'lubridate'

The following objects are masked from 'package:base':

    date, intersect, setdiff, union
```

```r
library(tseries)
```

```
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
```

```r
library(urca)
```

```r
# Charger le fichier
load("data/GDP-US-1990-2023.RData")

# Vérifier le nom de l'objet chargé
ls()
```

```
 [1] "choc"                                "chocs"
 [3] "data"                                "dens_10"
 [5] "dens_5"                              "dens_none"
 [7] "df"                                  "df_Y_us"
 [9] "dY"                                  "eps"
[11] "epsilon"                             "generate_random_walk_avec_const"
[13] "i"                                   "model"
[15] "n"                                   "N"
[17] "n_obs"                               "n_simulations"
[19] "p_value"                             "phi"
[21] "phi_vals"                            "pourcentage_rejets"
[23] "quantiles"                           "rejets"
```

```
[25] "results"                          "serie"
[27] "sigma2"                           "simulate_ar1"
[29] "simulation_cst"                   "t"
[31] "t_d10"                            "t_d5"
[33] "t_none"                           "t_stats"
[35] "X"                                "Y"
[37] "Y_DT"                             "Y_lag"
[39] "Y_TS"
```

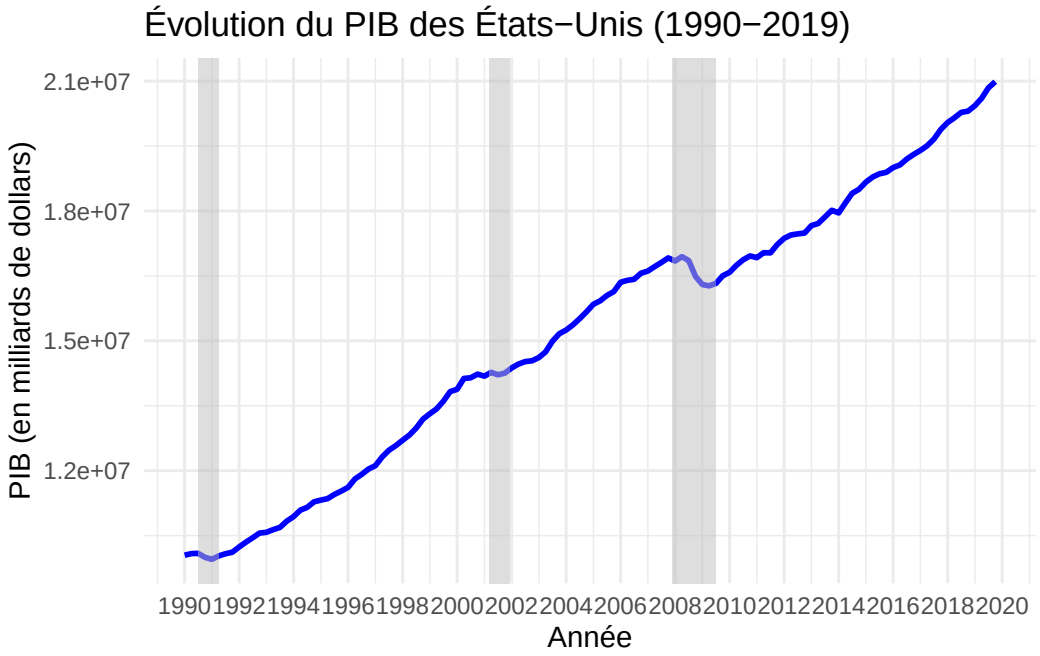
5.1.1 Définition des zonées ombrées sous-periode 1990-2019

```
df_Y_us2 <- df_Y_us[c(-121:-140),]
#str(df_Y_us2)
recessions <- data.frame(
  start = as.Date(c("1990-07-01", "2001-03-01", "2007-12-01")),
  end = as.Date(c("1991-03-31", "2001-11-30", "2009-06-30"))
)
```

5.1.2 Graphique

```
ggplot(df_Y_us2, aes(x = Date, y = PIB)) +
  geom_line(color = "blue", size = 1) + # Tracer la ligne du PIB
  # Ajouter des zones ombrées pour les récessions
  geom_rect(data = recessions, aes(xmin = start, xmax = end, ymin = -Inf, ymax = Inf)
            fill = "gray", alpha = 0.5, inherit.aes = FALSE) +
  labs(title = "Évolution du PIB des États-Unis (1990-2019)",
       x = "Année", y = "PIB (en milliards de dollars)") +
  theme_minimal() + # Style minimaliste
  scale_x_date(labels = date_format("%Y"), breaks = "2 years") # Étiquettes de l'axe
```

```
Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
i Please use `linewidth` instead.
```



Interprétation

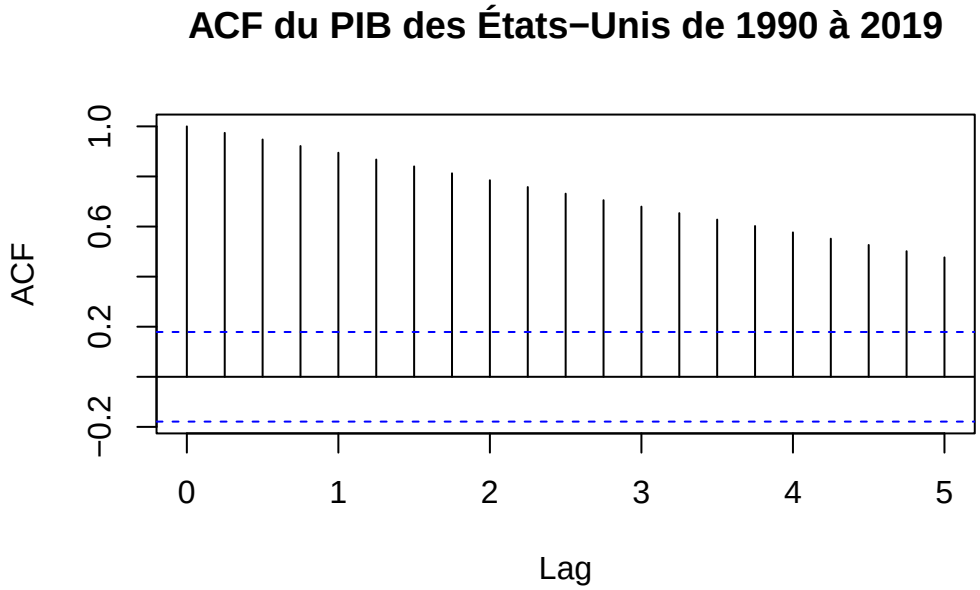
Tout d'abord, les zones grises représentent les périodes de récession aux États-Unis, telles que définies par le National Bureau of Economic Research. Elles correspondent à des phases où l'activité économique s'est contractée, entraînant un ralentissement voire une baisse du PIB. Visuellement, ces zones permettent de repérer les moments où la dynamique de croissance est interrompue ou infléchie, comme lors de la crise de 2008, qui se traduit ici par une chute marquée du PIB. Elles offrent donc un repère visuel essentiel pour relier les évolutions économiques aux chocs conjoncturels majeurs.

On remarque également que chaque zone grisée correspond à une baisse temporaire du PIB ou à une phase de stagnation, contrastant avec le reste de la période où la croissance est continue et soutenue. Cela souligne clairement l'impact des récessions sur la trajectoire de l'économie américaine.

5.2 ACF et PACF de la série

5.2.1 ACF

```
#conversion en série temporelle
ts_pib <- ts(df_Y_us2$PIB, start = c(1990, 1), frequency = 4)


acf(ts_pib, main = "ACF du PIB des États-Unis de 1990 à 2019")
```

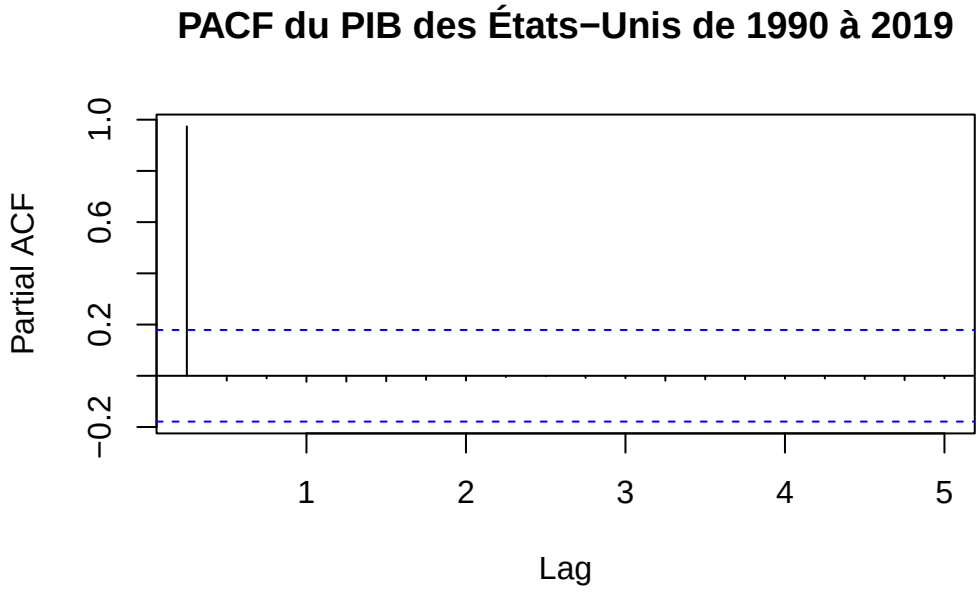


Interprétation

Le graphique de l'autocorrélation (ACF) du PIB américain sur la période 1990–2019 met en évidence une structure d'une série non stationnaire. L'autocorrélation au premier décalage est très élevée (proche de 1), et les coefficients décroissent lentement sans jamais devenir non significatifs. Cette persistance de la mémoire indique que les valeurs passées du PIB influencent fortement ses valeurs futures, ce qui est caractéristique d'un processus avec racine unitaire.

5.2.2 PACF

```
# PACF
pacf(ts_pib, main = "PACF du PIB des États-Unis de 1990 à 2019")
```



Interprétation

Le graphique de la fonction d'autocorrélation partielle (PACF) du PIB en niveau présente une structure très nette : seul le premier lag est significatif, tandis que tous les autres sont proches de zéro et non significatifs au-delà du seuil de confiance. Ce type de découplage brutal après le premier retard est typique d'un processus de type AR(1) non stationnaire, donc d'une marche aléatoire. ce graphqiue est égaleme,nt typique des bruits blancs

5.3 Mise en œuvre de la procédure de test de racine unitaire avec le test de Dickey-Fuller simple puis

le test de Dickey-Fuller augmenté.

5.3.1 Modèle 3: Avec constante et tendance

```
# Modèle 3: Avec tendance et constante
cat("\n--- TEST DF MODÈLE 3: AVEC TENDANCE ET CONSTANTE ---\n")
```

```
--- TEST DF MODÈLE 3: AVEC TENDANCE ET CONSTANTE ---
```

```
# Test de Dickey-Fuller simple
df_test_modele3 <- ur.df(ts_pib, type = "trend", lags = 0)
summary(df_test_modele3)
```

```
###############################################
```

```
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression trend


Call:
lm(formula = z.diff ~ z.lag.1 + 1 + tt)

Residuals:
    Min      1Q  Median      3Q     Max
-456929  -34007    9637   44662  176173

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.924e+05  2.076e+05   1.409    0.162
z.lag.1     -2.231e-02  2.125e-02  -1.050    0.296
tt           2.288e+03  1.927e+03   1.188    0.237

Residual standard error: 87550 on 116 degrees of freedom
Multiple R-squared:  0.02144,   Adjusted R-squared:  0.004568
F-statistic: 1.271 on 2 and 116 DF,  p-value: 0.2845


Value of test-statistic is: -1.05 44.5732 1.2707

Critical values for test statistics:
      1pct  5pct 10pct
tau3 -3.99 -3.43 -3.13
phi2  6.22  4.75  4.07
phi3  8.43  6.49  5.47
```

```r
# Test de Dickey-Fuller augmenté (avec retards sélectionnés automatiquement)
adf_test_modele3 <- ur.df(ts_pib, type = "trend", selectlags = "AIC")
summary(adf_test_modele3)
```

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression trend


Call:
lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)

Residuals:
```

```
    Min      1Q  Median      3Q     Max
-380439  -41569    3547   54152  194889

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.735e+05  1.936e+05   1.929   0.0562 .
z.lag.1     -3.357e-02  1.991e-02  -1.686   0.0945 .
tt           3.183e+03  1.804e+03   1.764   0.0803 .
z.diff.lag   3.931e-01  8.650e-02   4.544 1.38e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 81200 on 114 degrees of freedom
Multiple R-squared:  0.1698,    Adjusted R-squared:  0.148
F-statistic: 7.772 on 3 and 114 DF,  p-value: 9.097e-05


Value of test-statistic is: -1.6862 10.5836 1.6977

Critical values for test statistics:
      1pct  5pct 10pct
tau3 -3.99 -3.43 -3.13
phi2  6.22  4.75  4.07
phi3  8.43  6.49  5.47
```

Interprétation

Le test de Dickey Fuller augmenté nous confirme cela -1,6862 > -3,43, nous rejetons H_0 uniquement au seuil de 10%

Conclusion principale : la série contient toujours une racine unitaire nous pouvons passer au modèle 2.

5.3.2 Modèle 2: Avec constante et sans tendance

```r
df_test_modele2 <- ur.df(ts_pib, type = "drift", lags = 0)
summary(df_test_modele2)
```

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression drift


Call:
```

```
lm(formula = z.diff ~ z.lag.1 + 1)

Residuals:
    Min      1Q  Median      3Q     Max
-465560  -39738    5674   48368  164302

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 5.043e+04  3.990e+04   1.264    0.209
z.lag.1     2.740e-03  2.581e-03   1.062    0.291

Residual standard error: 87700 on 117 degrees of freedom
Multiple R-squared:  0.00954,   Adjusted R-squared:  0.001074
F-statistic: 1.127 on 1 and 117 DF,  p-value: 0.2906


Value of test-statistic is: 1.0615 65.9231

Critical values for test statistics:
      1pct  5pct 10pct
tau2 -3.46 -2.88 -2.57
phi1  6.52  4.63  3.81
```

```r
adf_test_modele2 <- ur.df(ts_pib, type = "drift", selectlags = "AIC")
summary(adf_test_modele2)
```

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression drift


Call:
lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)

Residuals:
    Min      1Q  Median      3Q     Max
-395744  -41688    8802   49393  193647

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 3.840e+04  3.797e+04   1.012    0.314
z.lag.1     1.294e-03  2.457e-03   0.526    0.600
z.diff.lag  3.754e-01  8.670e-02   4.330 3.21e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 81940 on 115 degrees of freedom
Multiple R-squared:  0.1471,    Adjusted R-squared:  0.1323
F-statistic: 9.919 on 2 and 115 DF,  p-value: 0.0001061


Value of test-statistic is: 0.5265 14.0606

Critical values for test statistics:
      1pct  5pct 10pct
tau2 -3.46 -2.88 -2.57
phi1  6.52  4.63  3.81
```

Interprétation

La statistique de test est supérieure à la valeur critique de -2.88 à 5%, donc nous ne rejetons pas l'hypothèse nulle de racine unitaire. Cela suggère que la série n'est pas stationnaire. Le second modèle nous amène à la même conclusion. Nous devons donc passer au modèle 1 (sans constante ni tendance).

5.3.3 Modèle 1: Sans constante et tendance

```r
#Test de Dickey-Fuller simple
    df_test_modele1 <- ur.df(ts_pib, type = "none", lags = 0)
    summary(df_test_modele1)
```

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression none


Call:
lm(formula = z.diff ~ z.lag.1 - 1)

Residuals:
    Min      1Q  Median      3Q     Max
-468991  -29485    6066   49954  170381

Coefficients:
         Estimate Std. Error t value Pr(>|t|)
z.lag.1 0.0059360  0.0005214   11.38   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Residual standard error: 87920 on 118 degrees of freedom
Multiple R-squared:  0.5234,    Adjusted R-squared:  0.5194
F-statistic: 129.6 on 1 and 118 DF,  p-value: < 2.2e-16


Value of test-statistic is: 11.3839

Critical values for test statistics:
      1pct  5pct 10pct
tau1 -2.58 -1.95 -1.62
```

```r
# Test de Dickey-Fuller augmenté
    adf_test_modele1 <- ur.df(ts_pib, type = "none", selectlags = "AIC")
    summary(adf_test_modele1)
```

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression none


Call:
lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)

Residuals:
    Min      1Q  Median      3Q     Max
-396738  -40804    8645   50635  189817

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
z.lag.1    0.0036741  0.0007059   5.205 8.48e-07 ***
z.diff.lag 0.3835047  0.0863343   4.442 2.05e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 81950 on 116 degrees of freedom
Multiple R-squared:  0.5927,    Adjusted R-squared:  0.5856
F-statistic: 84.38 on 2 and 116 DF,  p-value: < 2.2e-16


Value of test-statistic is: 5.205

Critical values for test statistics:
      1pct  5pct 10pct
tau1 -2.58 -1.95 -1.62
```

Interprétation

Les deux tests indiquent que la série possède une racine unitaire, ce qui signifie qu'elle n'est pas stationnaire en niveau. En effet la statistique de test est supérieur a l'ensemble des valeurs critique (exemple avec le DF augmenté: 5.25 > -1.95).

Peu importe le modèle testé (avec ou sans lags), l'hypothèse nulle de non-stationnarité n'est jamais rejetée. La série est donc non stationnaire, ce qui confirme ce qui a été dit avant avec l'ACF et PACF.

5.4 Test de stationarité de KPSS

```
kpss_test <- kpss.test(ts_pib, null = "Trend")
```

```
Warning in kpss.test(ts_pib, null = "Trend"): p-value smaller than printed
p-value
```

```
print(kpss_test)
```

```

    KPSS Test for Trend Stationarity

data:  ts_pib
KPSS Trend = 0.28651, Truncation lag parameter = 4, p-value = 0.01
```

Interprétation

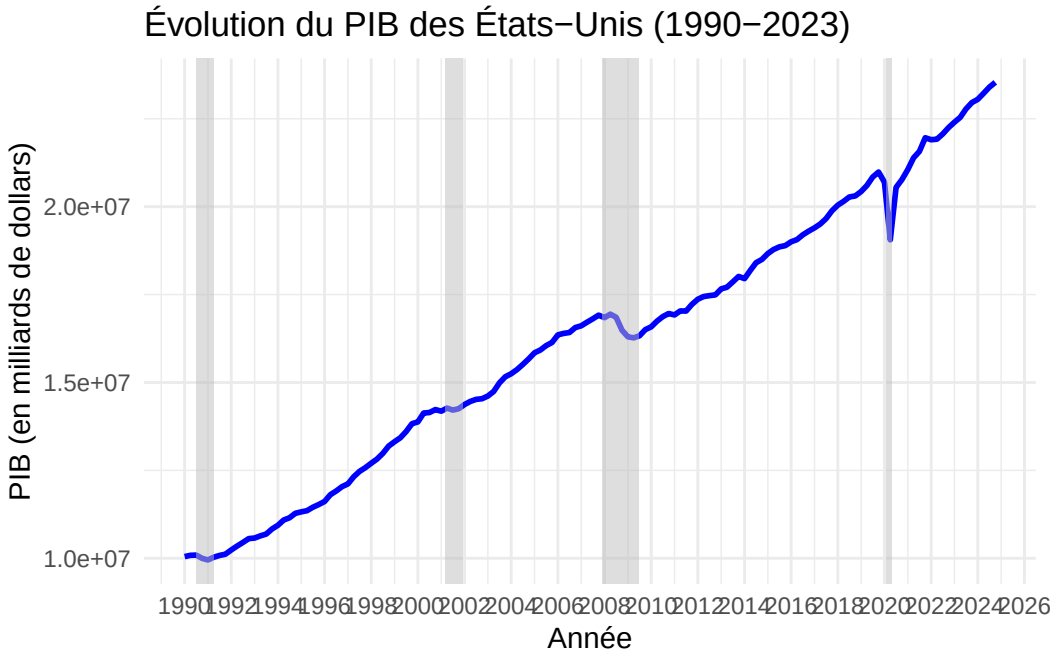
La p-value est faible ($0.01 < 0.05$), on rejette donc l'hypothèse nulle de stationnarité, la série est non stationnaire. Les deux tests confirment que la série est non stationnaire.

5.5 Analyse avec la période globale (1990-2023)

5.5.1 Graphique

```
recessions <- data.frame(
  start = as.Date(c("1990-07-01", "2001-03-01", "2007-12-01", "2020-02-01")),
  end = as.Date(c("1991-03-31", "2001-11-30", "2009-06-30", "2020-04-30"))
)


# Tracer le PIB avec les zones ombrées pour les récessions
ggplot(df_Y_us, aes(x = Date, y = PIB)) +
  geom_line(color = "blue", size = 1) + # Tracer la ligne du PIB
  # Ajouter des zones ombrées pour les récessions
  geom_rect(data = recessions, aes(xmin = start, xmax = end, ymin = -Inf, ymax = Inf)
```

```
            fill = "gray", alpha = 0.5, inherit.aes = FALSE) +
  labs(title = "Évolution du PIB des États-Unis (1990-2023)",
       x = "Année", y = "PIB (en milliards de dollars)") +
  theme_minimal() + # Style minimaliste
  scale_x_date(labels = date_format("%Y"), breaks = "2 years") # Étiquettes de l'axe
```

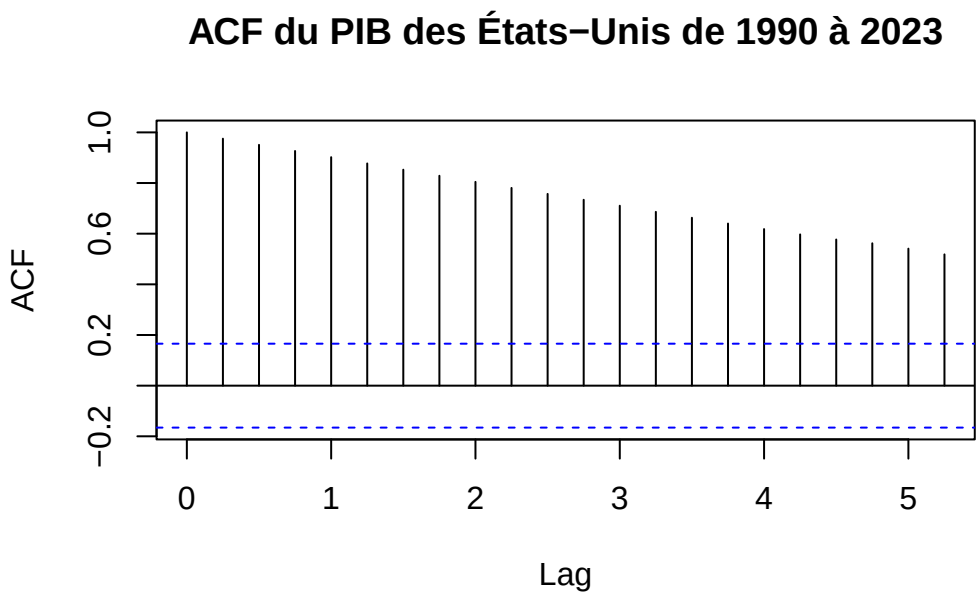


Interprétation

Concernant le graphique sur la période complète, on observe les mêmes dynamiques que précédemment : les zones grisées correspondent aux périodes de récession définies par le NBER, durant lesquelles le PIB marque soit un ralentissement, soit une baisse temporaire. À cela s'ajoute désormais la récession liée à la crise du Covid-19 en 2020, bien visible sur le graphique par une chute marquée du PIB. Cette observation confirme que ces phases conjoncturelles ont un impact clair et immédiat sur la trajectoire du PIB américain.

5.5.2 ACF

```
#conversion en série temporelle
ts_pib2 <- ts(df_Y_us$PIB, start = c(1990, 1), frequency = 4)
# ACF
acf(ts_pib2, main = "ACF du PIB des États-Unis de 1990 à 2023")
```

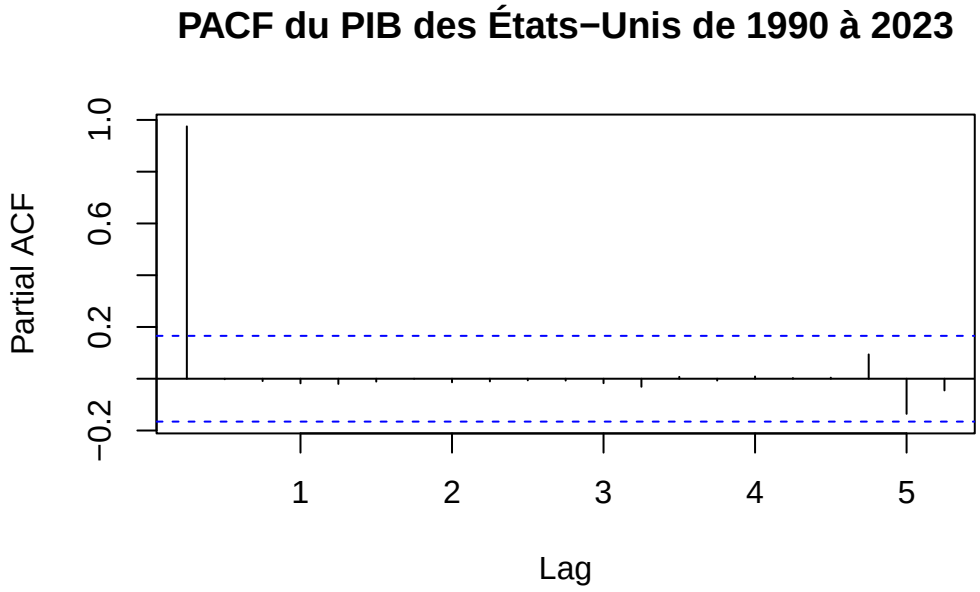


Interprétation

L'ACF sur la période 1990–2023 présente la même structure qu'auparavant : les autocorrélations sont très élevées au début, et décroissent lentement sans devenir rapidement nulles. Cette persistance confirme que la série est fortement non stationnaire, avec un processus avec racine unitaire. Par conséquent, le choc lié à la crise du Covid-19, ne modifie pas la structure fondamentale de la série.

5.5.3 PACF

```
# PACF
pacf(ts_pib2, main = "PACF du PIB des États-Unis de 1990 à 2023")
```



Interprétation Le graphique de la PACF confirme les observations faites précédemment : seul le premier lag est significatif, tandis que les suivants sont proches de zéro et non significatifs. Cette structure suggère que le PIB suit toujours une dynamique de type AR(1) non stationnaire. C'est toujours typique des bruits blancs.

5.5.4 Choix du modèle adéquat

5.5.4.1 Modèle 3: Avec constante et tendance

```
# Modèle 3: Avec tendance et constante
cat("\n--- TEST DF MODÈLE 3: AVEC TENDANCE ET CONSTANTE ---\n")
```

```
--- TEST DF MODÈLE 3: AVEC TENDANCE ET CONSTANTE ---
```

```
# Test de Dickey-Fuller simple
df_test_modele3 <- ur.df(ts_pib2, type = "trend", lags = 0)
summary(df_test_modele3)
```

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression trend


Call:
lm(formula = z.diff ~ z.lag.1 + 1 + tt)
```

```
Residuals:
     Min       1Q   Median       3Q      Max
-1767377   -49618    12110    63105  1179891

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.077e+06  3.955e+05   2.724  0.00730 **
z.lag.1     -1.047e-01  4.093e-02  -2.559  0.01158 *
tt           1.009e+04  3.813e+03   2.646  0.00909 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 209600 on 136 degrees of freedom
Multiple R-squared:  0.05133,   Adjusted R-squared:  0.03738
F-statistic: 3.679 on 2 and 136 DF,  p-value: 0.02779


Value of test-statistic is: -2.5593 12.3819 3.6792

Critical values for test statistics:
      1pct  5pct 10pct
tau3 -3.99 -3.43 -3.13
phi2  6.22  4.75  4.07
phi3  8.43  6.49  5.47
```

```r
# Test de Dickey-Fuller augmenté (avec retards sélectionnés automatiquement)
adf_test_modele3 <- ur.df(ts_pib2, type = "trend", selectlags = "AIC")
summary(adf_test_modele3)
```

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression trend


Call:
lm(formula = z.diff ~ z.lag.1 + 1 + tt + z.diff.lag)

Residuals:
     Min       1Q   Median       3Q      Max
-1820767   -46867    17267    60228   974708

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  9.109e+05  4.094e+05   2.225   0.0277 *
```

```
z.lag.1     -8.647e-02  4.258e-02  -2.031   0.0443 *
tt           8.452e+03  3.960e+03   2.134   0.0347 *
z.diff.lag  -1.364e-01  8.694e-02  -1.569   0.1191
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 209300 on 134 degrees of freedom
Multiple R-squared:  0.06789,   Adjusted R-squared:  0.04702
F-statistic: 3.253 on 3 and 134 DF,  p-value: 0.02381


Value of test-statistic is: -2.0308 13.2023 2.5901

Critical values for test statistics:
      1pct  5pct 10pct
tau3 -3.99 -3.43 -3.13
phi2  6.22  4.75  4.07
phi3  8.43  6.49  5.47
```

Interprétation

Les résultats du test montrent une p-value de 0.02779 pour la statistique F, ce qui suggère que le modèle est globalement significatif. Cependant, en ce qui concerne la stationnarité, la valeur de la statistique de test est de -2.5593, tandis que la valeur critique à 5% est -3.43. Comme la statistique de test est supérieure à la valeur critique, on ne rejette pas l'hypothèse nulle de présence d'une racine unitaire. Cela signifie que la série est non stationnaire.

5.5.4.2 Modèle 2: Avec constante et sans tendance

```r
df_test_modele2 <- ur.df(ts_pib2, type = "drift", lags = 0)
summary(df_test_modele2)
```

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression drift


Call:
lm(formula = z.diff ~ z.lag.1 + 1)

Residuals:
     Min       1Q   Median       3Q      Max
-1746668   -40407    15121    56574  1386765
```

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 5.144e+04  8.033e+04   0.640    0.523
z.lag.1     2.832e-03  4.860e-03   0.583    0.561

Residual standard error: 214200 on 137 degrees of freedom
Multiple R-squared:  0.002474,  Adjusted R-squared:  -0.004808
F-statistic: 0.3397 on 1 and 137 DF,  p-value: 0.5609


Value of test-statistic is: 0.5829 14.4382

Critical values for test statistics:
      1pct  5pct 10pct
tau2 -3.46 -2.88 -2.57
phi1  6.52  4.63  3.81
```

```r
adf_test_modele2 <- ur.df(ts_pib2, type = "drift", selectlags = "AIC")
summary(adf_test_modele2)
```

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression drift


Call:
lm(formula = z.diff ~ z.lag.1 + 1 + z.diff.lag)

Residuals:
     Min       1Q   Median       3Q      Max
-1823568   -32928    11499    59999  1062191

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  5.385e+04  8.047e+04   0.669   0.5045
z.lag.1      3.814e-03  4.884e-03   0.781   0.4362
z.diff.lag  -1.854e-01  8.495e-02  -2.183   0.0308 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 212000 on 135 degrees of freedom
Multiple R-squared:  0.03621,   Adjusted R-squared:  0.02193
F-statistic: 2.536 on 2 and 135 DF,  p-value: 0.08296
```

```
Value of test-statistic is: 0.7809 17.0766

Critical values for test statistics:
      1pct  5pct 10pct
tau2 -3.46 -2.88 -2.57
phi1  6.52  4.63  3.81
```

Interprétation

La statistique des deux tests est de 0.5829 et 0.7809, qui est supérieure à la valeur critique de -2.88 à 5%, donc nous ne rejetons pas l'hypothèse nulle de présence d'une racine unitaire. Cela suggère que la série est non stationnaire. Par conséquent, nous passons au modèle 1

5.5.4.3 Modèle 1: Sans constante et tendance

```r
#Test de Dickey-Fuller simple
    df_test_modele1 <- ur.df(ts_pib2, type = "none", lags = 0)
    summary(df_test_modele1)
```

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression none


Call:
lm(formula = z.diff ~ z.lag.1 - 1)

Residuals:
     Min       1Q   Median       3Q      Max
-1757954   -28516    10619    53572  1380439

Coefficients:
        Estimate Std. Error t value Pr(>|t|)
z.lag.1 0.005863   0.001097   5.347 3.62e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 213700 on 138 degrees of freedom
Multiple R-squared:  0.1716,    Adjusted R-squared:  0.1656
F-statistic: 28.59 on 1 and 138 DF,  p-value: 3.616e-07


Value of test-statistic is: 5.3468
```

```
Critical values for test statistics:
      1pct  5pct 10pct
tau1 -2.58 -1.95 -1.62
```

```r
# Test de Dickey-Fuller augmenté
    adf_test_modele1 <- ur.df(ts_pib2, type = "none", selectlags = "AIC")
    summary(adf_test_modele1)
```

```
###############################################
# Augmented Dickey-Fuller Test Unit Root Test #
###############################################

Test regression none


Call:
lm(formula = z.diff ~ z.lag.1 - 1 + z.diff.lag)

Residuals:
     Min       1Q   Median       3Q      Max
-1835282   -29475    14184    60053  1055597

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
z.lag.1     0.006981   0.001200   5.817 4.08e-08 ***
z.diff.lag -0.185467   0.084777  -2.188   0.0304 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 211600 on 136 degrees of freedom
Multiple R-squared:  0.1997,    Adjusted R-squared:  0.1879
F-statistic: 16.97 on 2 and 136 DF,  p-value: 2.638e-07


Value of test-statistic is: 5.8175

Critical values for test statistics:
      1pct  5pct 10pct
tau1 -2.58 -1.95 -1.62
```

Interprétation

Les deux tests indiquent que la série n'est pas stationnaire en niveau, car elle présente une racine unitaire. En effet, la statistique de test est supérieure à l'ensemble des valeurs critiques (par exemple, avec le test de Dickey-Fuller augmenté : 5.8175 > -1.95).

Peu importe le modèle testé (avec ou sans retards), l'hypothèse nulle de présence d'une racine unitaire n'est jamais rejetée. Cela confirme que la série est non stationnaire, ce

qui est également cohérent avec les résultats obtenus précédemment à partir de l'ACF et PACF.

###Test de stationarité de KPSS

```
kpss_test <- kpss.test(ts_pib2, null = "Trend")
```

```
Warning in kpss.test(ts_pib2, null = "Trend"): p-value smaller than printed
p-value
```

```
print(kpss_test)
```

```

    KPSS Test for Trend Stationarity

data:  ts_pib2
KPSS Trend = 0.23393, Truncation lag parameter = 4, p-value = 0.01
```

Interprétation Le test de KPSS appliqué au PIB sur la période 1990–2023 donne une statistique de test de 0.2339, avec une p-value inférieure à 0.01.

Cela signifie que l'on rejette l'hypothèse nulle de stationnarité autour d'une tendance, même au seuil de 1%. Par cosnéquent, le test suggère que la série n'est pas stationnaire, ce qui vient confirmer les résultats des tests de Dickey-Fuller ainsi que l'analyse visuelle de l'ACF et de la PACF.

5.5.5 Conclusion de la période globale (1990-2023)

Après avoir appliqué l'ensemble des tests de stationnarité sur la période étendue de 1990 à 2023, nous parvenons aux mêmes conclusions que précédemment (1990–2019) : la série du PIB reste non stationnaire et présente une racine unitaire, quels que soient les modèles testés (avec ou sans constante, avec ou sans tendance).

Les résultats des tests de Dickey-Fuller (simples et augmentés), tout comme ceux du test de KPSS, confirment la non-stationnarité observée visuellement dans les graphiques ACF et PACF. Ainsi, l'allongement de la période n'altère pas la dynamique structurelle de la série, et vient renforcer les constats établis sur la période initiale.