Taller 1 - Simulación e identificación de procesos estocásticos

Adaptado del taller 2022 de Federico Molina

Series Cronológicas 2024

Marzo 2024

1. Simulación de procesos estocásticos

1.1. Ruido Blanco Gaussiano

El Ruido Blanco es el proceso estocástico débilmente estacionario más simple y representa la "aleatoriedad pura". Cumple las siguientes propiedades:

1. $E(\varepsilon_t) = 0$
2. $\gamma_0 = \sigma^2$
3. $\gamma_k = 0$ si $k \neq 0$

Si las variables tienen una distribución normal, el proceso será Gaussiano.

```
# Como las observaciones son independientes, podemos simular un Ruido Blanco Gaussiano con la función r
RB <- ts(rnorm(1500, 0, 1))

# Otra forma: función arima.sim (por defecto, el Ruido Blanco es Gaussiano)
# RB <- arima.sim(model = list(), n = 1500)
```

```
# Graficamos el proceso
autoplot(RB) +
  geom_hline(yintercept = 0, col = "red") +
  labs(x = "Tiempo",
       y = "Valor")
```

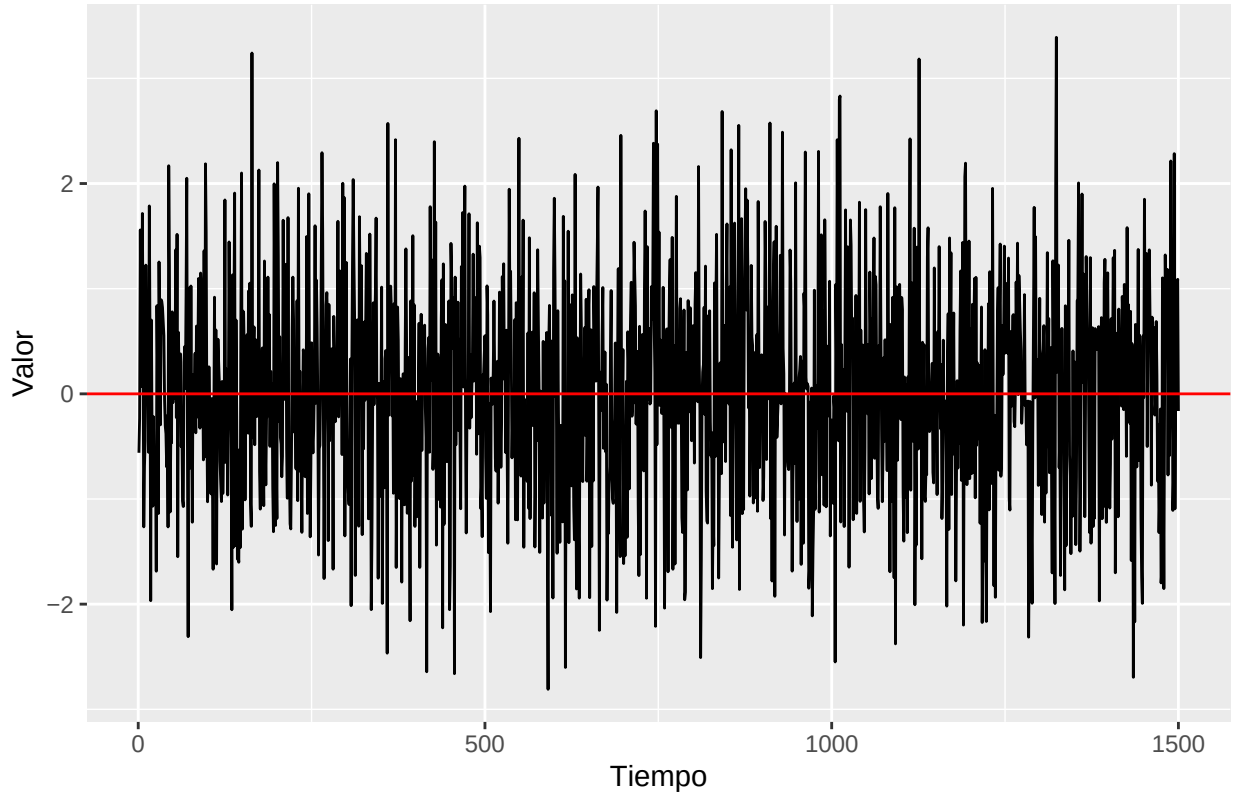

Figura 1: Simulación de un Ruido Blanco Gaussiano.

```
# Otra forma: usar plot()
# plot(RB, main = "Simulación de un Ruido Blanco Gaussiano",
#      xlab = "Tiempo", ylab = "Valor")
```

```
# FAC
acf_rb <- ggAcf(RB, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")

# FACP
pacf_rb <- ggAcf(RB, lag.max = 24, type = "partial") +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       title = "")

grid.arrange(acf_rb, pacf_rb)
```

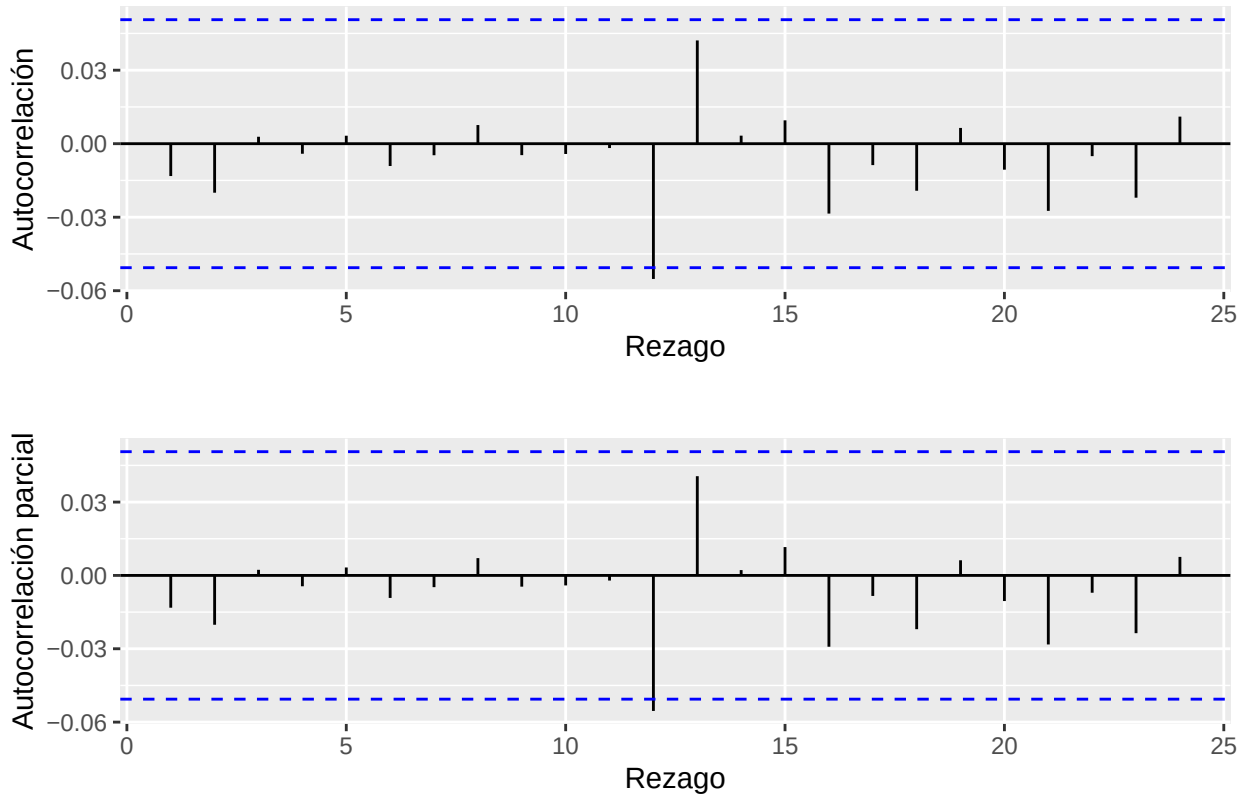

Figura 2: Funciones de Autocorrelación y Autocorrelación Parcial para un Ruido Blanco Gaussiano.

```
# Otra forma para obtener la FACP
# ggPacf(RB, lag.max = 24) +
#   labs(x = "Rezago",
#        y = "Autocorrelación parcial",
#        title = "")

# Otra forma de obtener la FAC y la FACP con R base

# acf(RB, main = "FAC para un Ruido Blanco",
#     xlab = "Rezago",
#     ylab = "Autocorrelación")

# pacf(RB, main = "FACP para un Ruido Blanco",
#      xlab = "Rezago",
#      ylab = "Autocorrelación parcial")
```

1.2. Random Walk

El proceso Random Walk es un caso particular de los procesos AR(1) cuando el coeficiente ϕ toma el valor 1, por lo que no es estacionario.

$$y_t = y_{t-1} + \varepsilon_t$$

siendo ε_t un Ruido Blanco.

```
# Simulamos un Ruido Blanco
RB <- rnorm (1500, 0, 1)
head(RB)
```

```
## [1] -0.8209867 -0.3072572 -0.9020980  0.6270687  1.1203550  2.1272136
```

```
# Obtenemos un Random Walk como la suma acumulada de shocks
RW <- cumsum(RB)
head(RW)
```

```
## [1] -0.8209867 -1.1282439 -2.0303419 -1.4032732 -0.2829182  1.8442954
```

```
# Convertimos el proceso a formato ts
RW <- ts(RW)

# Otra forma: función arima.sim
# RW <- arima.sim(model = list(order = c(0, 1, 0)), n = 1500)
```

```
# Graficamos el proceso
autoplot(RW) +
  geom_hline(yintercept = 0, col = "red") +
  labs(x = "Tiempo",
       y = "Valor")
```

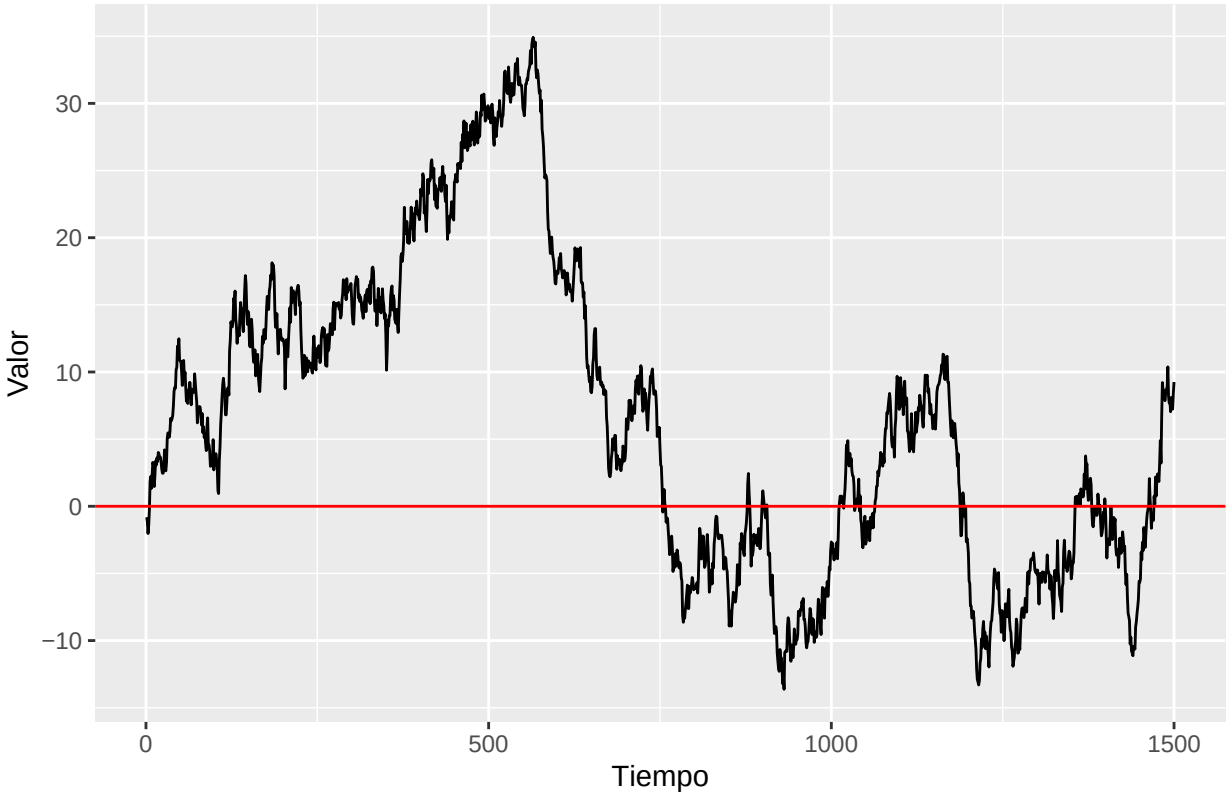

Figura 3: Simulación de un Random Walk.

```
# FAC
ggAcf(RW, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")
```

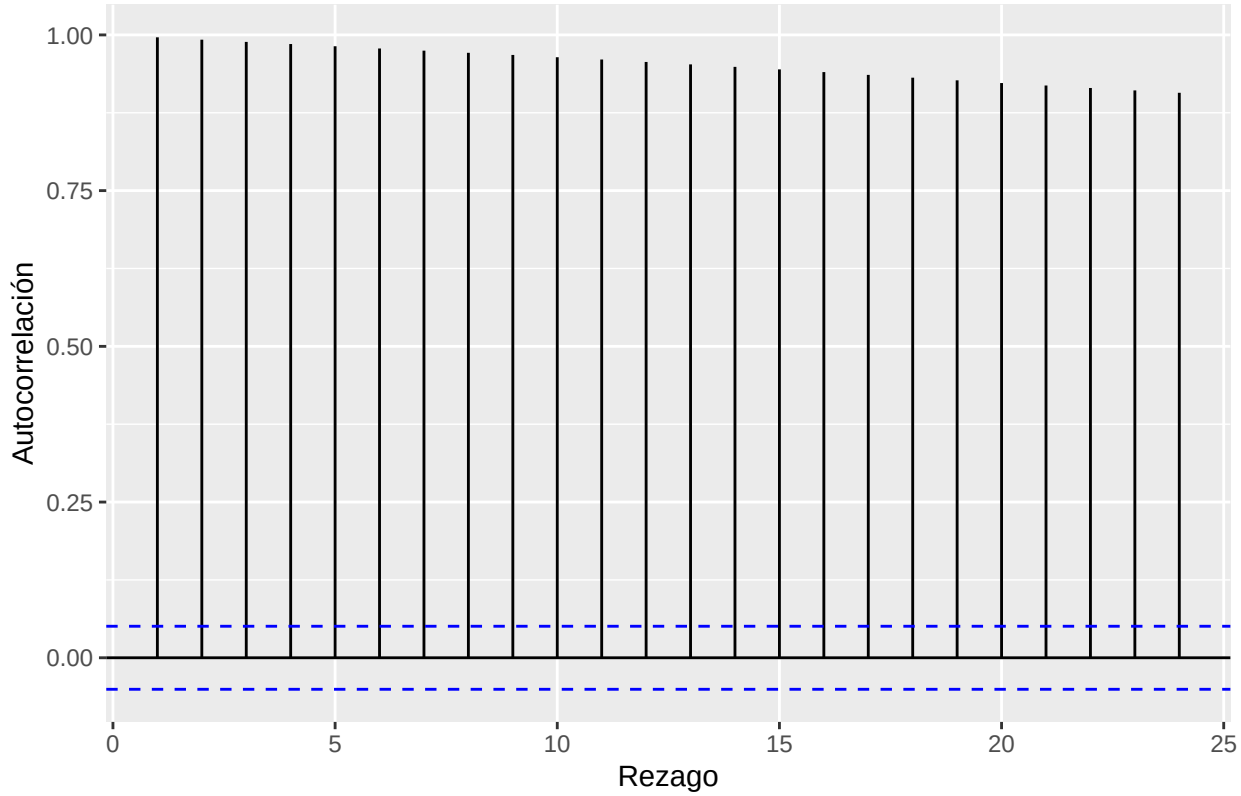

Figura 4: Función de Autocorrelación para un Random Walk.

Cuando se agrega una constante, se obtiene un Random Walk con *drift*:

$$y_t = \delta + y_{t-1} + \varepsilon_t$$

siendo ε_t un Ruido Blanco.

```
# Simulamos un Ruido Blanco
RB <- rnorm (1500, 0, 1)
head(RB)
```

```
## [1] -0.15030748 -0.32775713 -1.44816529 -0.69728458  2.59849023 -0.03741501
```

```
# Obtenemos un Random Walk como la suma acumulada de shocks más el drift de 0.2
delta <- 0.2
RW_drift <- RB + delta
RW_drift <- cumsum(RW_drift)
head(RW_drift)
```

```
## [1]  0.04969252 -0.07806461 -1.32622990 -1.82351449  0.97497575  1.13756073
```

```r
# Convertimos el proceso a formato ts
RW_drift <- ts(RW_drift)

# Otra forma: función arima.sim
# RW_drift <- arima.sim(model = list(order = c(0, 1, 0)), n = 1500, mean = 0.2)
```

```r
# Graficamos el proceso
autoplot(RW_drift) +
  labs(x = "Tiempo",
       y = "Valor")
```

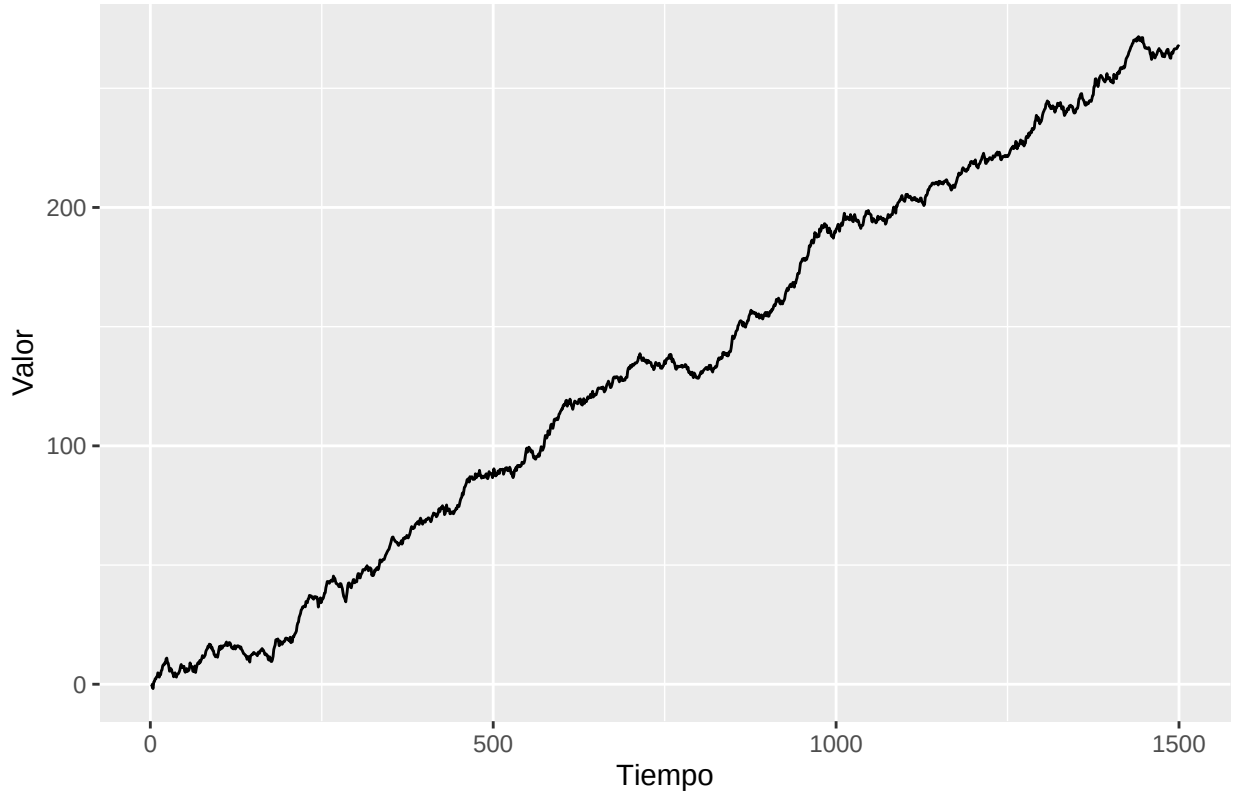

Figura 5: Simulación de un Random Walk con drift.

```r
# FAC
ggAcf(RW_drift, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")
```

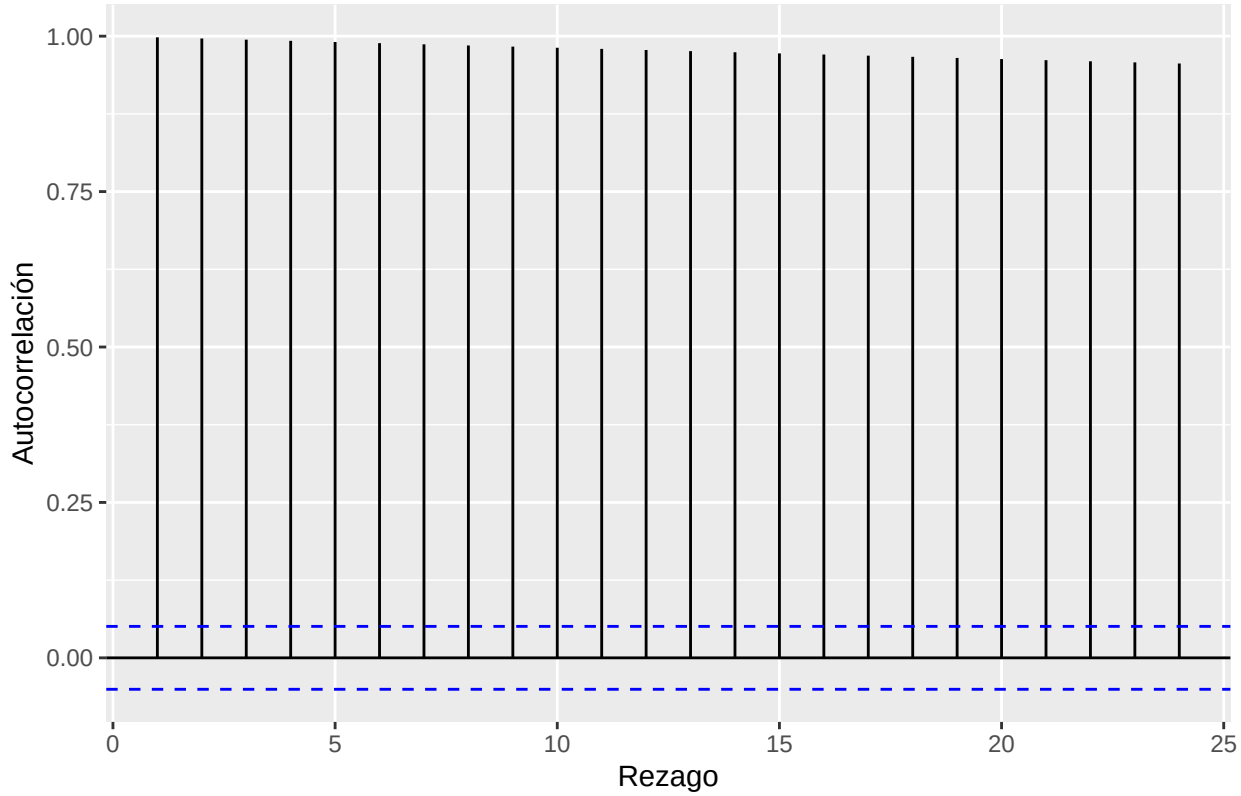

Figura 6: Función de Autocorrelación para un Random Walk con drift.

1.3. Procesos AR(1)

Los procesos autorregresivos son aquellos en los que el valor en un determinado período depende de valores anteriores más un shock, el cual introduce la aleatoriedad en la serie. En particular, los procesos AR(1) dependen de lo ocurrido en el período inmediatamente anterior, de forma que pueden escribirse como:

$$y_t = \phi y_{t-1} + \varepsilon_t$$

siendo ε_t un Ruido Blanco.

En los procesos autorregresivos estacionarios, se observa una caída exponencial en los coeficientes de autocorrelación hasta no ser significativamente distintos de cero. Cuando el signo de ϕ en un proceso AR(1) es negativo, dicha caída alterna su signo.

A su vez, a menor valor absoluto de dicho coeficiente, menor persistencia tendrá el proceso y más rápidamente se reducirán las autocorrelaciones. Si $|\phi| < 1$, el proceso es estacionario, lo cual equivale a que la raíz del polinomio característico se encuentre fuera del círculo unitario:

$$y_t - \phi y_{t-1} = \varepsilon_t \Rightarrow (1 - \phi L) yt = \varepsilon_t$$

$$1 - \phi L = 0 \Rightarrow L = \frac{1}{\phi} \Rightarrow \left|\frac{1}{\phi}\right| > 1 \Leftrightarrow |\phi| < 1$$

Por su parte, a través del autocorrelograma parcial, se determina el orden del proceso. De esta manera, para un proceso AR(1), sólo el primer coeficiente de autocorrelación parcial debería ser significativamente distinto de cero (positivo cuando $\phi > 0$ y negativo cuando $\phi < 0$).

```
# Simulamos un proceso AR(1) con phi = 0,7
simula_ar1_1 <- arima.sim(list(ar = 0.7), n = 1500)
class(simula_ar1_1) # Se crea un objeto de clase ts
```

```
## [1] "ts"
```

```
head(simula_ar1_1)
```

```
## Time Series:
## Start = 1
## End = 6
## Frequency = 1
## [1]  1.2636116 -0.3467380  0.3638529 -0.1348934  0.3003896 -0.6650458
```

```
# Graficamos el proceso
autoplot(simula_ar1_1) +
  geom_hline(yintercept = 0, col = "red") +
  labs(x = "Tiempo",
       y = "Valor")
```

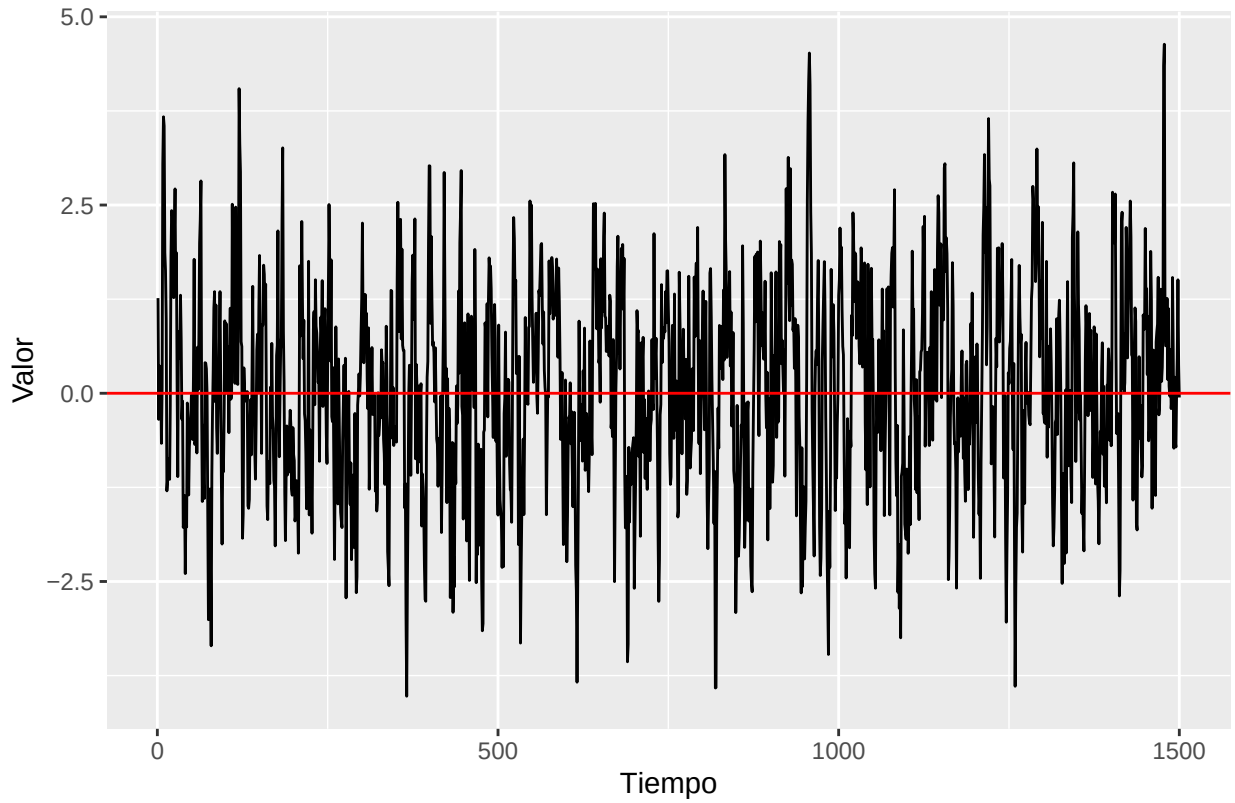

Figura 7: Simulación de un proceso AR(1) con coeficiente positivo y alta persistencia.

```r
# FAC
acf_simula_ar1_1 <- ggAcf(simula_ar1_1, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")

# FACP
pacf_simula_ar1_1 <- ggAcf(simula_ar1_1, lag.max = 24, type = "partial") +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       title = "")

grid.arrange(acf_simula_ar1_1, pacf_simula_ar1_1)
```

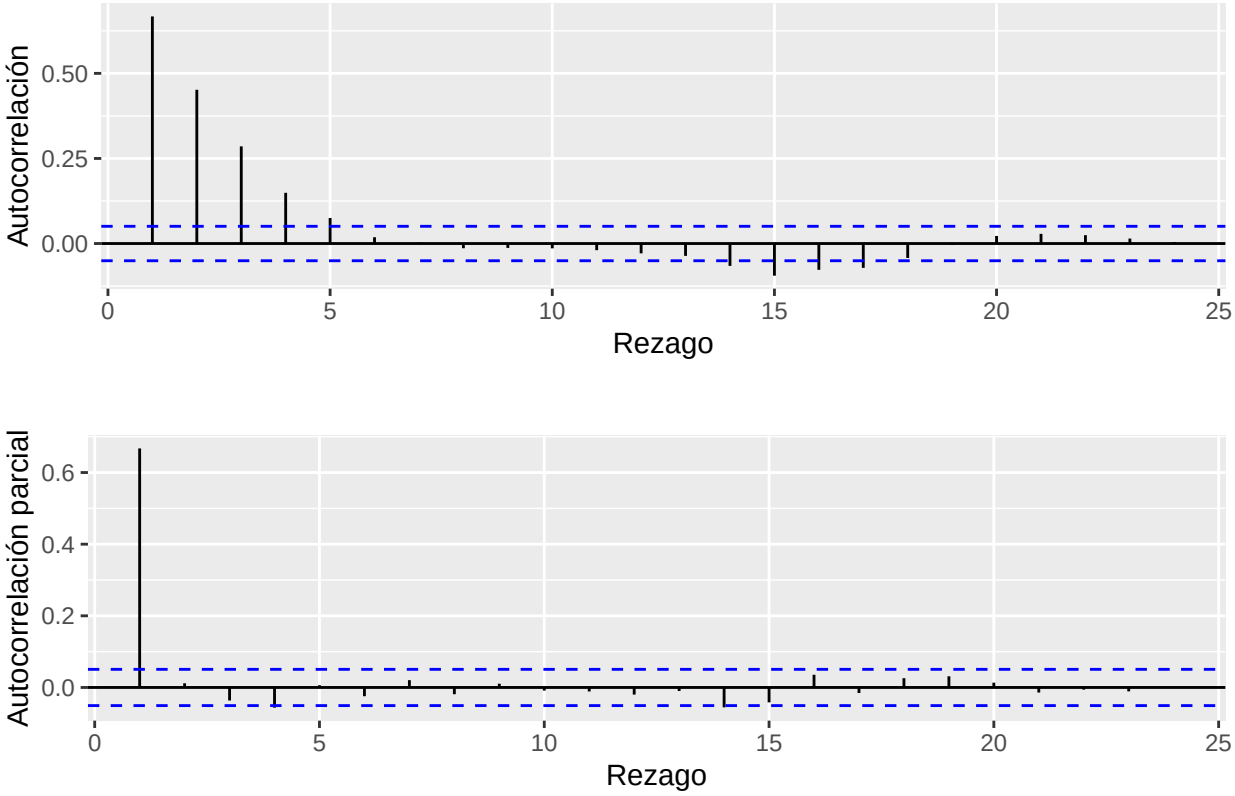

Figura 8: Funciones de Autocorrelación y Autocorrelación Parcial para un proceso AR(1) con coeficiente positivo y alta persistencia.

```r
# Obtenemos las autocorrelaciones teóricas
ar1_1_acf_teorico <- ARMAacf(ar = 0.7, lag.max = 14, pacf = FALSE)
ar1_1_pacf_teorico <- ARMAacf(ar = 0.7, lag.max = 14, pacf = TRUE)

# Obtenemos las autocorrelaciones estimadas
ar1_1_acf_est <- ggAcf(simula_ar1_1, plot = FALSE, lag.max = 14, type = "correlation")
ar1_1_pacf_est <- ggAcf(simula_ar1_1, plot = FALSE, lag.max = 14, type = "partial")
```

```r
# Ordenamos los datos de la FAC para poder graficarlos
ar1_1_teorico_est_acf <- data.frame(rezago = 0:14, "Teórica" = ar1_1_acf_teorico, "Estimación" = ar1_1_a
ar1_1_teorico_est_acf <- ar1_1_teorico_est_acf %>%
  pivot_longer(cols = c("Teórica", "Estimación"), values_to = "valores")

# Graficamos las FAC teórica y estimada
ar1_1_acf <- ggplot(ar1_1_teorico_est_acf) +
  geom_col(aes(x= rezago, y = valores, fill = name),
           position = "dodge", width = 0.2) +
  labs(x = "Rezago",
       y = "Autocorrelación",
       fill = "FAC")

# Ordenamos los datos de la FACP para poder graficarlos
ar1_1_teorico_est_pacf <- data.frame(rezago = 1:14, "Teórica" = ar1_1_pacf_teorico, "Estimación" = ar1_1
ar1_1_teorico_est_pacf <- ar1_1_teorico_est_pacf %>%
  pivot_longer(cols = c("Teórica", "Estimación"), values_to = "valores")

# Graficamos las FACP teórica y estimada
ar1_1_pacf <- ggplot(ar1_1_teorico_est_pacf) +
  geom_col(aes(x= rezago, y = valores, fill = name),
           position = "dodge", width = 0.2) +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       fill = "FACP")

grid.arrange(ar1_1_acf, ar1_1_pacf)
```

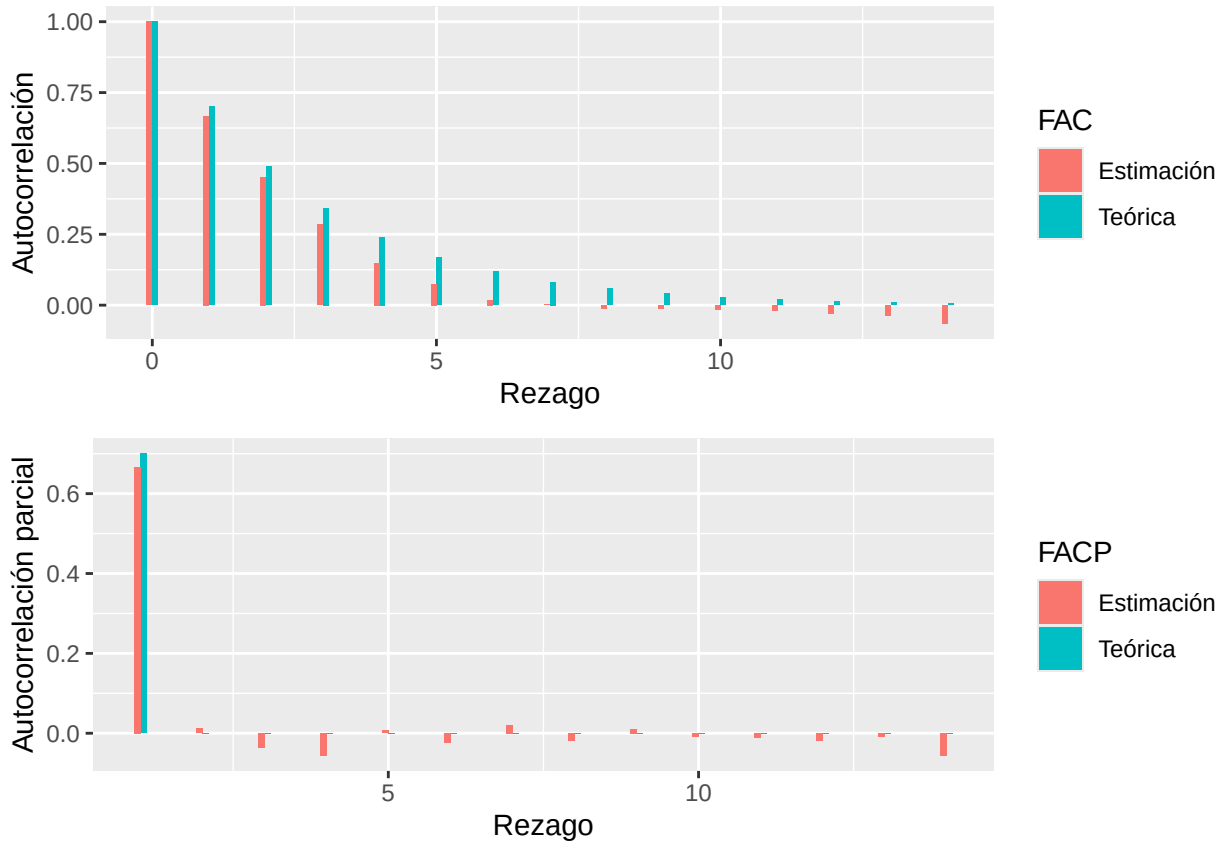

Figura 9: Funciones de Autocorrelación y Autocorrelación Parcial teóricas y estimadas para 1500 observaciones de un proceso AR(1) con coeficiente positivo.

```
# Simulamos un proceso AR(1) con phi = -0,3
simula_ar1_2 <- arima.sim(list(ar = -0.3), n = 1500)
```

```
# Graficamos el proceso
autoplot(simula_ar1_2) +
  geom_hline(yintercept = 0, col = "red") +
  labs(x = "Tiempo",
       y = "Valor")
```

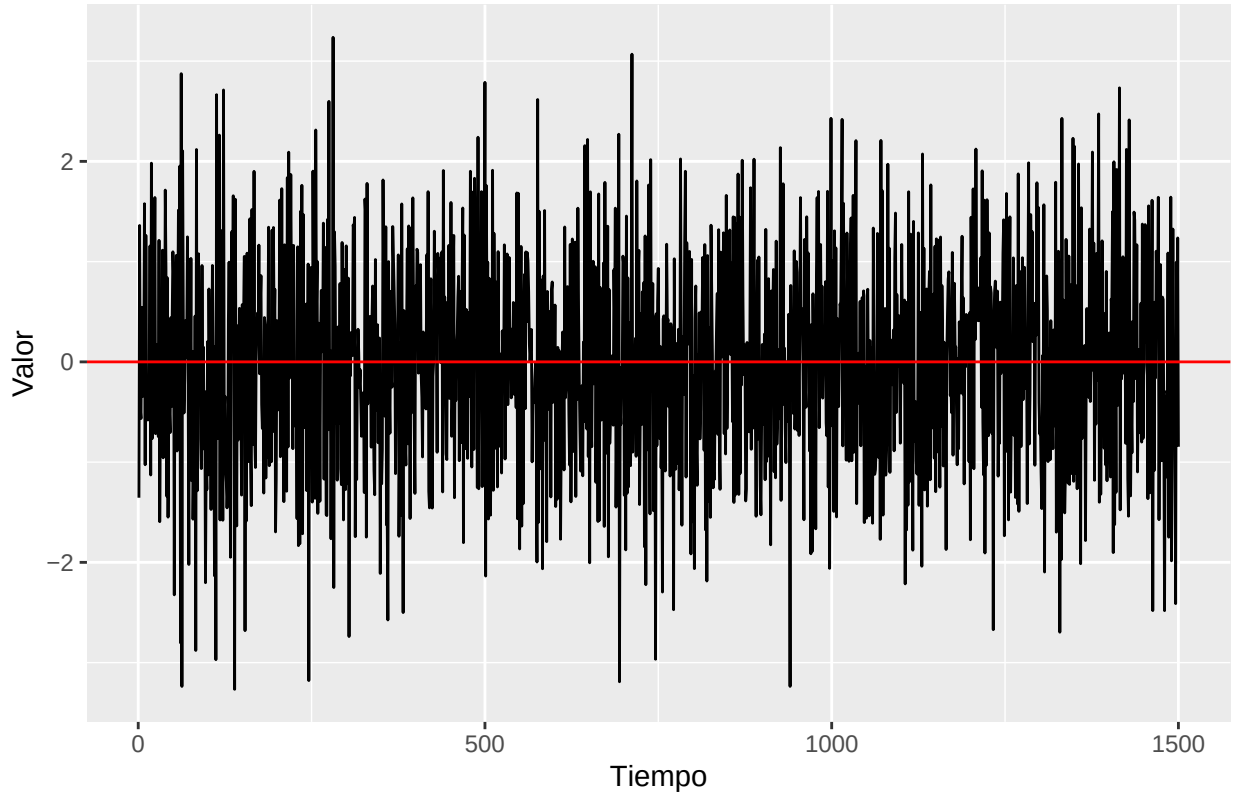

Figura 10: Simulación de un proceso AR(1) con coeficiente negativo y baja persistencia.

```r
# FAC
acf_simula_ar1_2 <- ggAcf(simula_ar1_2, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")

# FACP
pacf_simula_ar1_2 <- ggAcf(simula_ar1_2, lag.max = 24, type = "partial") +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       title = "")

grid.arrange(acf_simula_ar1_2, pacf_simula_ar1_2)
```

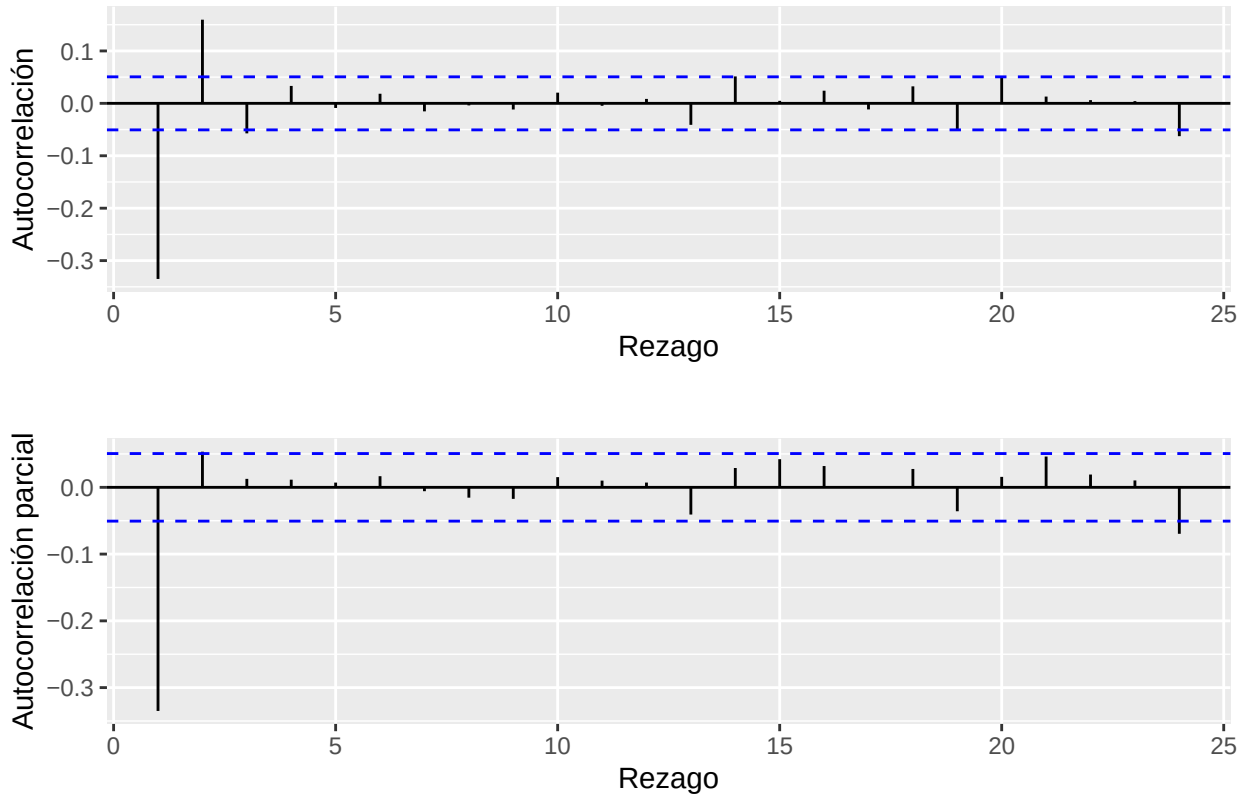

Figura 11: Funciones de Autocorrelación y Autocorrelación Parcial para un proceso AR(1) con coeficiente negativo y baja persistencia.

```
# Obtenemos las autocorrelaciones teóricas
ar1_2_acf_teorico <- ARMAacf(ar = -0.3, lag.max = 14, pacf = FALSE)
ar1_2_pacf_teorico <- ARMAacf(ar = -0.3, lag.max = 14, pacf = TRUE)

# Obtenemos las autocorrelaciones estimadas
ar1_2_acf_est <- ggAcf(simula_ar1_2, plot = FALSE, lag.max = 14, type = "correlation")
ar1_2_pacf_est <- ggAcf(simula_ar1_2, plot = FALSE, lag.max = 14, type = "partial")

# Ordenamos los datos de la FAC para poder graficarlos
ar1_2_teorico_est_acf <- data.frame(rezago = 0:14, "Teórica" = ar1_2_acf_teorico, "Estimación" = ar1_2_a
ar1_2_teorico_est_acf <- ar1_2_teorico_est_acf %>%
  pivot_longer(cols = c("Teórica", "Estimación"), values_to = "valores")

# Graficamos las FAC teórica y estimada
ar1_2_acf <- ggplot(ar1_2_teorico_est_acf) +
  geom_col(aes(x= rezago, y = valores, fill = name),
           position = "dodge", width = 0.2) +
  labs(x = "Rezago",
       y = "Autocorrelación",
       fill = "FAC")

# Ordenamos los datos de la FACP para poder graficarlos
```

```r
ar1_2_teorico_est_pacf <- data.frame(rezago = 1:14, "Teórica" = ar1_2_pacf_teorico, "Estimación" = ar1_2
ar1_2_teorico_est_pacf <- ar1_2_teorico_est_pacf %>%
  pivot_longer(cols = c("Teórica", "Estimación"), values_to = "valores")

# Graficamos las FACP teórica y estimada
ar1_2_pacf <- ggplot(ar1_2_teorico_est_pacf) +
  geom_col(aes(x= rezago, y = valores, fill = name),
           position = "dodge", width = 0.2) +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       fill = "FACP")

grid.arrange(ar1_2_acf, ar1_2_pacf)
```

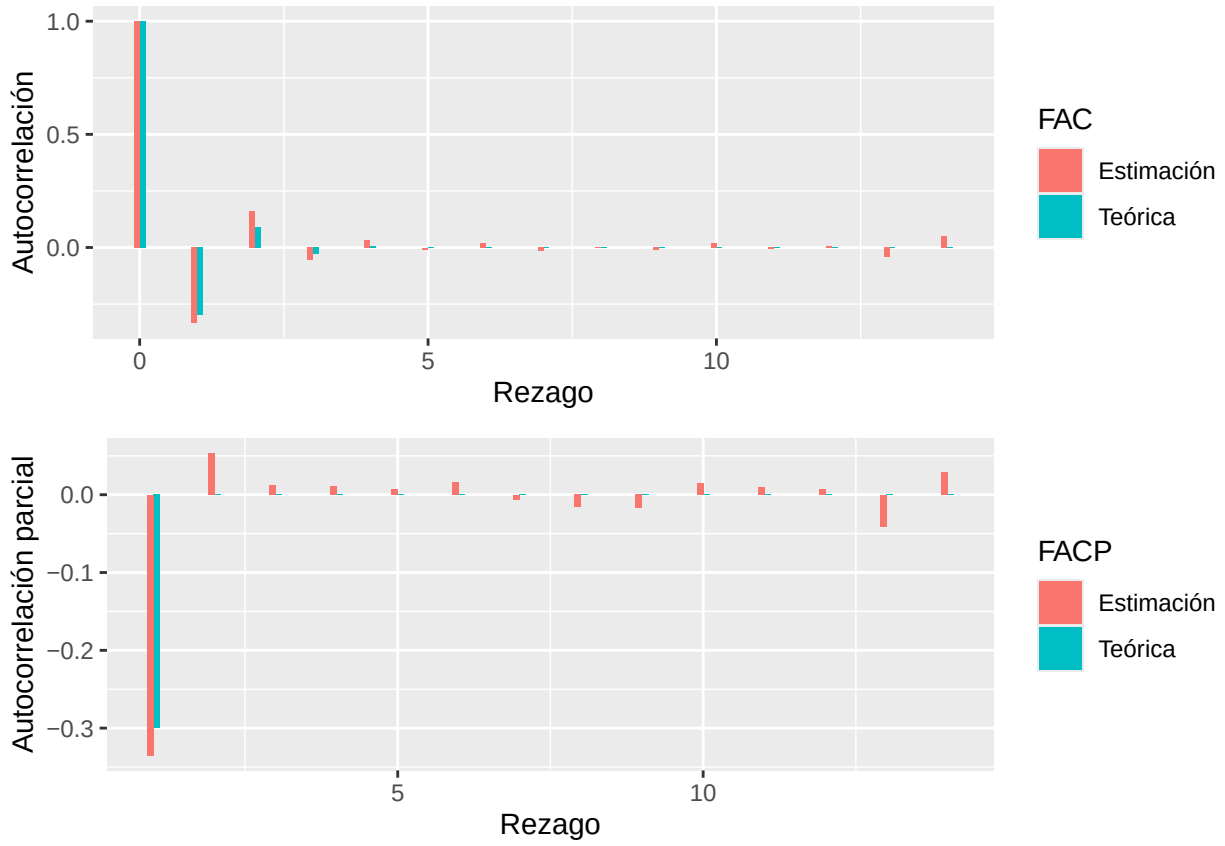

Figura 12: Funciones de Autocorrelación y Autocorrelación Parcial teóricas y estimadas para 1500 observaciones de un proceso AR(1) con coeficiente negativo.

1.4. Procesos MA(1)

Los procesos de medias móviles son aquellos que pueden expresarse como combinación lineal de shocks pasados y presentes. En los procesos MA(1), únicamente se considera el Ruido Blanco presente y el shock del período inmediatamente anterior:

$$Y_t = \varepsilon_t - \theta\varepsilon_{t-1}$$

siendo ε_t un Ruido Blanco.

Al tratarse de combinaciones lineales finitas de shocks por definición estacionarios, los procesos MA de orden **finito** serán también estacionarios.

En los modelos de medias móviles, los autocorrelogramas se comportan "al revés" que en los procesos autorregresivos: la Función de Autocorrelación Parcial debe exhibir una caída exponencial y la Función de Autocorrelación denota el orden del modelo. De esta manera, para un proceso MA(1) sólo el primer coeficiente de autocorrelación será significativamente distinto de cero.

De acuerdo con la especificación anterior en la que θ está precedida por un signo negativo, si $\theta > 0$, los autocorrelogramas parciales presentan signo positivo para todos los rezagos. En cambio, cuando $\theta < 0$, las autocorrelaciones parciales alternan su signo.

```
# Simulamos un proceso MA(1) con theta = 0,7
# En la función arima.sim(), el signo del coeficiente está invertido
simula_ma_1 <- arima.sim(list(ma = -0.7), n = 1500)
```

```
# Graficamos el proceso
autoplot(simula_ma_1) +
  geom_hline(yintercept = 0, col = "red") +
  labs(x = "Tiempo",
       y = "Valor")
```

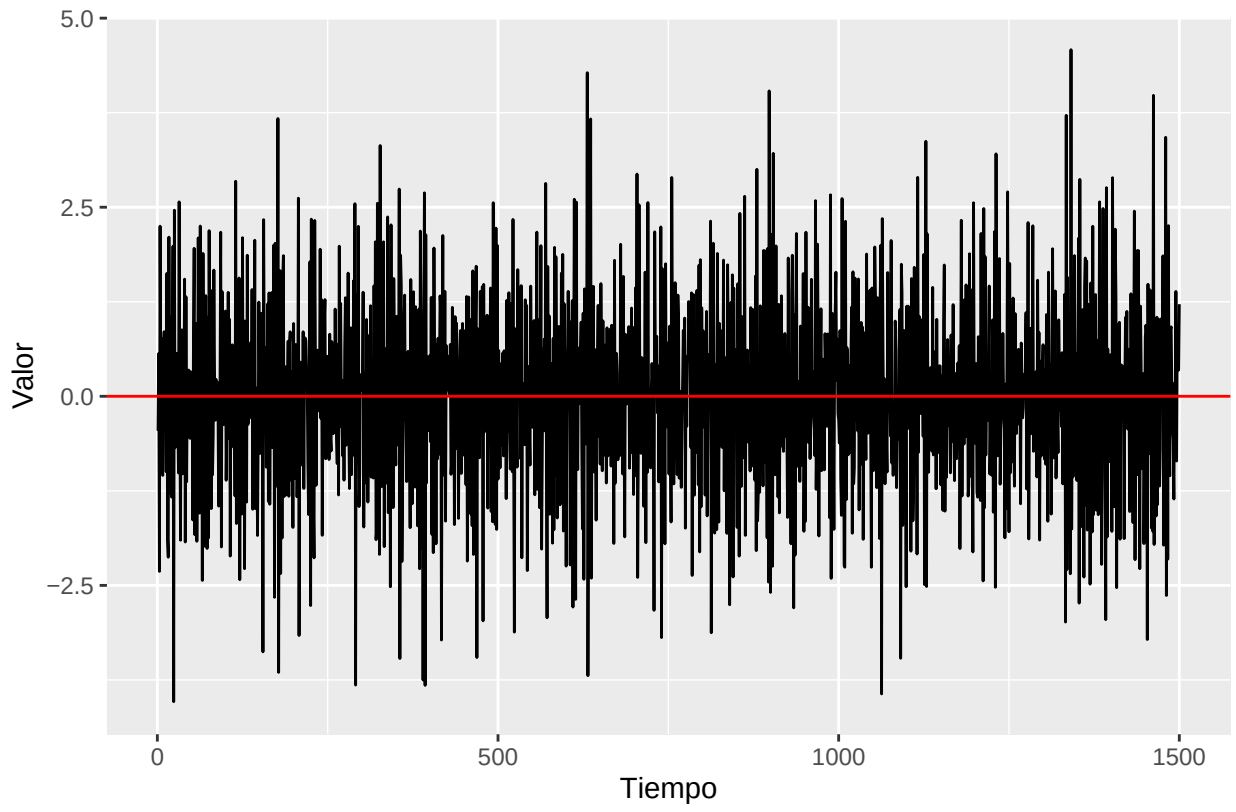

Figura 13: Simulación de un proceso MA(1) con coeficiente positivo.

```r
# FAC
acf_simula_ma_1 <- ggAcf(simula_ma_1, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")

# FACP
pacf_simula_ma_1 <- ggAcf(simula_ma_1, lag.max = 24, type = "partial") +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       title = "")

grid.arrange(acf_simula_ma_1, pacf_simula_ma_1)
```

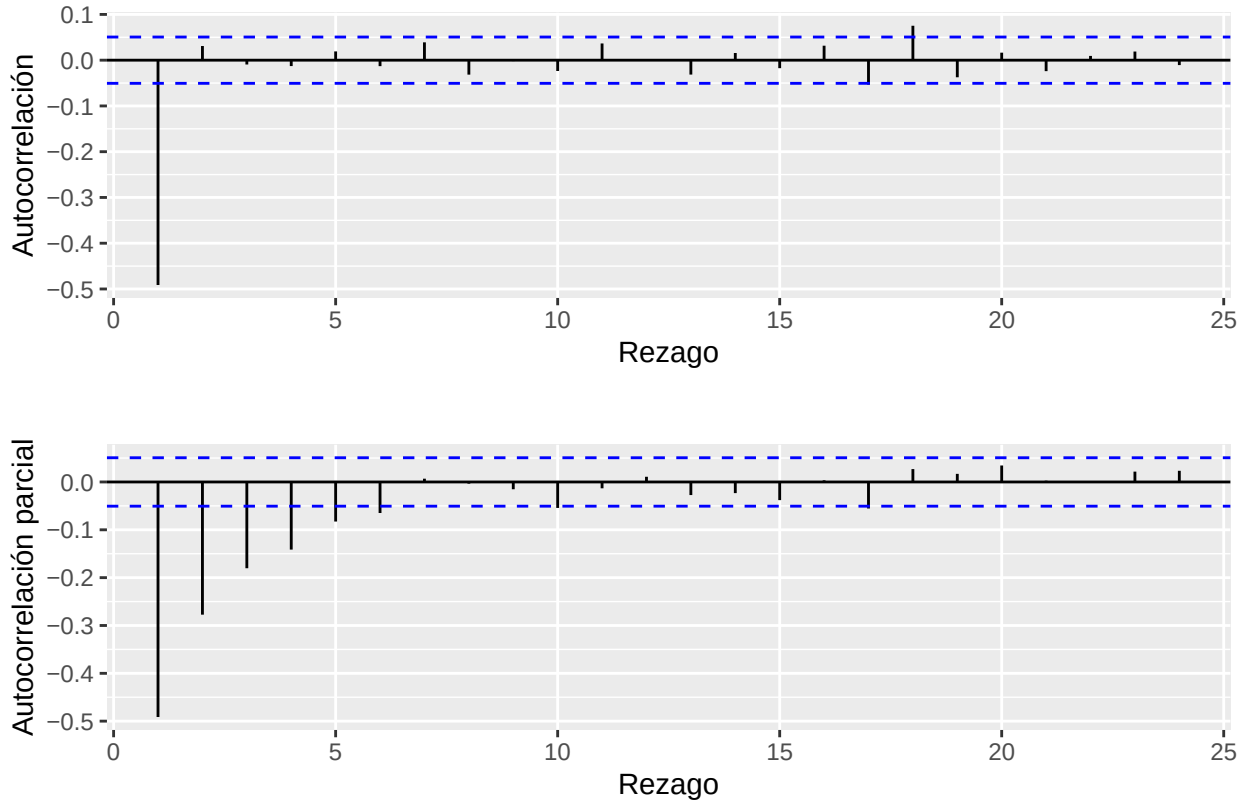

Figura 14: Funciones de Autocorrelación y Autocorrelación Parcial para un proceso MA(1) con coeficiente positivo.

```r
# Simulamos un proceso MA(1) con theta = -0,7
# En la función arima.sim(), el signo del coeficiente está invertido
simula_ma_2 <- arima.sim(list(ma = 0.7), n = 1500)
```

```r
# Graficamos el proceso
autoplot(simula_ma_1) +
  geom_hline(yintercept = 0, col = "red") +
```

```
  labs(x = "Tiempo",
       y = "Valor")
```

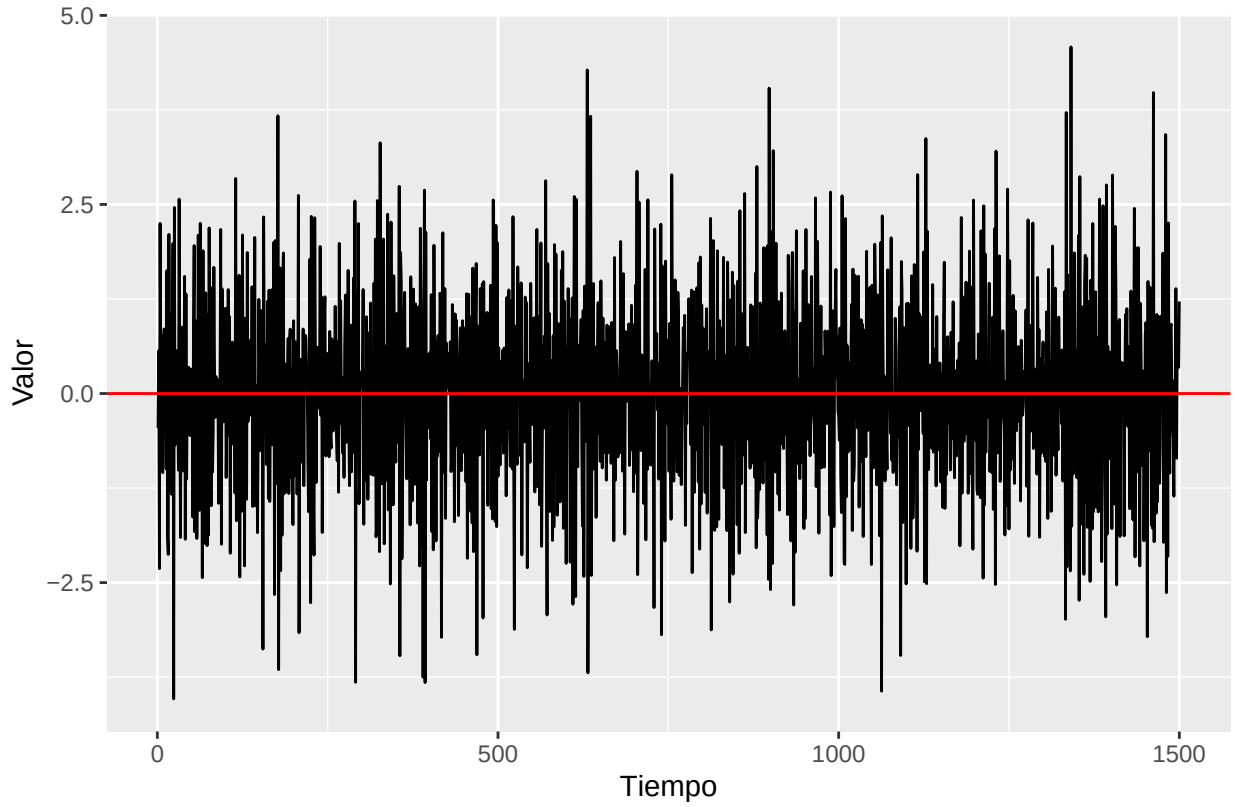

Figura 15: Simulación de un proceso MA(1) con coeficiente negativo.

```
# FAC
acf_simula_ma_2 <- ggAcf(simula_ma_2, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")

# FACP
pacf_simula_ma_2 <- ggAcf(simula_ma_2, lag.max = 24, type = "partial") +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       title = "")

grid.arrange(acf_simula_ma_2, pacf_simula_ma_2)
```

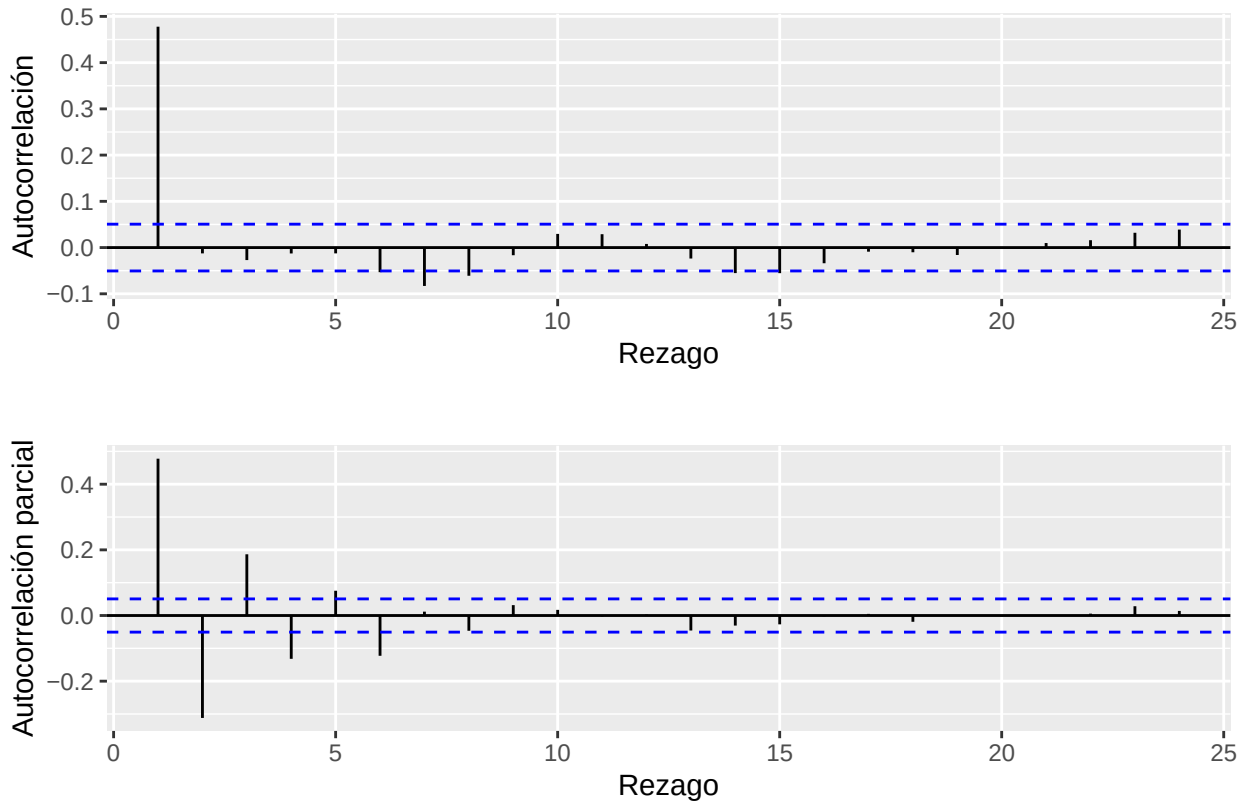

Figura 16: Funciones de Autocorrelación y Autocorrelación Parcial para un proceso MA(1) con coeficiente negativo.

1.5. Procesos AR(2)

Los procesos AR(2) pueden escribirse como:

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \varepsilon_t$$

siendo ε_t un Ruido Blanco.

Para determinar si el proceso es estacionario, alcanza con verificar que todas las raíces del polinomio característico $1 - \phi_1 L - \phi_2 L^2$ sean mayores a la unidad en términos absolutos.

En este caso, el autocorrelograma parcial mostrará que únicamente los primeros dos coeficientes son distintos de cero. Tal como sucedía con el resto de los procesos autorregresivos, si es estacionario, las autocorrelaciones simples deberían mostrar una caída exponencial.

```
# Simulamos un proceso AR(2) con phi1 = 0,7 y phi2 = -0,5
simula_ar2_1 <- arima.sim(n = 1500, list(ar = c(0.7, -0.5)))

# Chequeamos estacionariedad
# La función Mod() obtiene el valor absoluto y polyroot() las raíces del polinomio
Mod(polyroot(c(1, -0.7, 0.5)))
```

```
## [1] 1.414214 1.414214
```

```r
# Graficamos el proceso
autoplot(simula_ar2_1) +
  geom_hline(yintercept = 0, col = "red") +
  labs(x = "Tiempo",
       y = "Valor")
```

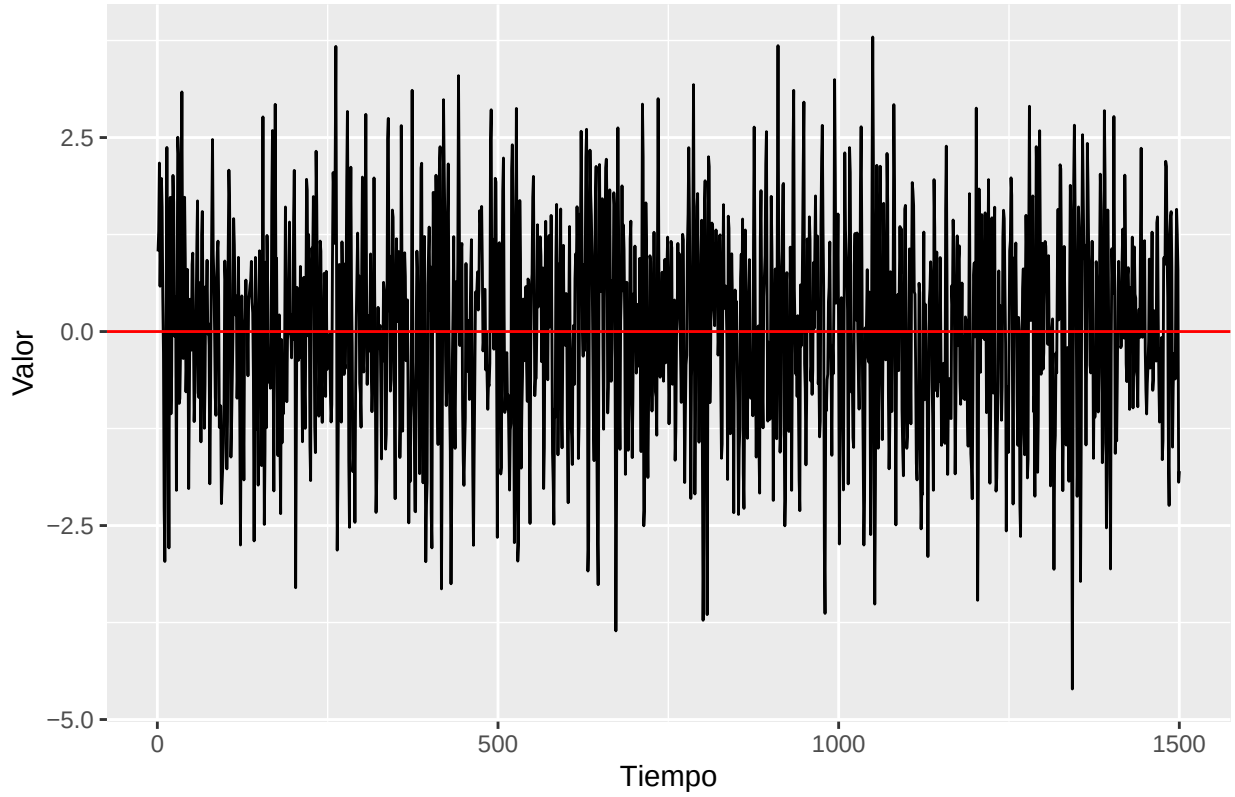

Figura 17: Simulación de un proceso AR(2) con coeficientes -0,7 y 0,5.

```r
# FAC
acf_simula_ar2_1 <- ggAcf(simula_ar2_1, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")

# FACP
pacf_simula_ar2_1 <- ggAcf(simula_ar2_1, lag.max = 24, type = "partial") +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       title = "")

grid.arrange(acf_simula_ar2_1, pacf_simula_ar2_1)
```

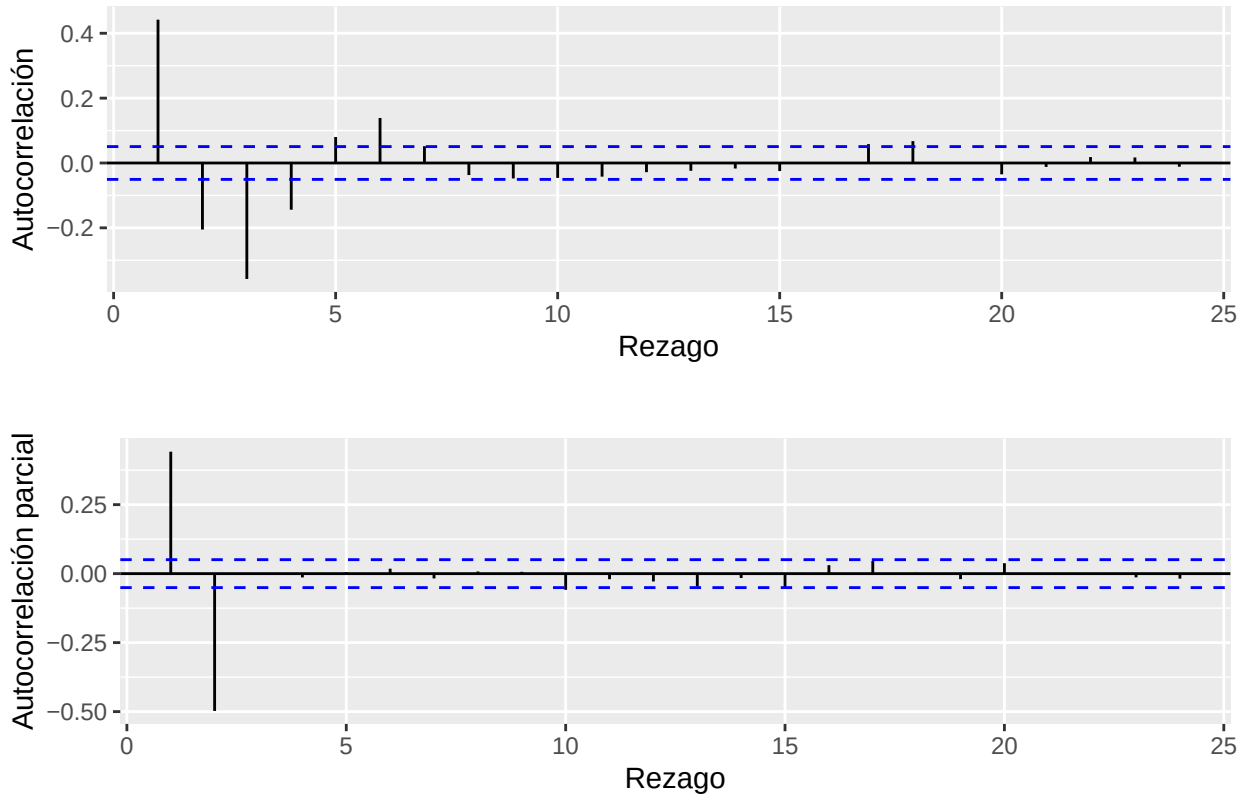

Figura 18: Funciones de Autocorrelación y Autocorrelación Parcial para un proceso AR(2) con coeficientes 0,7 y -0,5.

```
# Simulamos un proceso AR(2) con phi1 = -0,7 y phi2 = -0,5
simula_ar2_2 <- arima.sim(n = 1500, list(ar = c(-0.7, -0.5)))

# Chequeamos estacionariedad
# La función Mod() obtiene el valor absoluto y polyroot() las raíces del polinomio
Mod(polyroot(c(1, 0.7, 0.5)))
```

```
## [1] 1.414214 1.414214
```

```
# Graficamos el proceso
autoplot(simula_ar2_2) +
  geom_hline(yintercept = 0, col = "red") +
  labs(x = "Tiempo",
       y = "Valor")
```

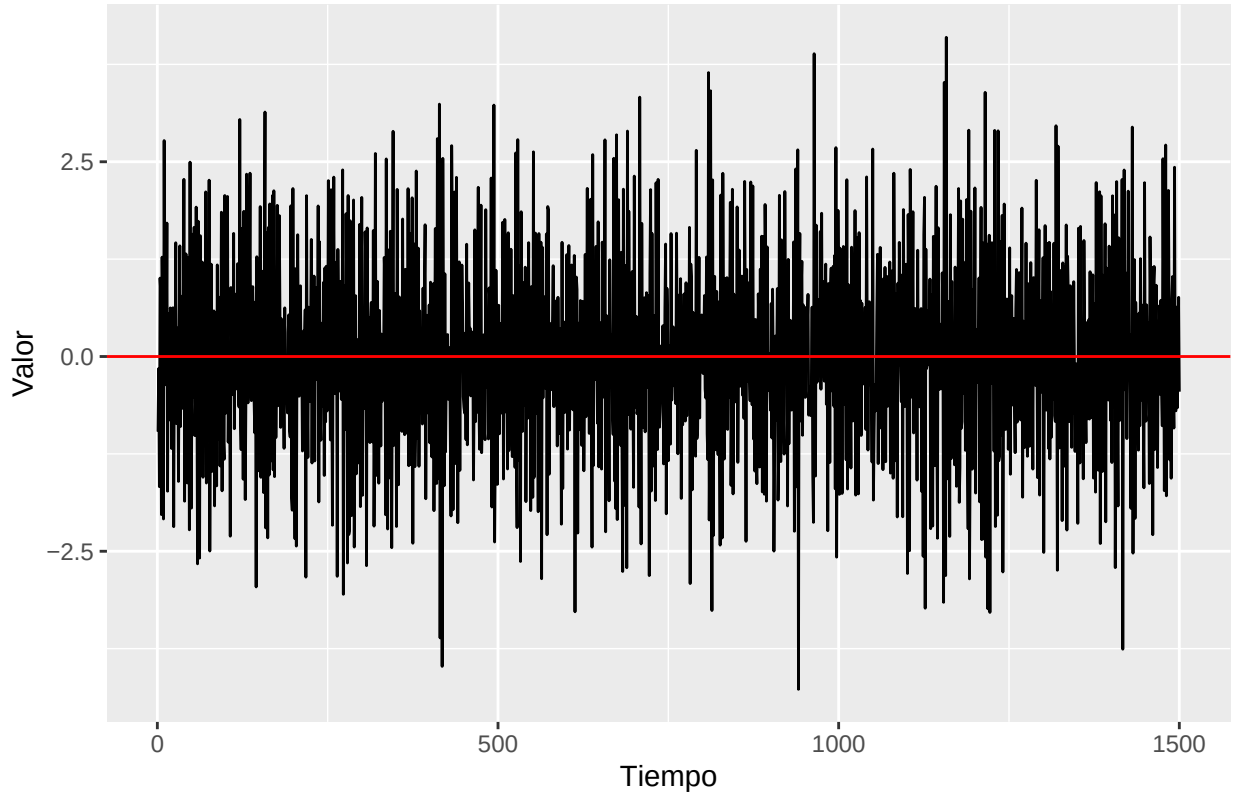

Figura 19: Simulación de un proceso AR(2) con coeficientes -0,7 y -0,5.

```
# FAC
acf_simula_ar2_2 <- ggAcf(simula_ar2_2, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")

# FACP
pacf_simula_ar2_2 <- ggAcf(simula_ar2_2, lag.max = 24, type = "partial") +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       title = "")

grid.arrange(acf_simula_ar2_2, pacf_simula_ar2_2)
```

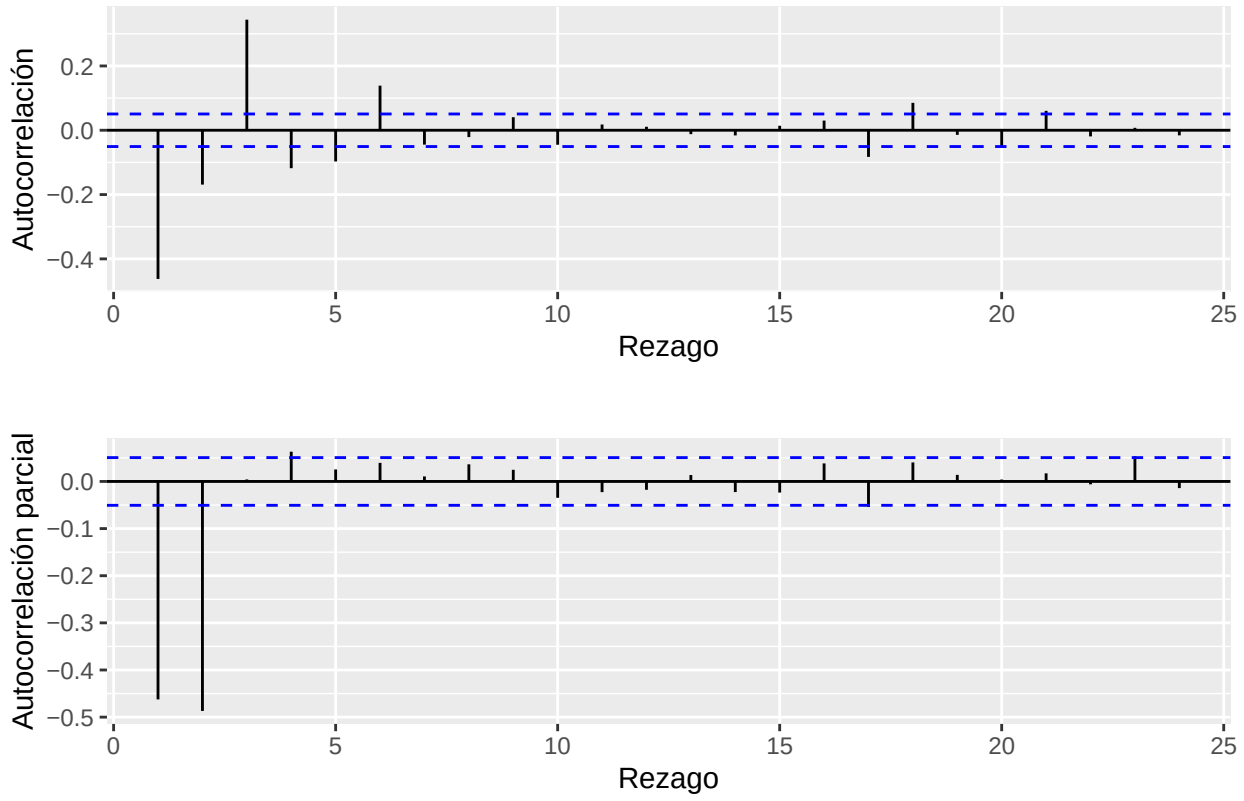

Figura 20: Funciones de Autocorrelación y Autocorrelación Parcial para un proceso AR(2) con coeficientes -0,7 y -0,5.

1.6. Procesos ARMA(1,1)

Los procesos ARMA contienen un componente autorregresivo y otro de medias móviles. En particular, un modelo ARMA(1,1) tiene la forma:

$$y_t = \phi y_{t-1} + \varepsilon_t - \theta \varepsilon_{t-1}$$

siendo ε_t un Ruido Blanco.

Para corroborar que el proceso sea estacionario, se debe verificar que el polinomio de su parte autorregresiva, $1 - \phi L$, tenga raíces mayores a uno en valor absoluto.

```
# Simulamos un proceso ARMA(1,1) con phi = 0,5 y theta = -0,5
simula_arma_1 <- arima.sim(list(ar = 0.5, ma = 0.5), n = 1500)
```

```
# Graficamos el proceso
autoplot(simula_arma_1) +
  geom_hline(yintercept = 0, col = "red") +
  labs(x = "Tiempo",
       y = "Valor")
```

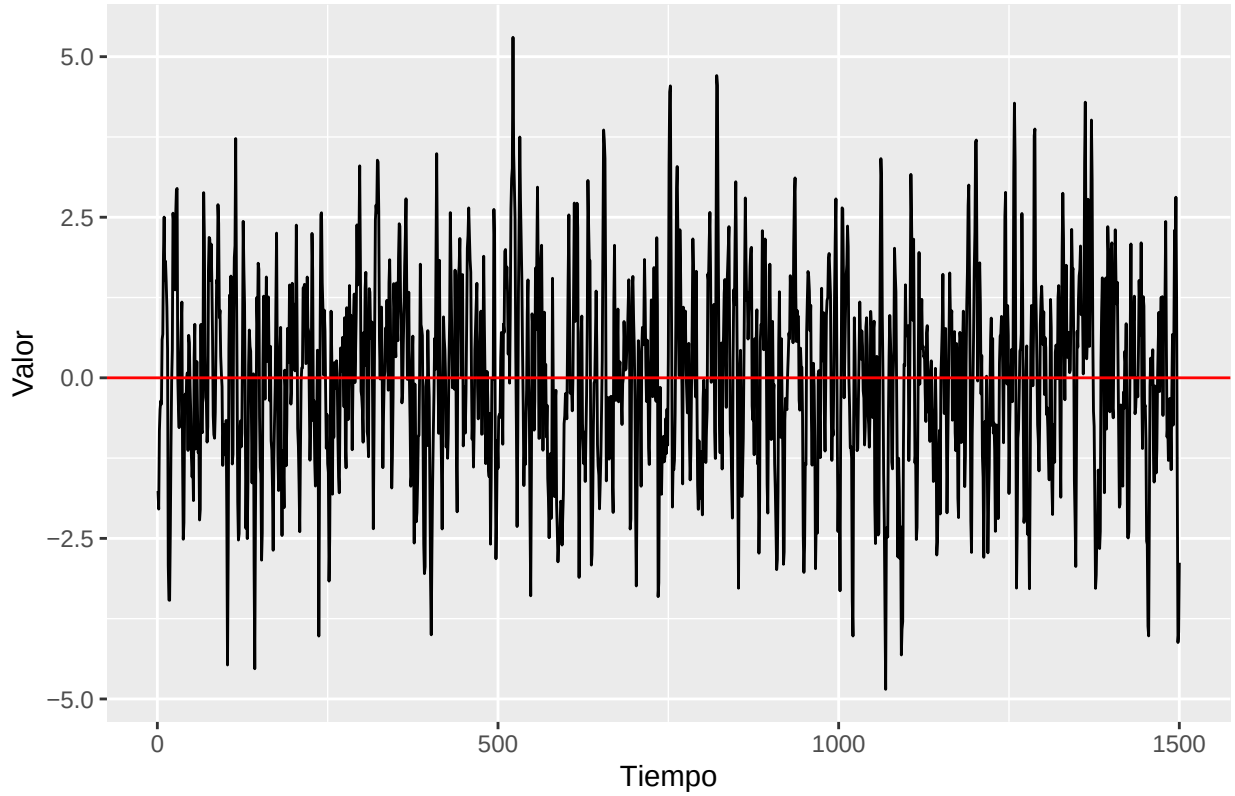

Figura 21: Simulación de un proceso ARMA(1,1) con coeficientes 0,5 y -0,5.

```
# FAC
acf_simula_arma_1 <- ggAcf(simula_arma_1, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")

# FACP
pacf_simula_arma_1 <- ggAcf(simula_arma_1, lag.max = 24, type = "partial") +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       title = "")

grid.arrange(acf_simula_arma_1, pacf_simula_arma_1)
```

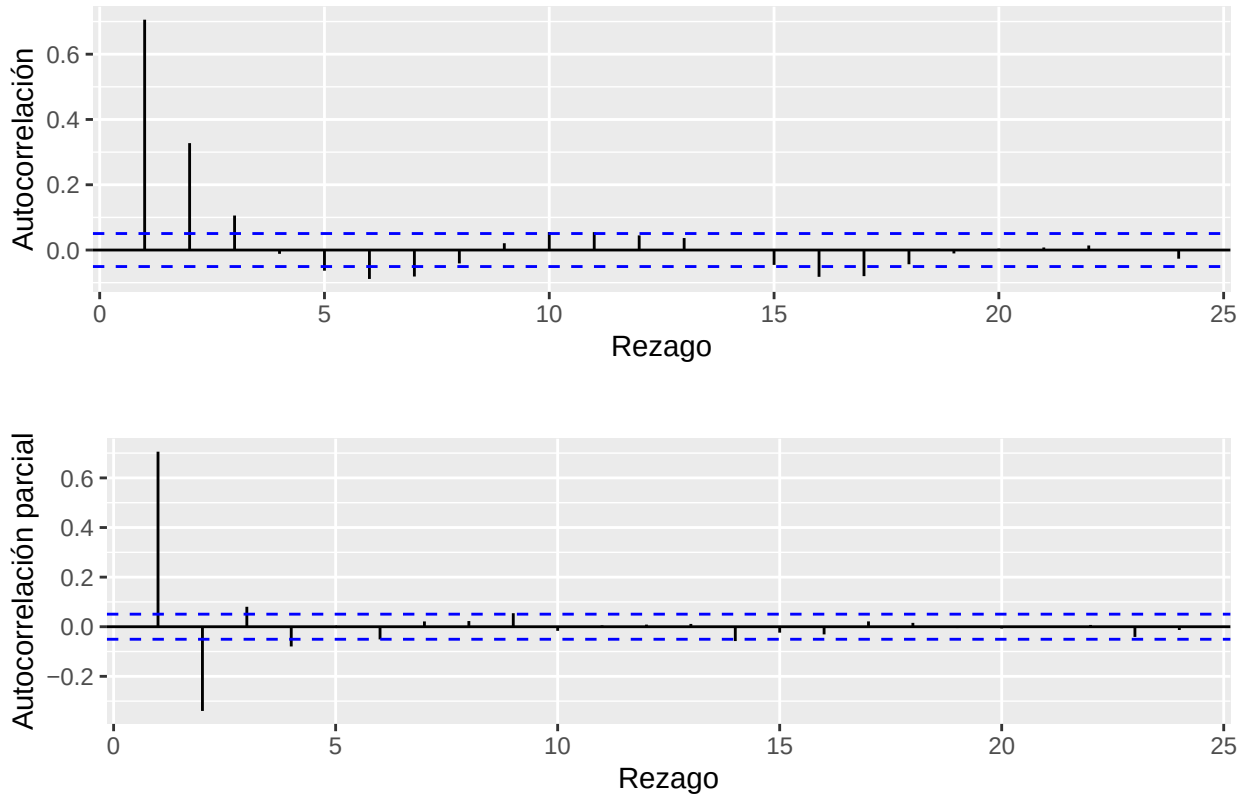

Figura 22: Funciones de Autocorrelación y Autocorrelación Parcial para un proceso ARMA(1,1) con coeficientes 0,5 y -0,5.

```
# Simulamos un proceso ARMA(1,1) con phi = -0,5 y theta = -0,5
simula_arma_2 <- arima.sim(list(ar = -0.5, ma = 0.5), n = 1500)
```

```
# Graficamos el proceso
autoplot(simula_arma_2) +
  geom_hline(yintercept = 0, col = "red") +
  labs(x = "Tiempo",
       y = "Valor")
```

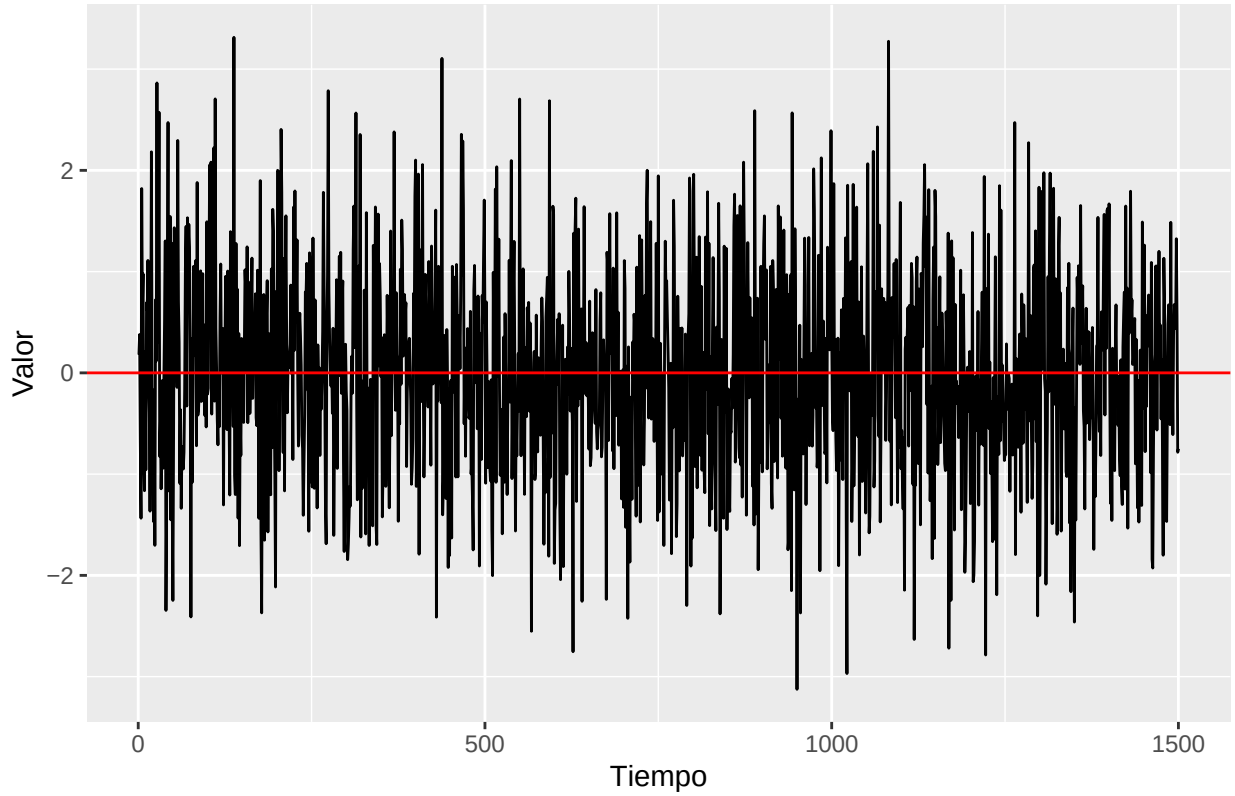

Figura 23: Simulación de un proceso ARMA(1,1) con coeficientes -0,5 y -0,5.

```
# FAC
acf_simula_arma_2 <- ggAcf(simula_arma_2, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")

# FACP
pacf_simula_arma_2 <- ggAcf(simula_arma_2, lag.max = 24, type = "partial") +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       title = "")

grid.arrange(acf_simula_arma_2, pacf_simula_arma_2)
```

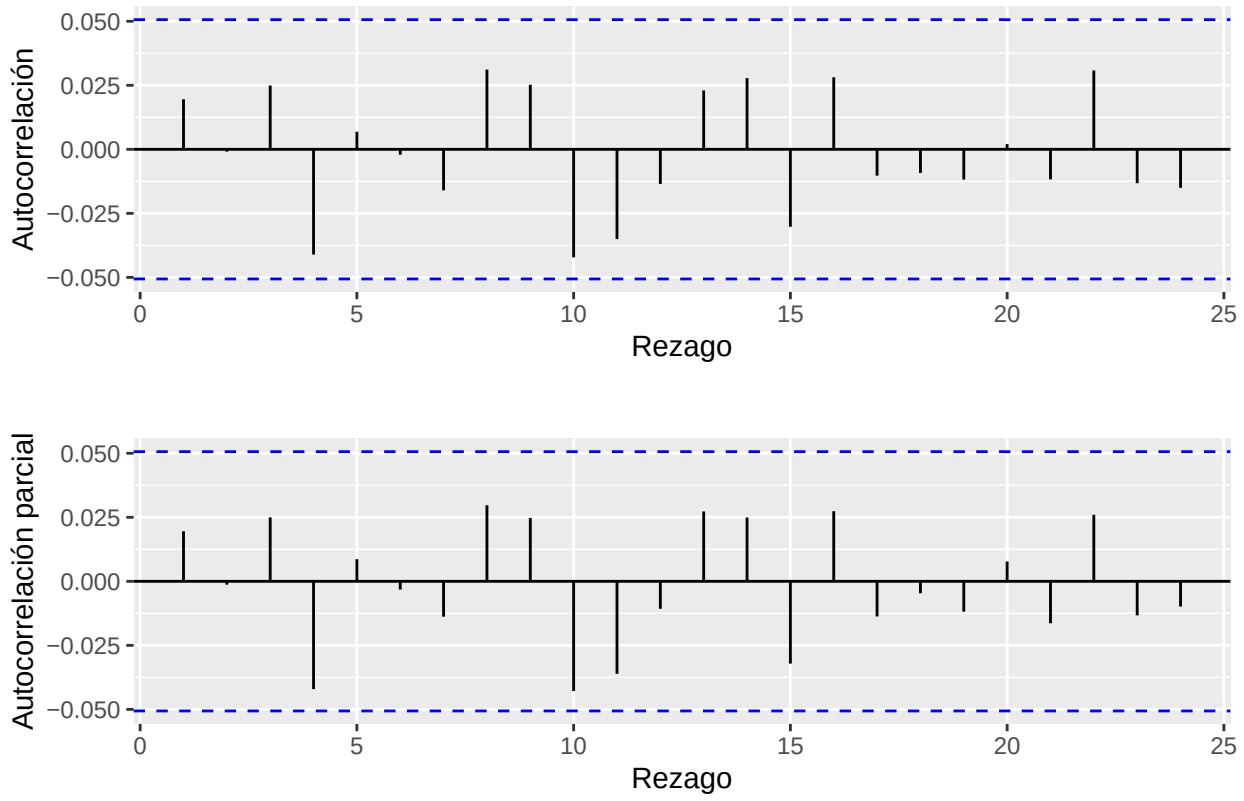

Figura 24: Funciones de Autocorrelación y Autocorrelación Parcial para un proceso ARMA(1,1) con coeficientes -0,5 y -0,5.

2. ARMA(p,q) como MA de orden infinito

Si un proceso ARMA(p,q) es estacionario, puede ser expresado como una combinación lineal de shocks infinitos, de forma que equivale a un proceso de medias móviles de orden infinito:

$$y_t = \varepsilon_t + \psi_1\varepsilon_{t-1} + \psi_2\varepsilon_{t-2} + \ldots = (1 + \psi_1 L + \psi_2 L^2 + \ldots)\varepsilon_t = \Psi(L)\varepsilon_t$$

Por otra parte, se tiene que:

$$\Phi(L)y_t = \Theta(L)\varepsilon_t \Rightarrow y_t = \frac{\Theta(L)}{\Phi(L)}\varepsilon_t$$

De esta manera, se concluye que:

$$\frac{\Theta(L)}{\Phi(L)} = \Psi(L) \Rightarrow \Theta(L) = \Psi(L)\Phi(L)$$

A partir de esta expresión, es posible despejar los valores del polinomio $\Psi(L)$ para cada proceso particular.

```
# Expresamos un proceso AR(1) con phi = 0.8 como un MA(inf)
ma_inf_1 <- ARMAtoMA(ar = 0.8, lag.max = 40)
```

```r
ma_inf_1 <- data.frame(Tiempo = 1:40, Coeficientes = ma_inf_1)
head(ma_inf_1)
```

```
##   Tiempo Coeficientes
## 1      1     0.800000
## 2      2     0.640000
## 3      3     0.512000
## 4      4     0.409600
## 5      5     0.327680
## 6      6     0.262144
```

```r
# Graficamos los primeros 40 coeficientes del proceso expresado como un MA(inf)

ggplot(ma_inf_1) +
  geom_line(aes(x = Tiempo, y = Coeficientes))
```

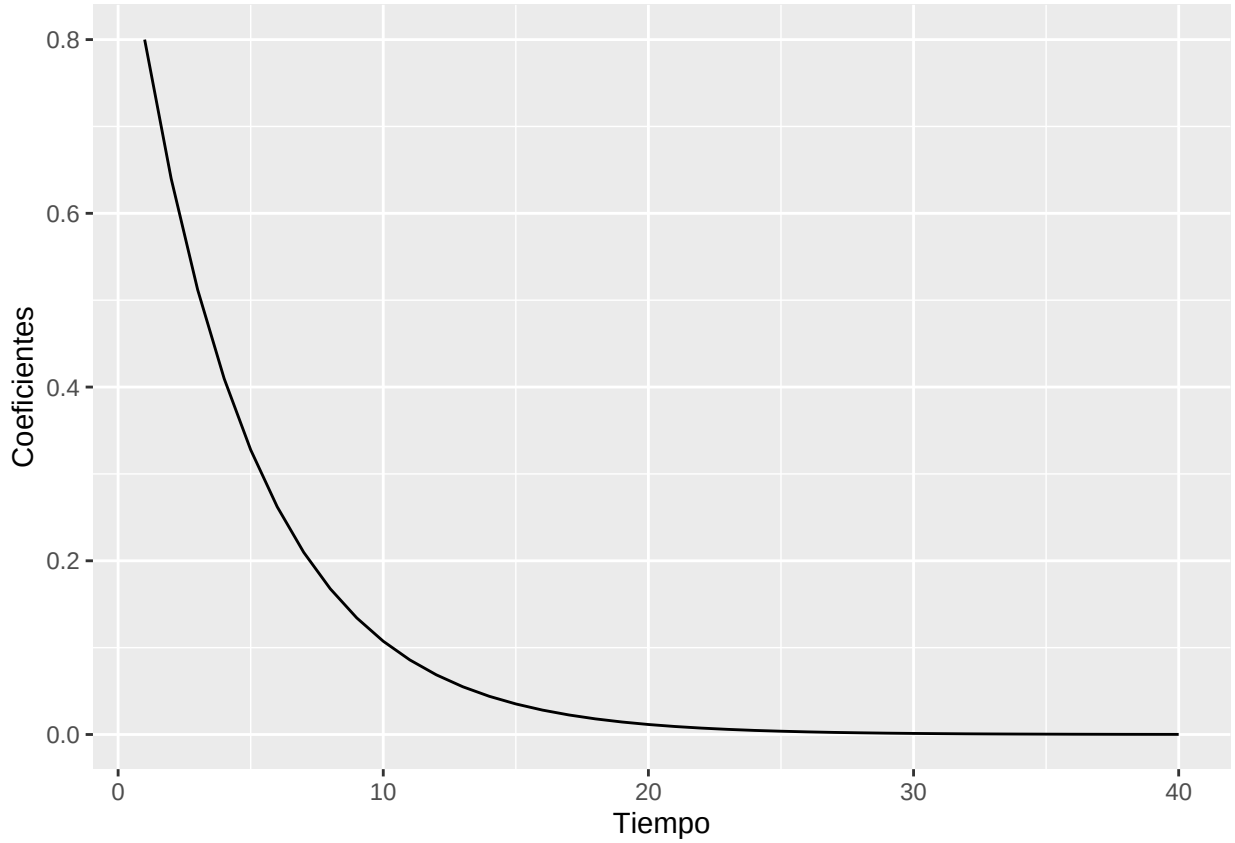

Figura 25: Primeros 40 coeficientes de la representación como un proceso MA de orden infinito de un proceso AR(1).

```r
# Expresamos un proceso AR(2) con phi1 = 0.7 y phi2 = -0,5 como un MA(inf)
ma_inf_2 <- ARMAtoMA(ar = c(0.7, -0.5), lag.max = 40)
ma_inf_2 <- data.frame(Tiempo = 1:40, Coeficientes = ma_inf_2)
head(ma_inf_2)
```

```
##   Tiempo Coeficientes
## 1      1     0.700000
## 2      2    -0.010000
## 3      3    -0.357000
## 4      4    -0.244900
## 5      5     0.007070
## 6      6     0.127399
```

```r
# Graficamos los primeros 40 coeficientes del proceso expresado como un MA(inf)

ggplot(ma_inf_2) +
  geom_line(aes(x = Tiempo, y = Coeficientes))
```

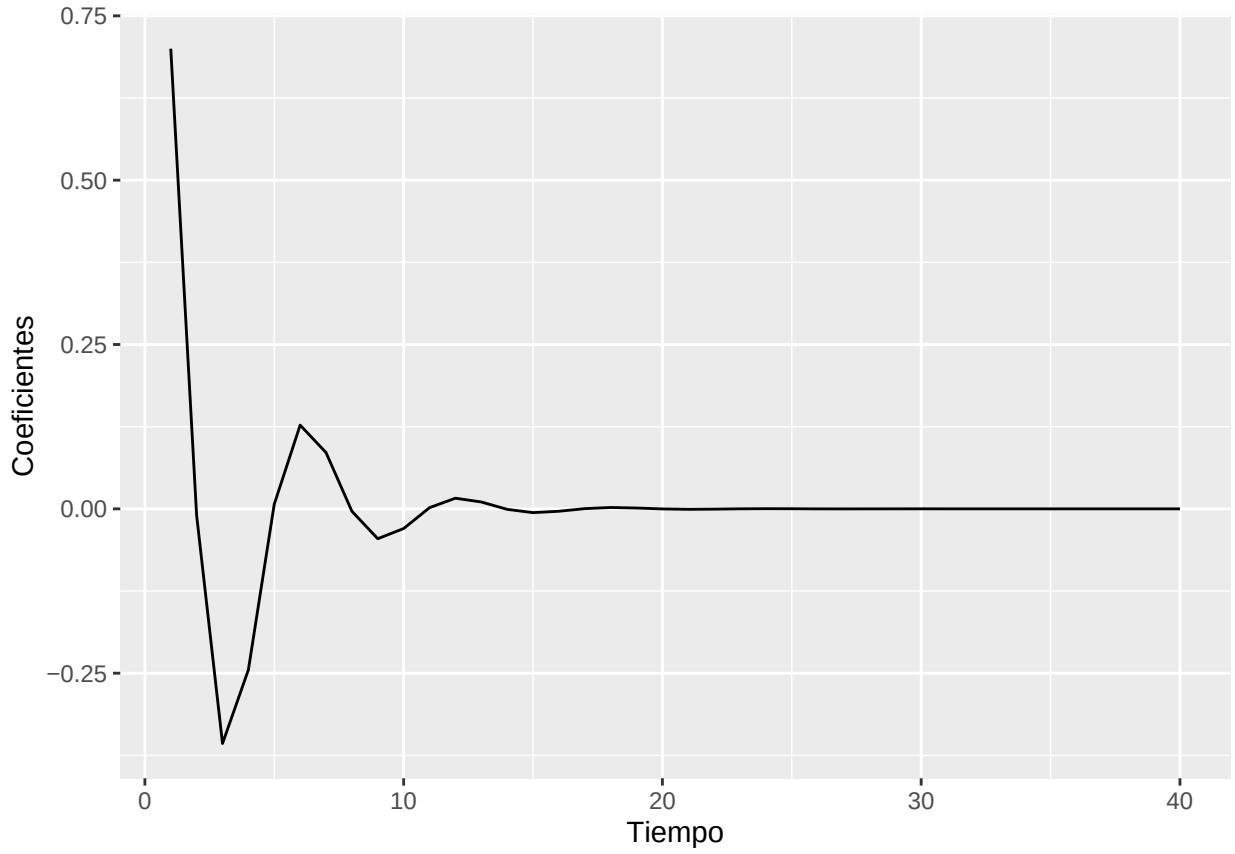

Figura 26: Primeros 40 coeficientes de la representación como un proceso MA de orden infinito de un proceso AR(2).

3. Identificación de procesos con pocas observaciones

3.1. Procesos AR(1)

```r
# Simulamos un proceso AR(1) con phi = 0,7 y 50 observaciones
simula_ar1_n50 <- arima.sim(list(ar = 0.7), n = 50)
```

```r
# Graficamos el proceso
autoplot(simula_ar1_n50) +
  geom_hline(yintercept = 0, col = "red") +
  labs(x = "Tiempo",
       y = "Valor")
```

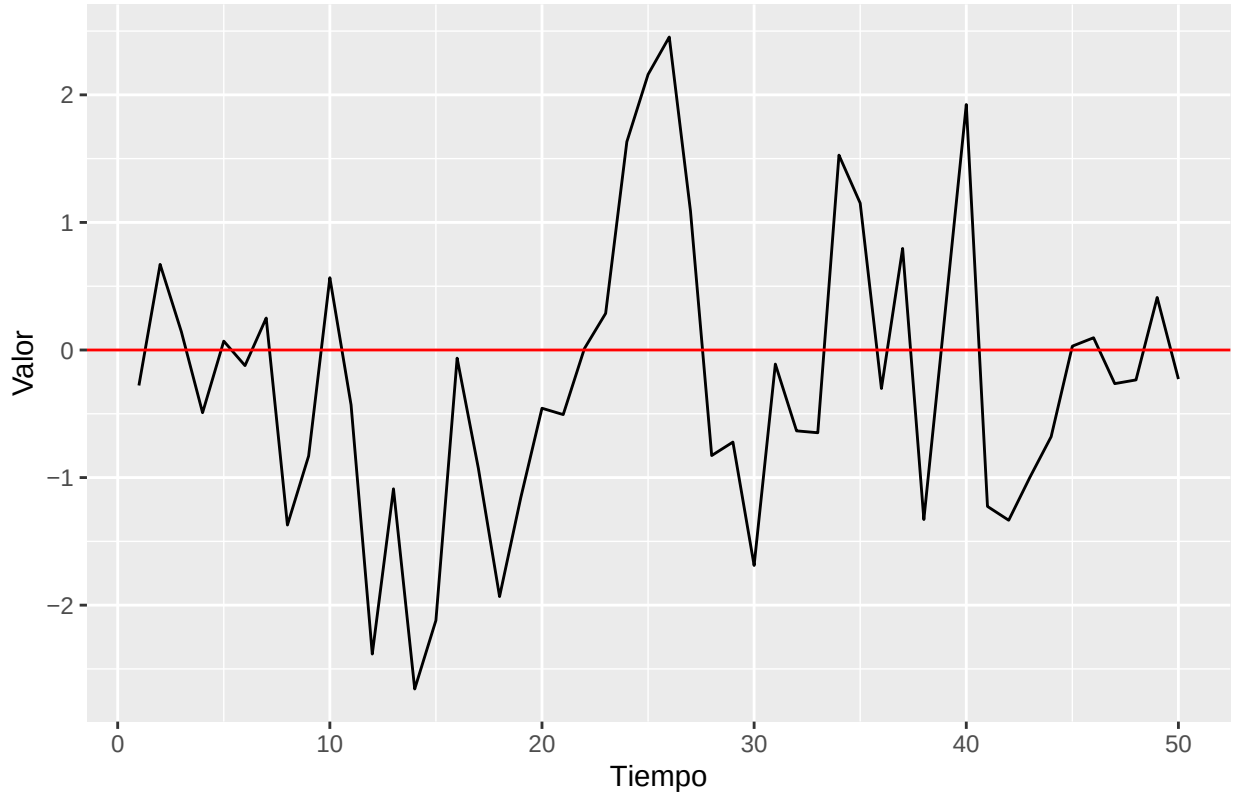

Figura 27: Simulación de 50 observaciones para un proceso AR(1).

```r
# FAC
acf_simula_ar1_n50 <- ggAcf(simula_ar1_n50, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")

# FACP
pacf_simula_ar1_n50 <- ggAcf(simula_ar1_n50, lag.max = 24, type = "partial") +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       title = "")

grid.arrange(acf_simula_ar1_n50, pacf_simula_ar1_n50)
```

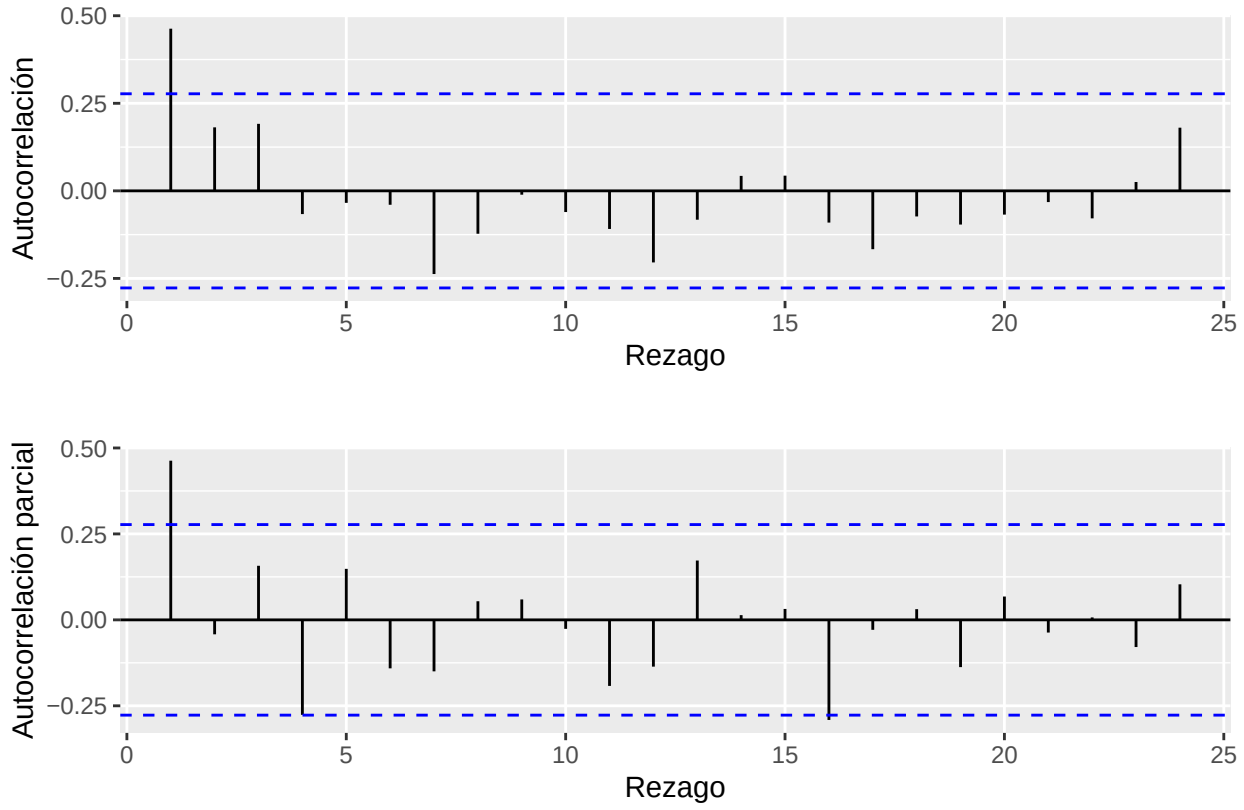

Figura 28: Funciones de Autocorrelación y Autocorrelación Parcial para 50 observaciones de un proceso AR(1).

```
# Obtenemos las autocorrelaciones teóricas
ar1_n50_acf_teorico <- ARMAacf(ar = 0.7, lag.max = 14, pacf = FALSE)
ar1_n50_pacf_teorico <- ARMAacf(ar = 0.7, lag.max = 14, pacf = TRUE)

# Obtenemos las autocorrelaciones estimadas
ar1_n50_acf_est <- ggAcf(simula_ar1_n50, plot = FALSE, lag.max = 14, type = "correlation")
ar1_n50_pacf_est <- ggAcf(simula_ar1_n50, plot = FALSE, lag.max = 14, type = "partial")

# Ordenamos los datos de la FAC para poder graficarlos
ar1_n50_teorico_est_acf <- data.frame(rezago = 0:14, "Teórica" = ar1_n50_acf_teorico, "Estimación" = ar
ar1_n50_teorico_est_acf <- ar1_n50_teorico_est_acf %>%
  pivot_longer(cols = c("Teórica", "Estimación"), values_to = "valores")

# Graficamos las FAC teórica y estimada
ar1_n50_acf <- ggplot(ar1_n50_teorico_est_acf) +
  geom_col(aes(x= rezago, y = valores, fill = name),
           position = "dodge", width = 0.2) +
  labs(x = "Rezago",
       y = "Autocorrelación",
       fill = "FAC")

# Ordenamos los datos de la FACP para poder graficarlos
```

```r
ar1_n50_teorico_est_pacf <- data.frame(rezago = 1:14, "Teórica" = ar1_n50_pacf_teorico, "Estimación" = a
ar1_n50_teorico_est_pacf <- ar1_n50_teorico_est_pacf %>%
  pivot_longer(cols = c("Teórica", "Estimación"), values_to = "valores")

# Graficamos las FACP teórica y estimada
ar1_n50_pacf <- ggplot(ar1_n50_teorico_est_pacf) +
  geom_col(aes(x= rezago, y = valores, fill = name),
           position = "dodge", width = 0.2) +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       fill = "FACP")

grid.arrange(ar1_n50_acf, ar1_n50_pacf)
```

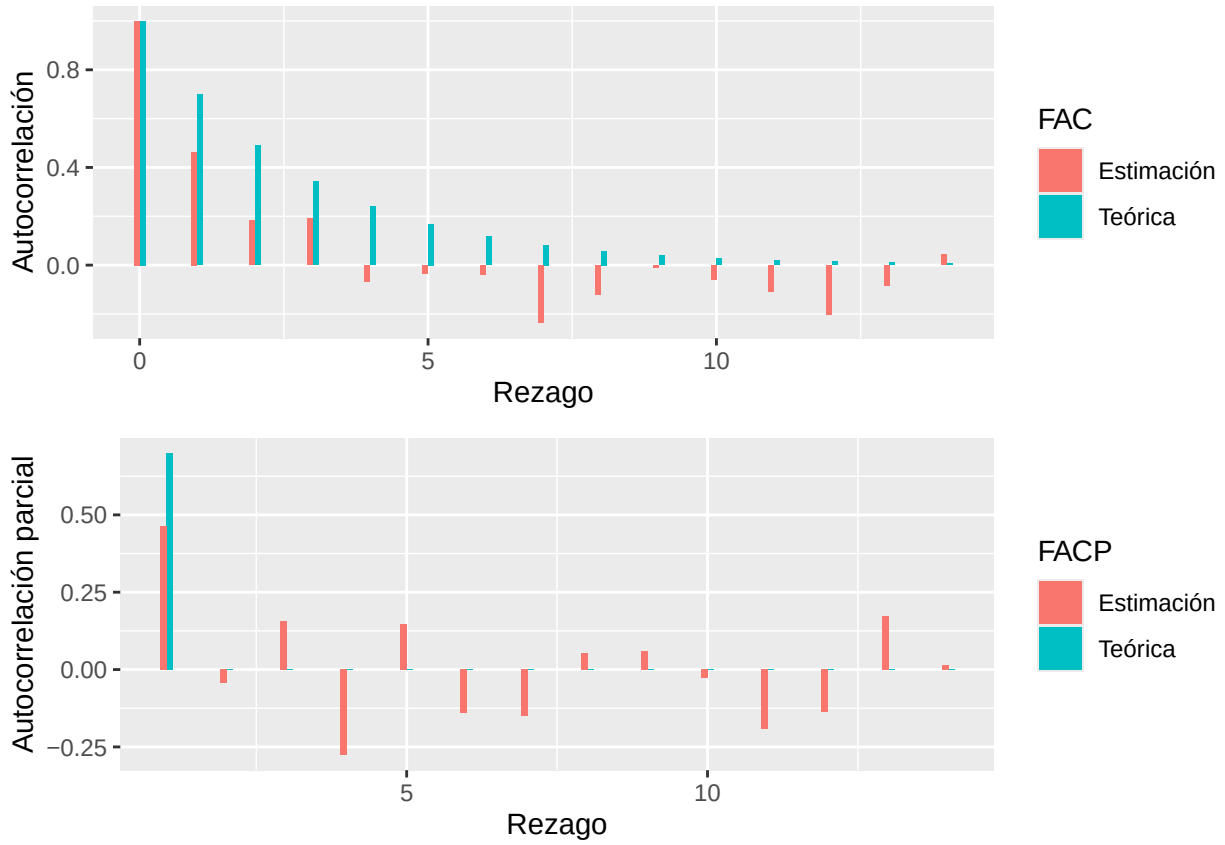

Figura 29: Funciones de Autocorrelación y Autocorrelación Parcial teóricas y estimadas para 50 observaciones de un proceso AR(1) con coeficiente positivo.

3.2. Procesos MA(1)

```r
# Simulamos un proceso MA(1) con theta = 0,8 y 50 observaciones
# En la función arima.sim(), el signo del coeficiente está invertido
simula_ma_n50 <- arima.sim(list(ma = -0.8), n = 50)
```

```r
# Graficamos el proceso
autoplot(simula_ma_n50) +
  geom_hline(yintercept = 0, col = "red") +
  labs(x = "Tiempo",
       y = "Valor")
```

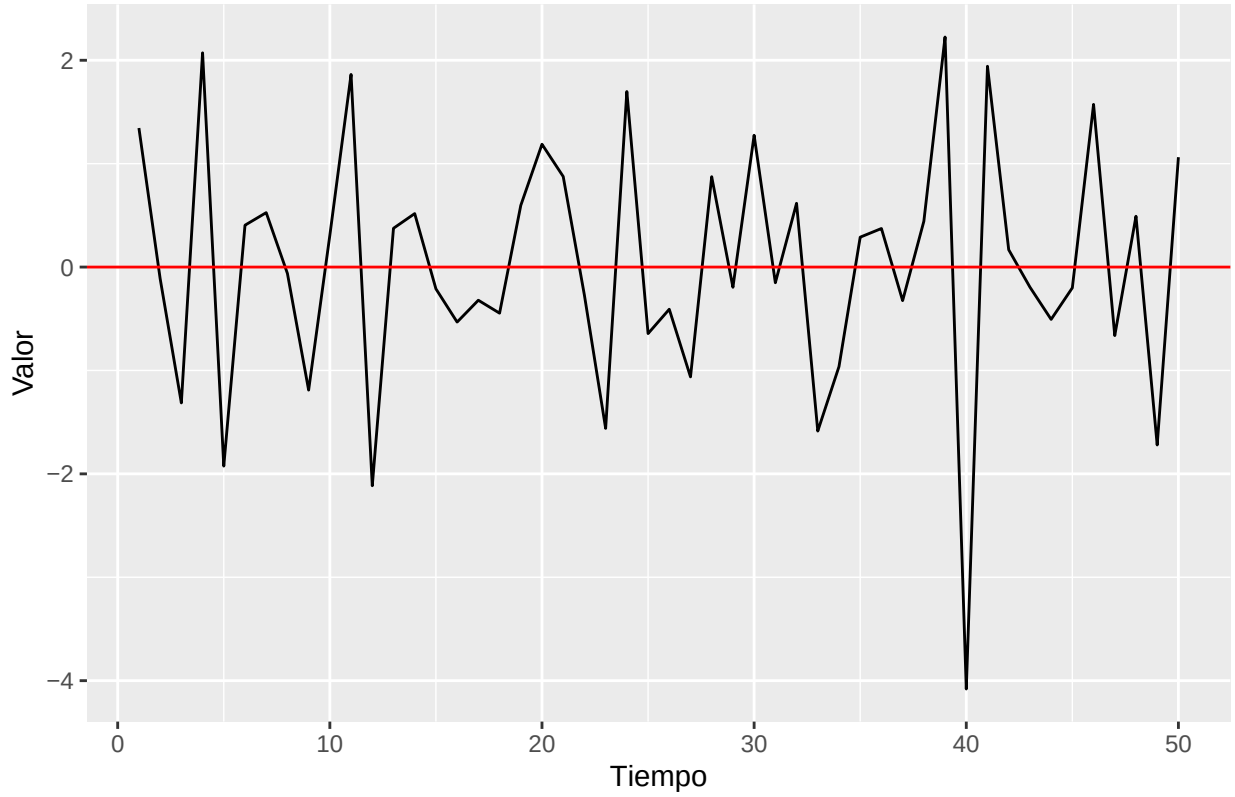

Figura 30: Simulación de 50 observaciones para un proceso MA(1).

```r
# FAC
acf_simula_ma_n50 <- ggAcf(simula_ma_n50, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")

# FACP
pacf_simula_ma_n50 <- ggAcf(simula_ma_n50, lag.max = 24, type = "partial") +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       title = "")

grid.arrange(acf_simula_ma_n50, pacf_simula_ma_n50)
```

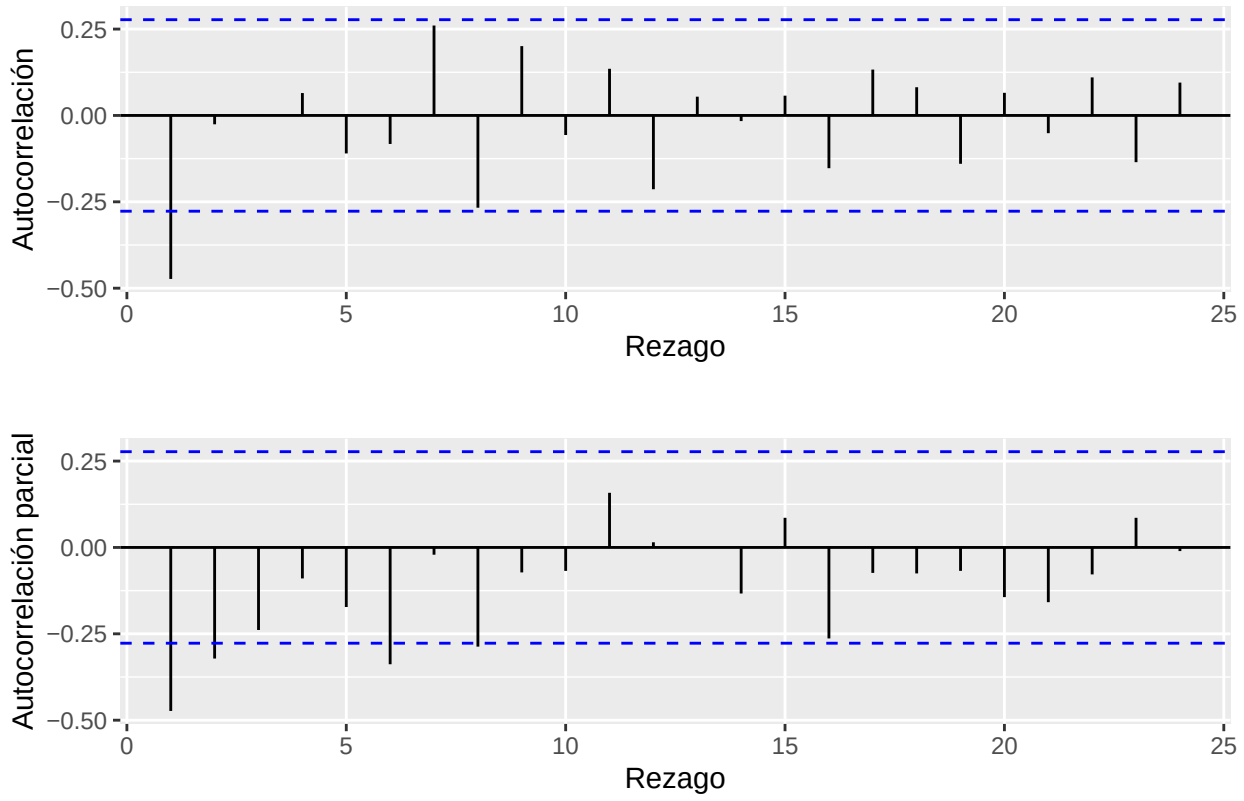

Figura 31: Funciones de Autocorrelación y Autocorrelación Parcial para 50 observaciones de un proceso MA(1).

3.3. Procesos AR(2)

```
# Simulamos 50 observaciones de un proceso AR(2) con phi1 = 0,7 y phi2 = -0,5
simula_ar2_n50 <- arima.sim(n = 50, list(ar = c(0.7, -0.5)))

# Chequeamos estacionariedad
# La función Mod() obtiene el valor absoluto y polyroot() las raíces del polinomio
Mod(polyroot(c(1, -0.7, 0.5)))
```

```
## [1] 1.414214 1.414214
```

```
# Graficamos el proceso
autoplot(simula_ar2_n50) +
  geom_hline(yintercept = 0, col = "red") +
  labs(x = "Tiempo",
       y = "Valor")
```

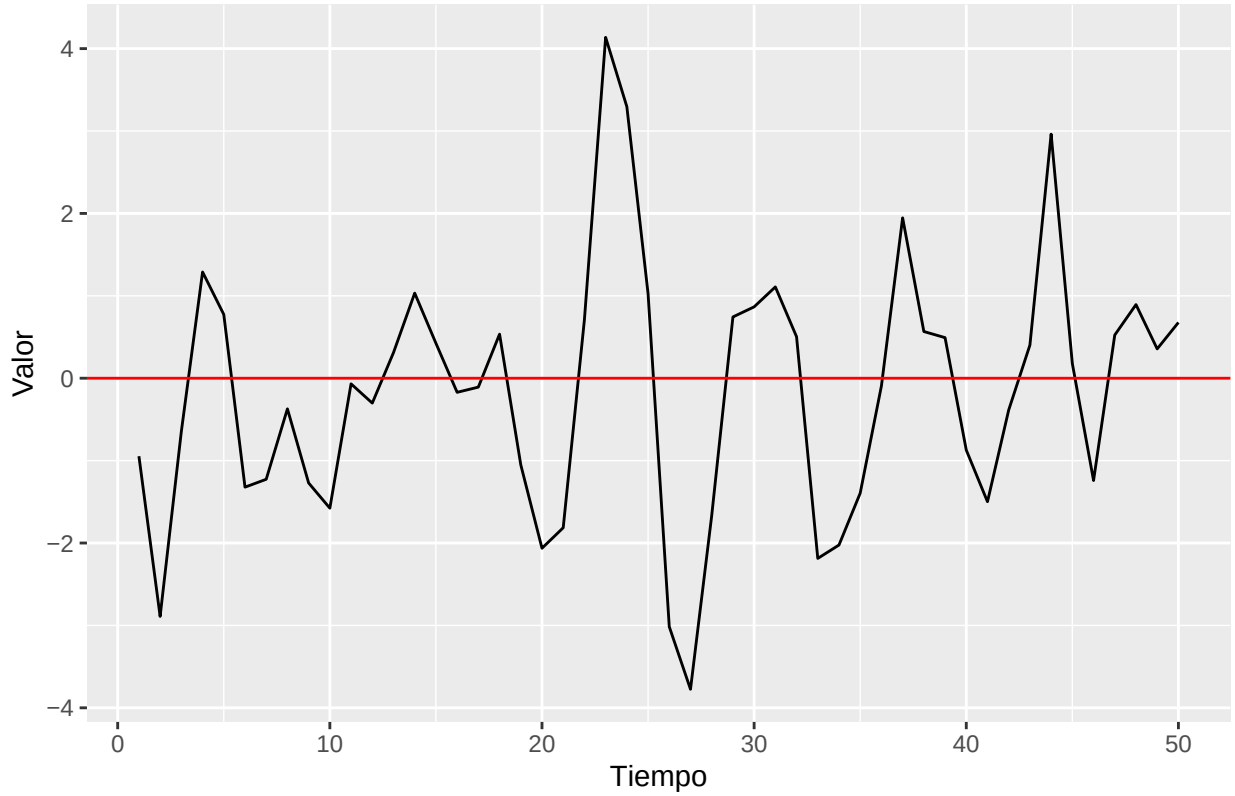

Figura 32: Simulación de 50 observaciones de un proceso AR(2).

```r
# FAC
acf_simula_ar2_n50 <- ggAcf(simula_ar2_n50, lag.max = 24, type = "correlation") +
  labs(x = "Rezago",
       y = "Autocorrelación",
       title = "")

# FACP
pacf_simula_ar2_n50 <- ggAcf(simula_ar2_n50, lag.max = 24, type = "partial") +
  labs(x = "Rezago",
       y = "Autocorrelación parcial",
       title = "")

grid.arrange(acf_simula_ar2_n50, pacf_simula_ar2_n50)
```

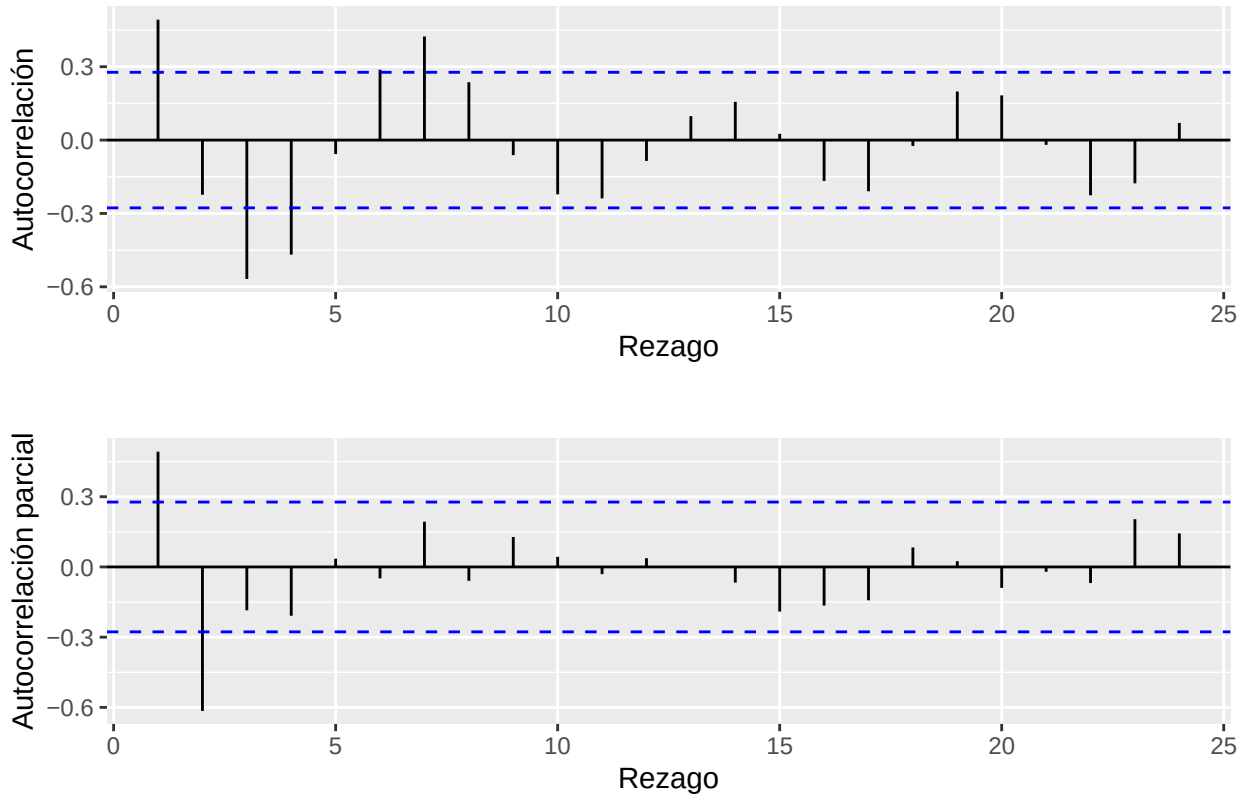

Figura 33: Funciones de Autocorrelación y Autocorrelación Parcial para 50 observaciones de un proceso AR(2).