Informe Assigment 2

ICAI. Machine Learning.

Álvaro Rodriguez y Pablo Sanz

Curso 2021-22. Última actualización: 2021-12-02

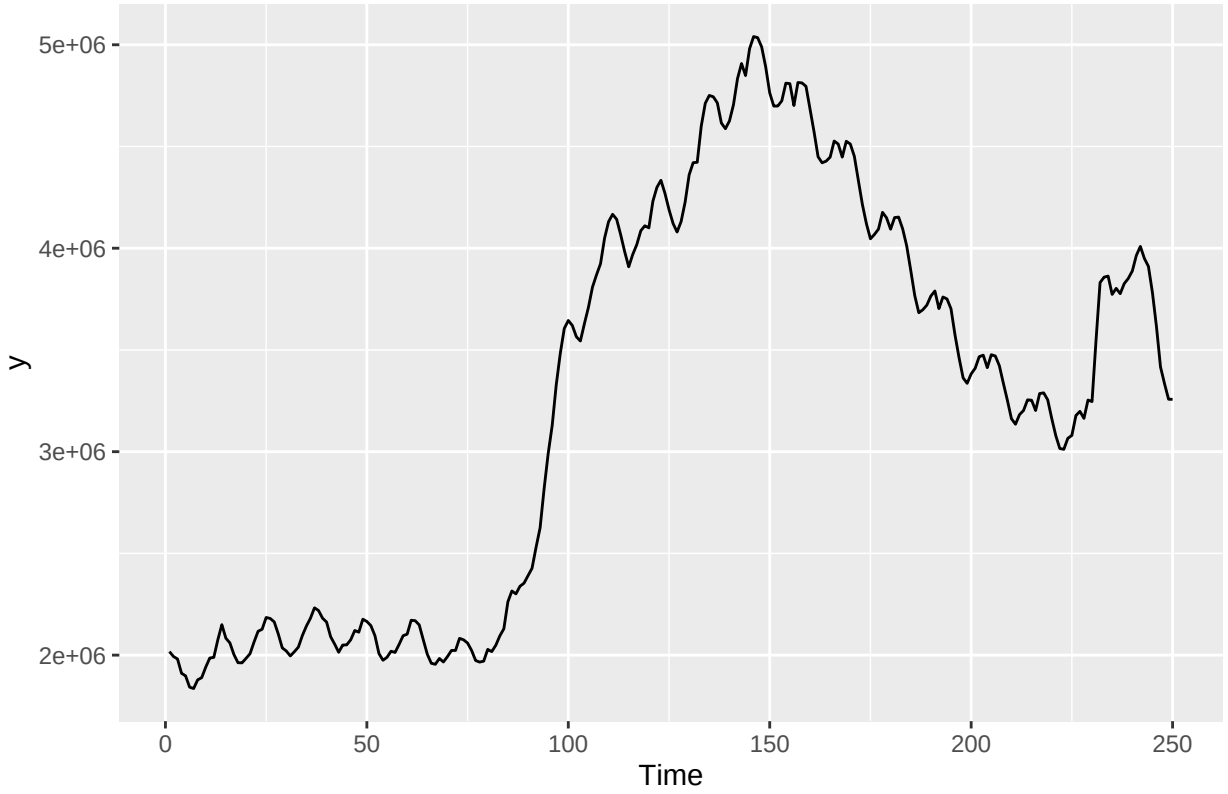
Índice

Presentación de los datos

Los datos a tratar pertenecen al número de desempleados de España a lo largo de los años. Como se puede ver en representación inferior, podemos observar diferentes épocas que ha sufrido el país en su historia más reciente.

Si empezamos por el extremo inferior que data sobre los inicios del año 2000, observamos que el empleo se mantiene más o menos constante en cuanto a tendencia, aunque se aprecia una gran estacionalidad según si estamos en los meses de invierno o verano. Este patrón se repite hasta el año 2008 donde con la gran crisis financiera mundial, el paro subió bruscamente situándose en más de 5.000.000 de personas en el año 2012. Pasado este momento, empieza a decrecer progresivamente observándose siempre aumentos y descensos según los meses de invierno o verano respectivamente hasta finales de 2019.

En este momento, a raíz de la aparción del COVID-19, la serie rompe todos los esquemas, aumentando en muy pocos meses casi 1.000.000 de desempleados debido a la crisis sanitaria y a todas las restricciones impuestas. Unos meses depués, estos altos niveles de desempleo empiezan a decrecer a gran ritmo hasta situarse en el momento actual donde el ritmo de bajada ha disminuido.

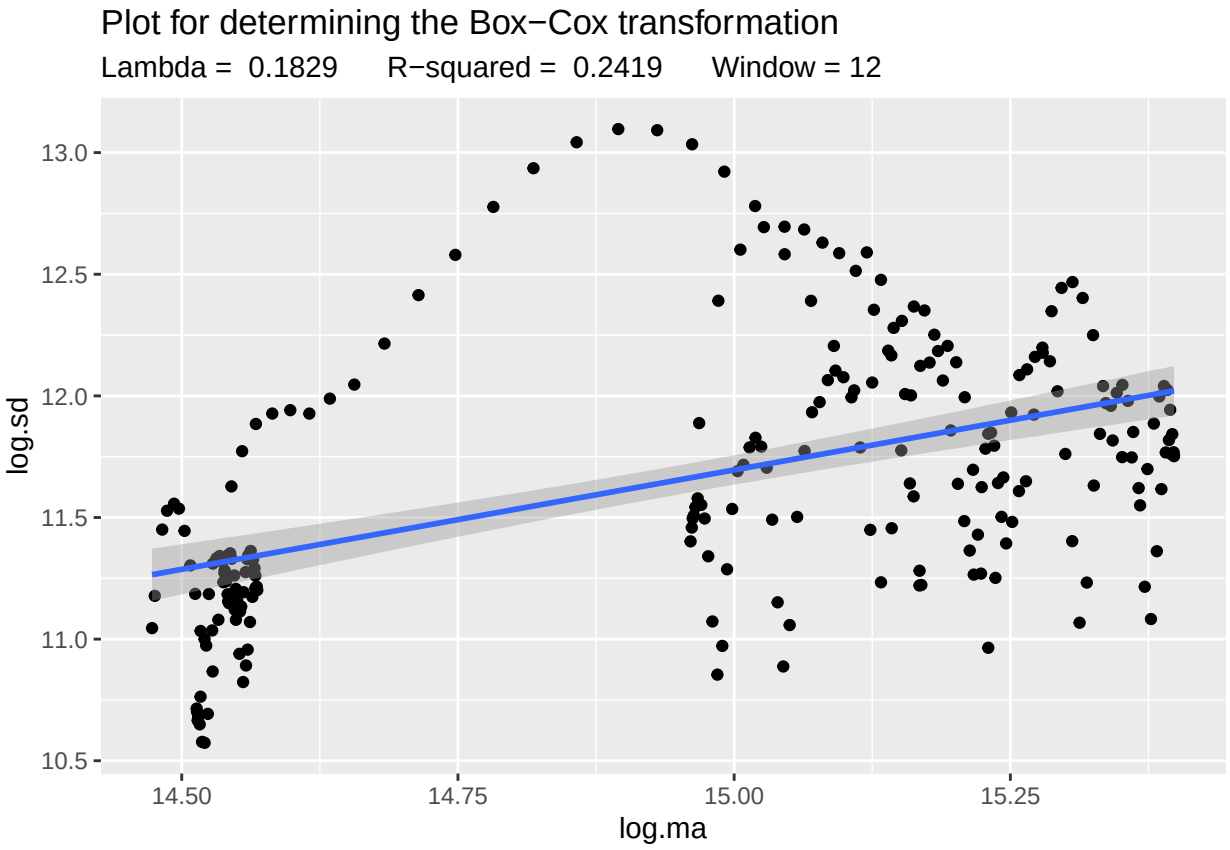


Predicción

En primer lugar explicaremos el proceso que nos llevó a dar como resultado que el paro en el mes de Novimembre es de **3.240.623**.

Análisis exploratorio

Lo primero que hicimos tras ver los datos del paro fue ver si la serie es estacionaria en varianza, es decir, que independientemente del nivel la varianza es constante. Esto se ve en el siguiente gráfico:



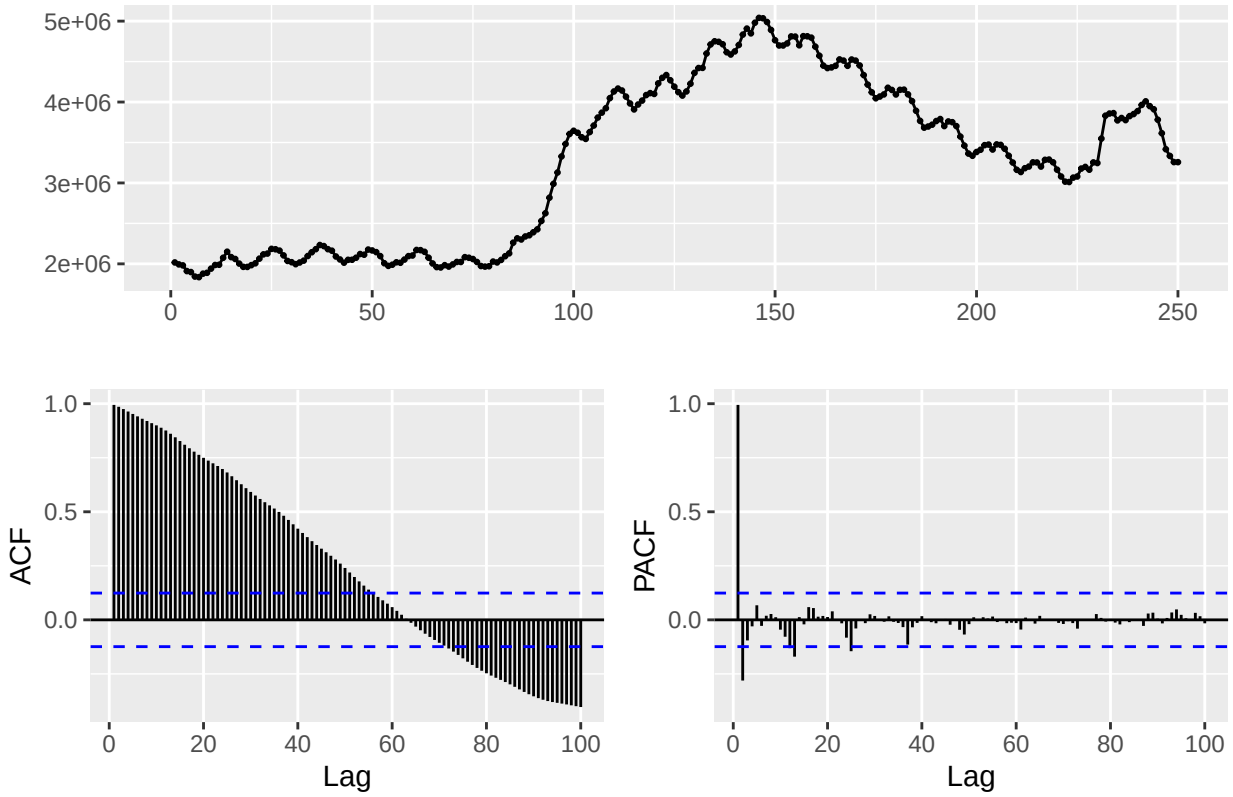

```
##    log.ma
## 0.1829074
```

Que refleja si es necesario realizar una transformación Box-Cox. Sin embargo, vemos que la línea azul no es muy creciente y que R-square no es muy alto (R-square=0.2419). Por lo tanto, no es necesario realizar una estabilización de varianza.

Seasonal ARIMA

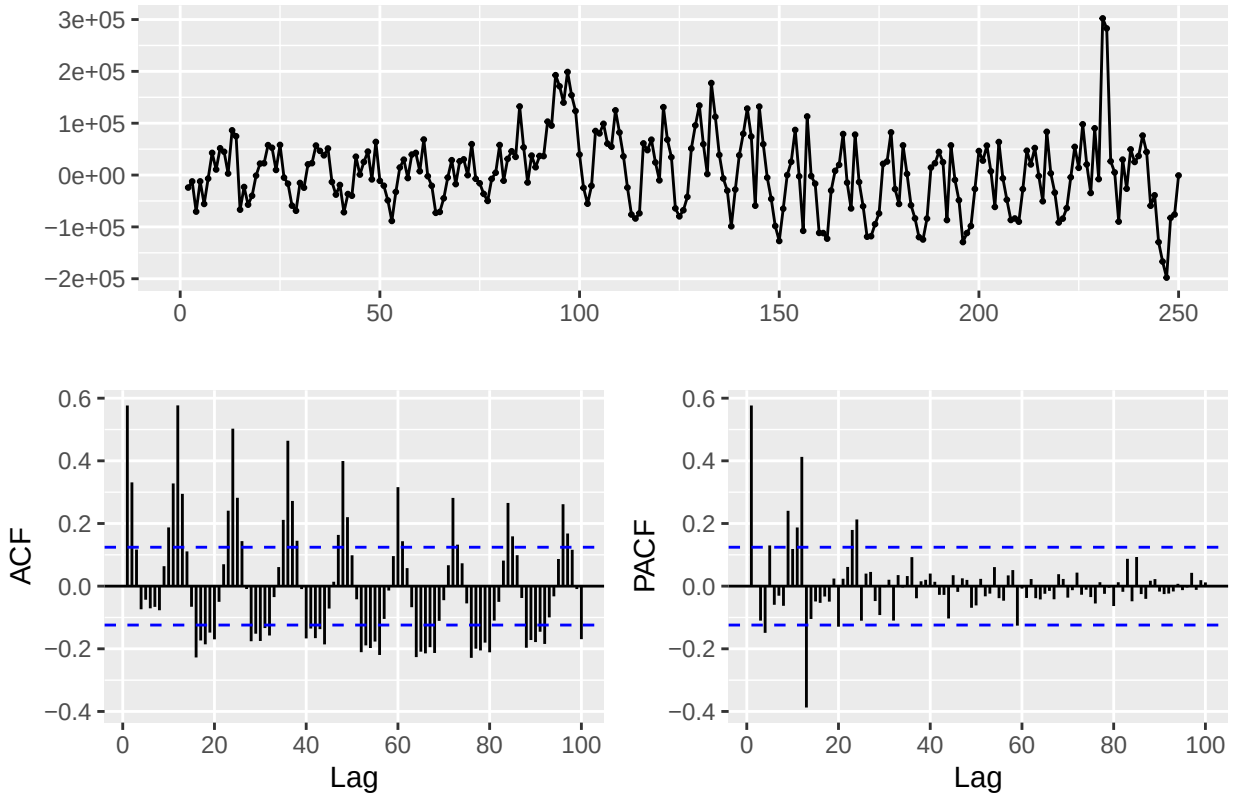
En primer lugar representamos la serie temporal junto con su ACF y PACF para inspeccionar la serie regular y estacional. Vemos que la serie necesita una diferenciación en la parte regular ya que el ACF disminuye lentamente a lo largo del tiempo.

```
ggtsdisplay(y,lag.max = 100)
```



Tras haber diferenciado en la parte regular, volvemos a mostrar el ACF y el PACF.

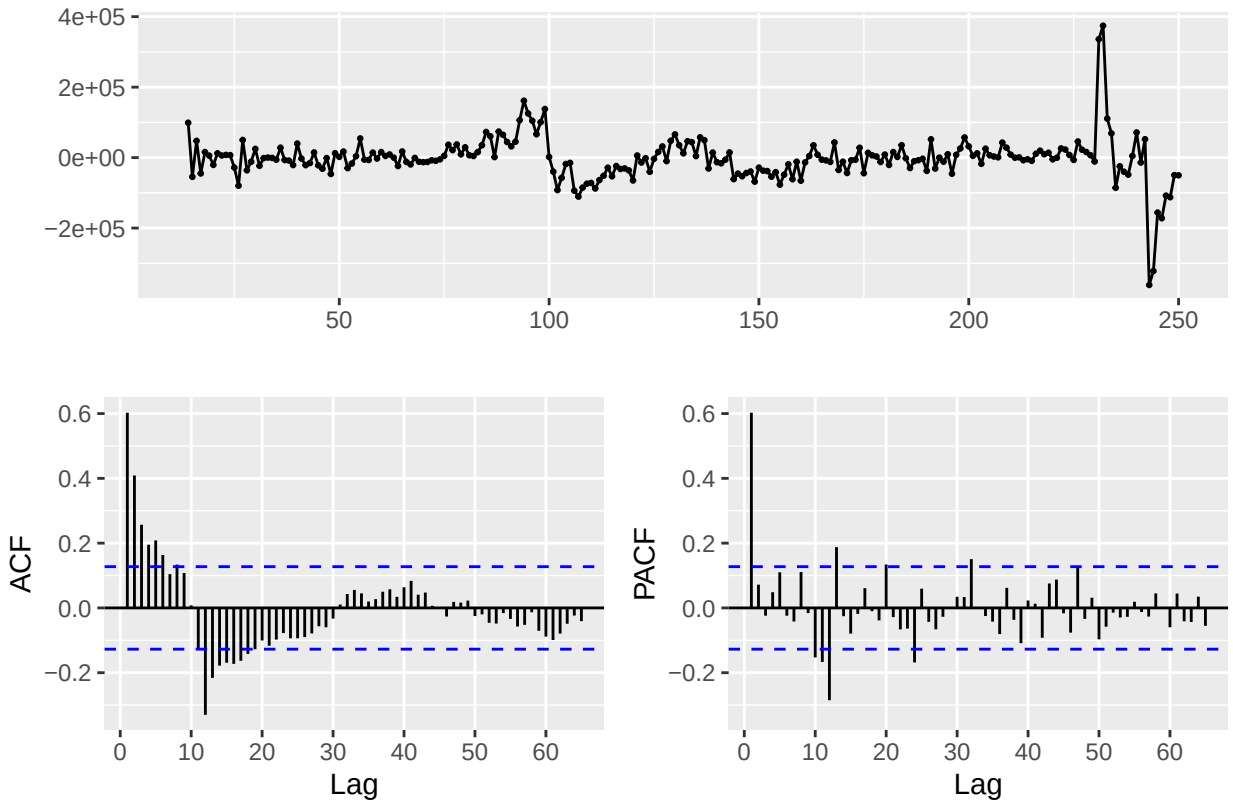
```
Bz <- diff(y,differences = 1)
ggtsdisplay(Bz,lag.max = 100)
```



En la imagen superior vemos que cada 12 puntos aparece un pico y por tanto deducimos que tenemos que diferenciar en la parte estacional también.

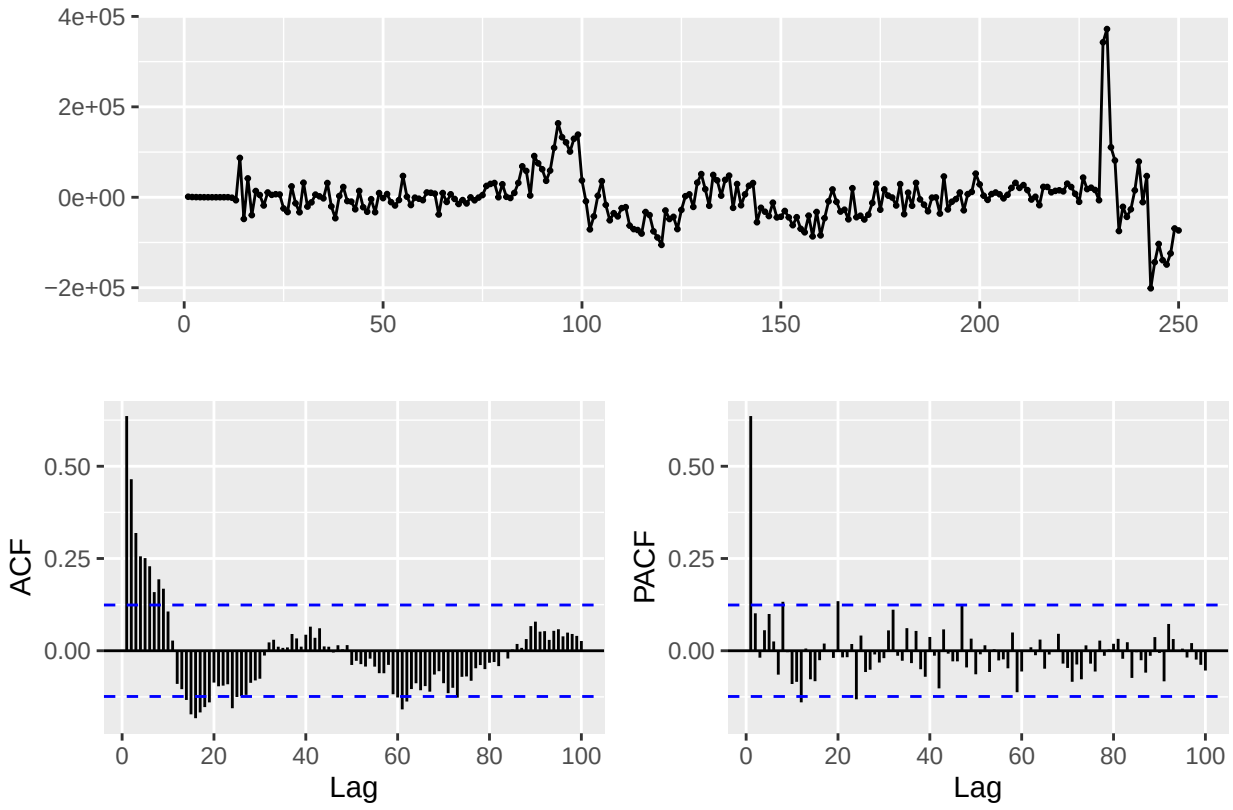
Para posteriormente poder ajustar correctamente nuestra parte estacional del modelo, vamos a diferenciar la parte estacional y estudiar su gráfica ACF y PACF.

```
Bz_s <- diff(Bz, lag = 12, differences = 1)
ggtsdisplay(Bz_s, lag.max = 65)
```



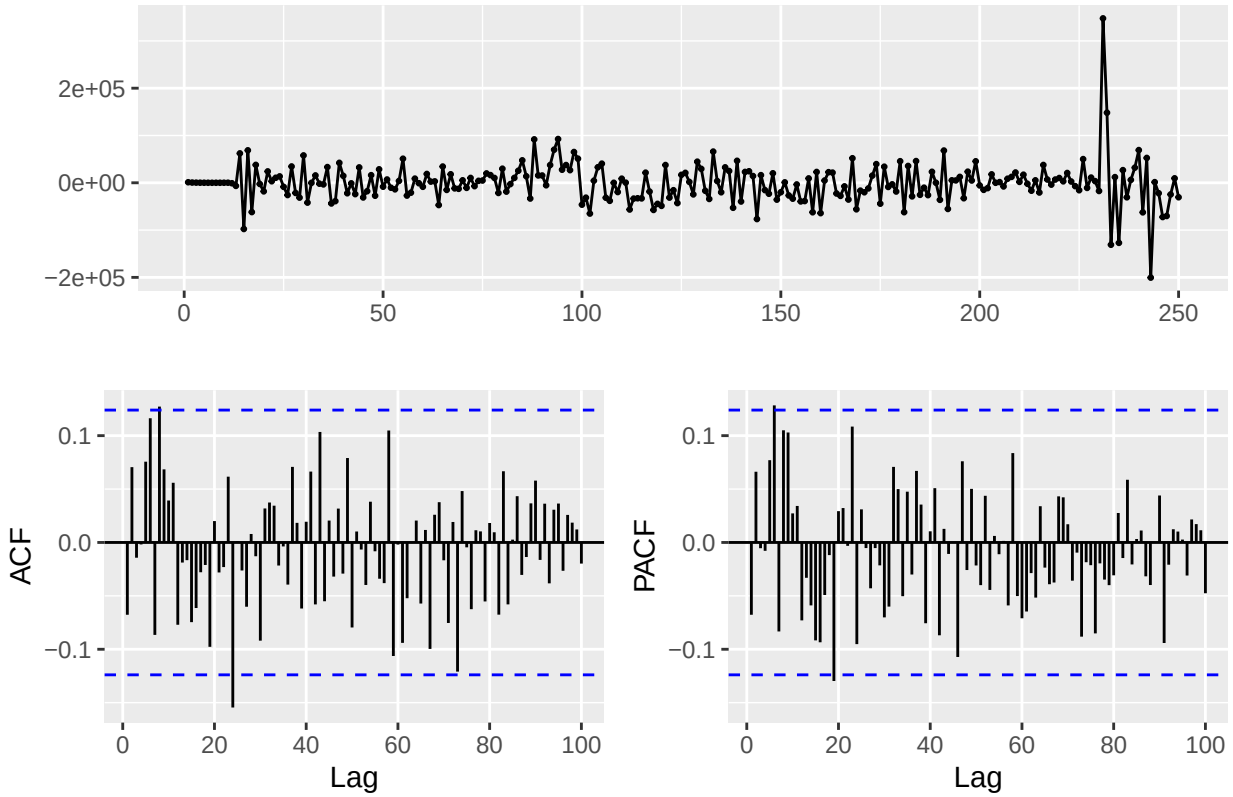
Podemos ver ahora que la parte estacional ha mejorado ya que no se observan picos constantes en periodos de 12 (si es cierto que en algún múltiplo de 12 de forma puntual existe algún pico). Sin embargo, observamos cómo el ACF va decreciendo progresivamente (lo que lo asociamos a un proceso autorregresivo) y en PACF hay un coeficiente significante en el punto 12, por lo que en la parte estacional aplicamos un proceso AR(1).

```
sarima.fit <- Arima(y,
                    order=c(0,1,0),
                    seasonal = list(order=c(1,1,0),period=12),
                    include.constant = FALSE)
ggtsdisplay(residuals(sarima.fit),lag.max = 100)
```



Ahora en los residuos podemos ver que la parte estacional parece ajustada más o menos, sin embargo; la parte regular no esta bien modelada. Para ello miramos los residuos, como va decreciendo poco a poco en ACF y vemos un coeficiente bastante significativo en el PACF, volvemos a aplicar un AR(1). Por tanto volvemos a modelar nuestros datos con la parte regular ajustada.

```
sarima.fit <- Arima(y,
                    order=c(1,1,0),
                    seasonal = list(order=c(1,1,0),period=12),
                    include.constant = FALSE)
ggtsdisplay(residuals(sarima.fit),lag.max = 100)
```



Tras ver como quedan los residuos, lo que haremos será ver alguna información relevante a cerca de nuestro modelo. Alguno de estos datos son datos de control y error como el RMSE o el MAE.

```
summary(sarima.fit)
```

```
## Series: y
## ARIMA(1,1,0)(1,1,0)[12]
##
## Coefficients:
##          ar1     sar1
##       0.6496  -0.5673
## s.e.  0.0497   0.0663
##
## sigma^2 estimated as 1.962e+09:  log likelihood=-2873.46
## AIC=5752.93   AICc=5753.03   BIC=5763.33
##
## Training set error measures:
##                      ME     RMSE      MAE        MPE      MAPE      MASE
## Training set -585.4926 42948.38 27682.11 0.02010853 0.8720577 0.4961341
##                      ACF1
## Training set -0.06769942
```

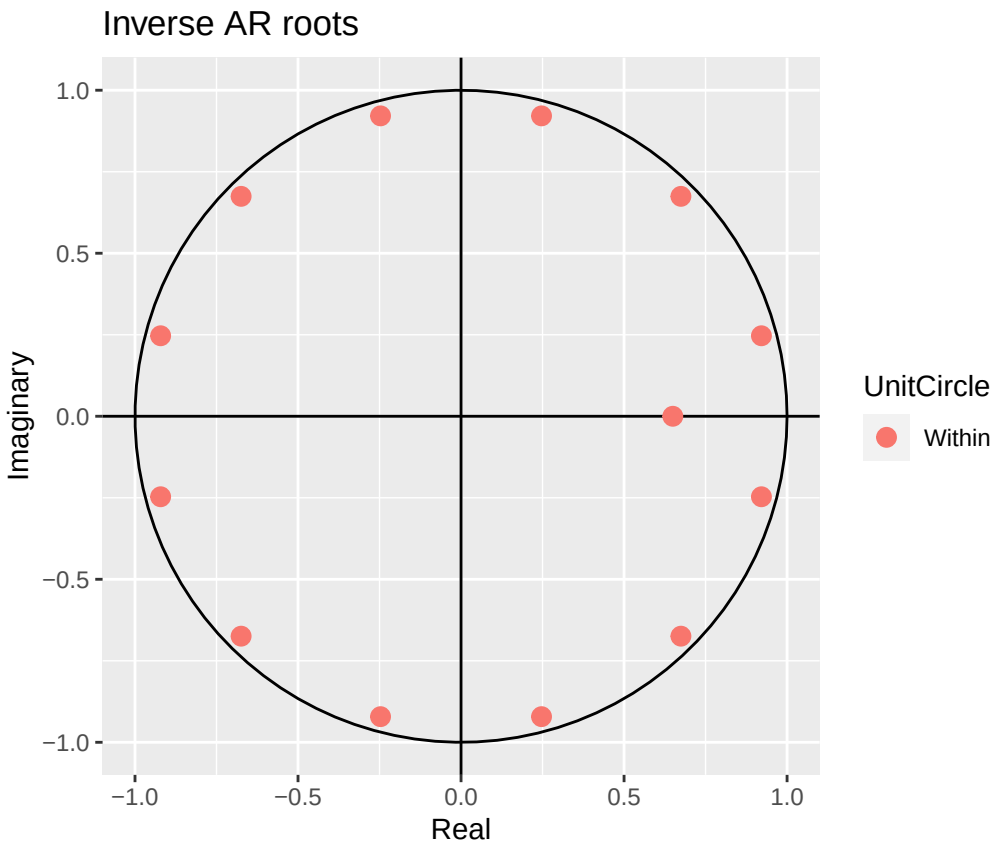
También es importante ver si nuestros coeficientes son realmente significativos, esto lo miraremos con la función coeftest.

```
coeftest(sarima.fit)
```

```
##
## z test of coefficients:
##
##       Estimate Std. Error z value  Pr(>|z|)
## ar1   0.649599   0.049729 13.0629 < 2.2e-16 ***
## sar1 -0.567347   0.066343 -8.5517 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Obtenemos que ambos coeficientes son relevantes, por lo que podemos pasar a otro punto. Lo siguiente será ver donde caen las raices del polinomio característico. Para ver esto usaremos las inversas de las raices, las cuales si caen dentro del circulo unidad complirán lo que se pide para tener un proceso estacionario.

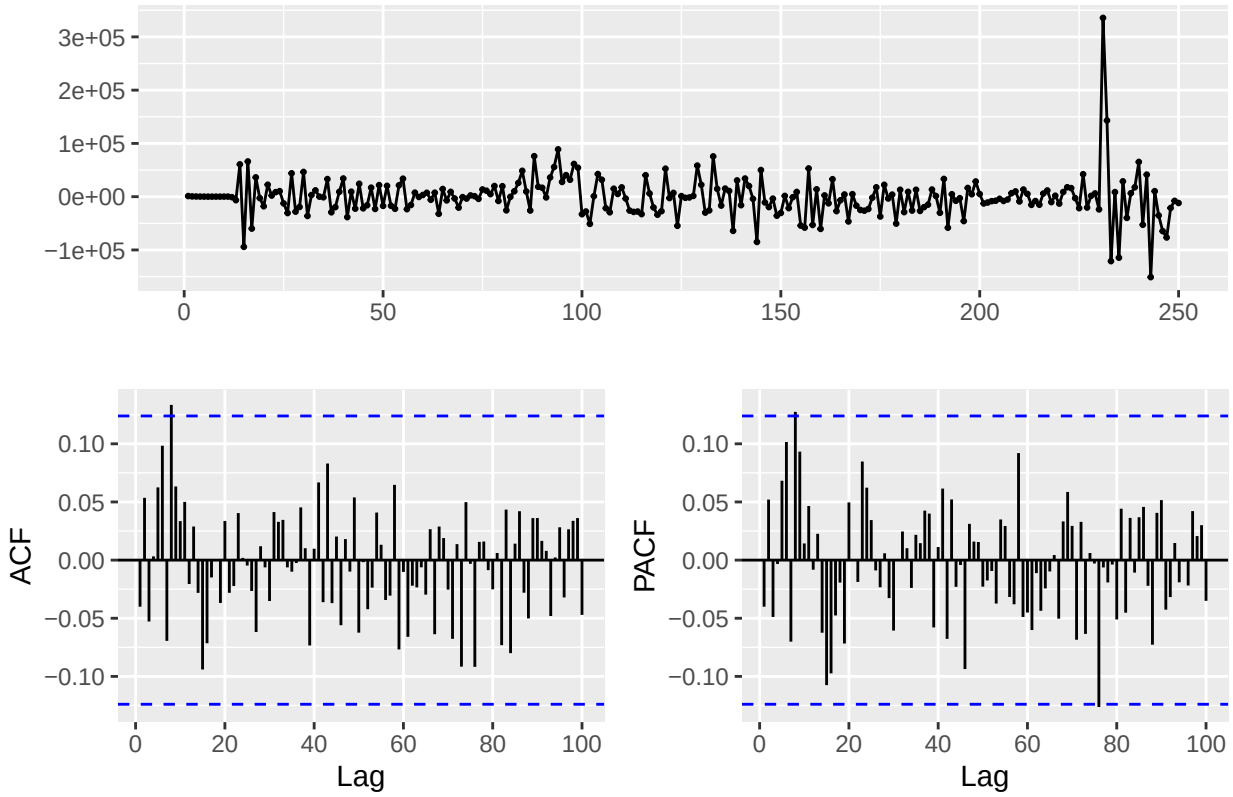
```
autoplot(sarima.fit)
```



No obstante, en la gráfica de los residuos se podía ver que estos no son perfectamente ruido blanco en la parte estacional, por este motivo intentamos ajustarlo con un MA(2).

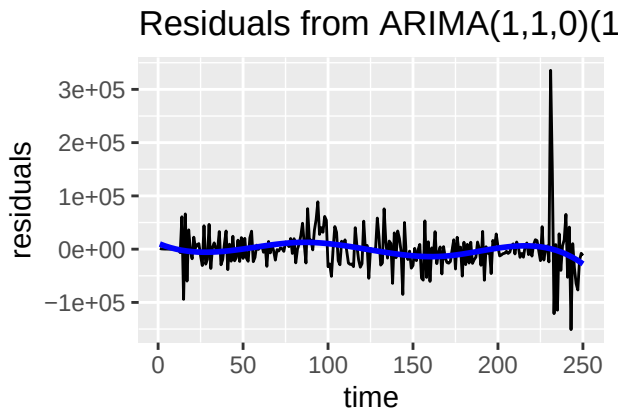
```
sarima.fit <- Arima(y,
                    order=c(1,1,0),
                    seasonal = list(order=c(1,1,2),period=12),
                    include.constant = FALSE)
```

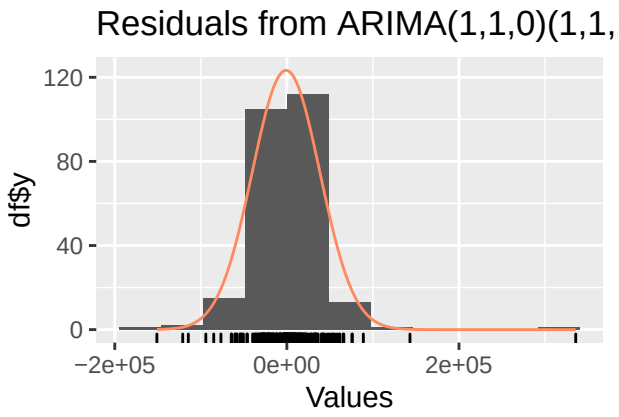
```
ggtsdisplay(residuals(sarima.fit),lag.max = 100)
```

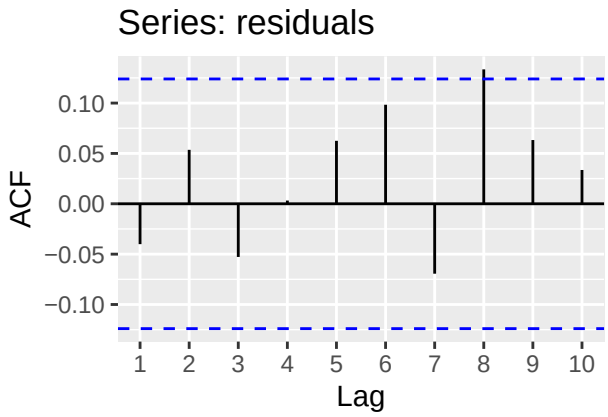


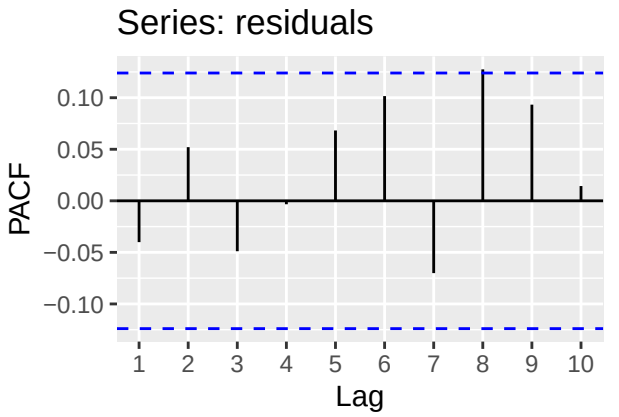
Y volvemos a comprobar los mismos elementos anteriores con el modelo final. En este caso vemos también como se comportan los residuos.

```
CheckResiduals.ICAI(sarima.fit, bins = 10)
```









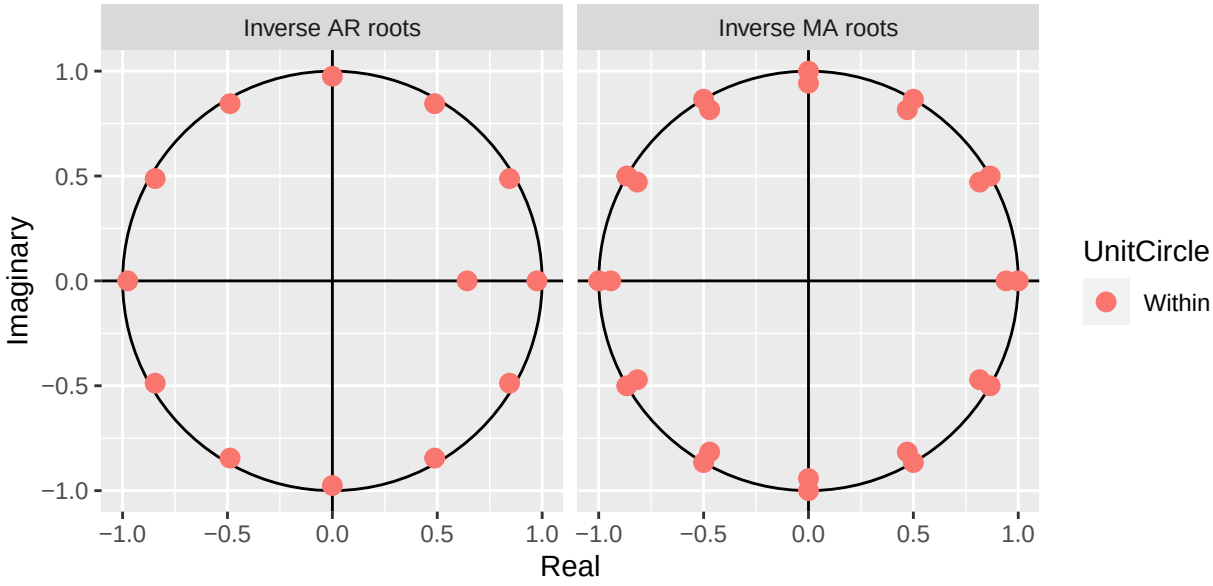

```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,0)(1,1,2)[12]
## Q* = 12.58, df = 6, p-value = 0.05021
##
## Model df: 4.   Total lags used: 10
```

Despues miramos los otros elementos restantes:

```
coeftest(sarima.fit)
```

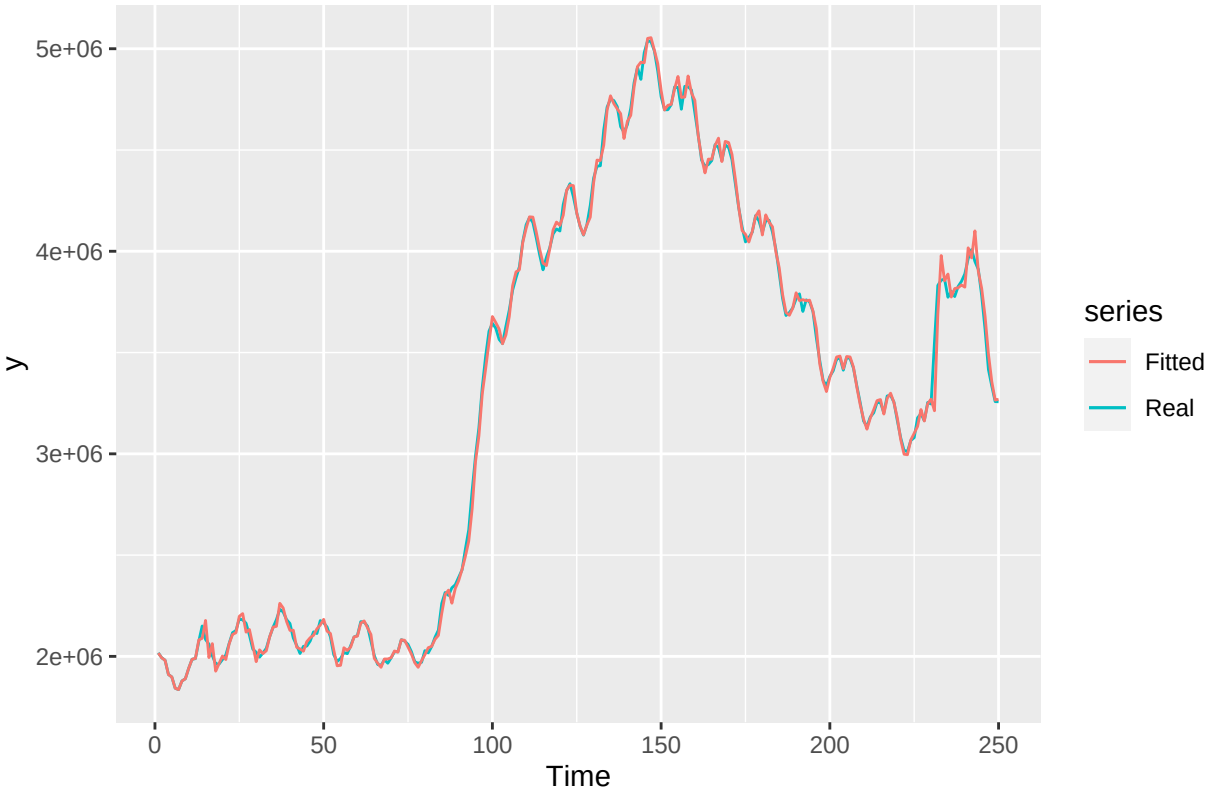
```
##
## z test of coefficients:
##
##       Estimate Std. Error z value  Pr(>|z|)
## ar1   0.642436   0.050165 12.8064 < 2.2e-16 ***
## sar1  0.741882   0.149813  4.9521 7.343e-07 ***
## sma1 -1.489325   0.192980 -7.7175 1.186e-14 ***
## sma2  0.489340   0.166420  2.9404  0.003278 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
autoplot(sarima.fit)
```



Vemos que el ruido generado es ruido blanco, que todas las coeficientes del modelo son relevantes y que generan unos p-valores pequeños. Además, comprobamos que el modelo genera datos predichos que se adaptan bien a los datos reales. Eso lo podemos ver en el siguiente gráfico:

```
autoplot(y, series = "Real")+
  forecast::autolayer(sarima.fit$fitted, series = "Fitted")
```



Como podemos observar, la curva azul y la roja son muy parecidas y se adapta bien a todos los cambios. Lo que es muy satisfactorio. Por ello predecimos obteniendo el resultado enviado para el assigment

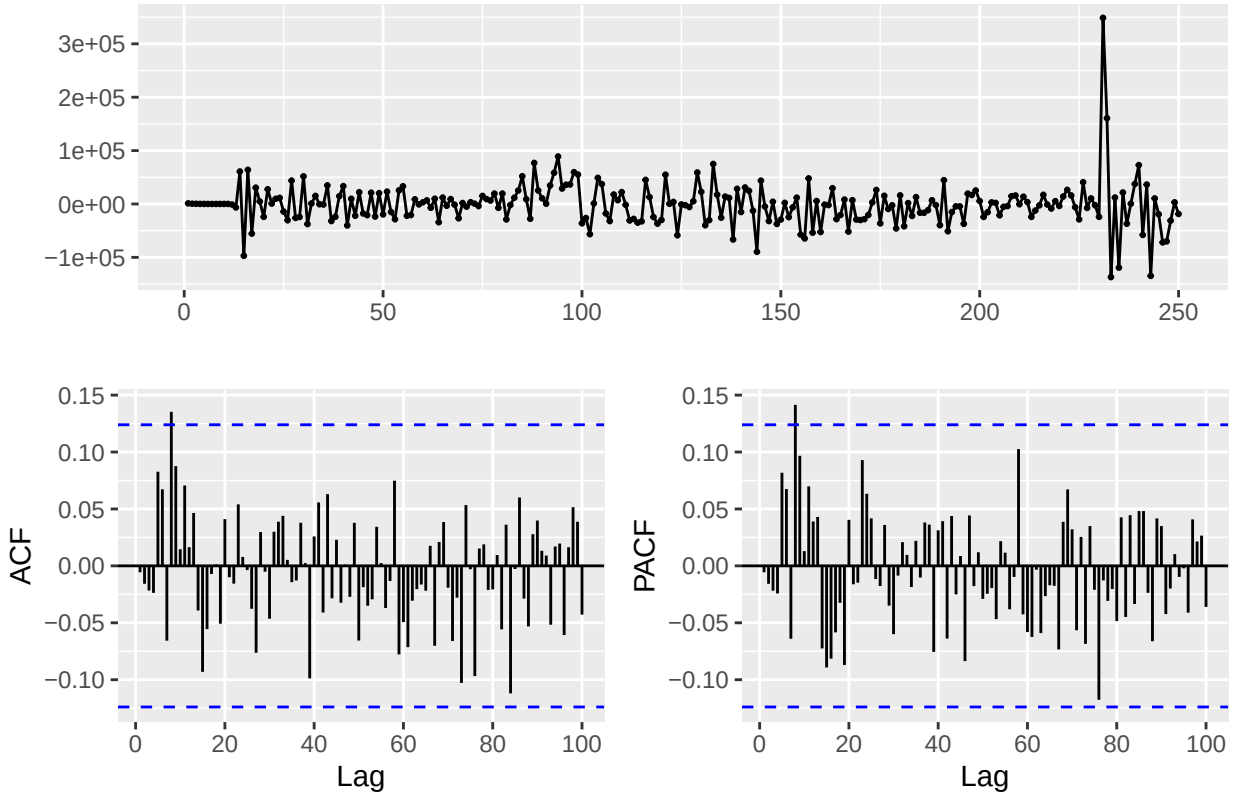
```
y_est <- forecast(sarima.fit, h=1)
y_est
```

```
##     Point Forecast   Lo 80   Hi 80   Lo 95   Hi 95
## 251        3240623 3187368 3293877 3159177 3322068
```

Otros modelos intentados

Probamos también a crear otros modelos. Partimos de la base de haber diferenciado una vez tanto la parte estacionaria como la regular. Vemos que el ACF va disminuyendo de forma senoidal en la parte estacionaria, por lo que añadimos un MA(1). Tras ello, tenemos que ajustar la parte regular, añadiendo RA(2) y un MA(1).

```
sarima2.fit <- Arima(y,
                     order=c(2,1,1),
                     seasonal = list(order=c(0,1,1),period=12),
                     include.constant = FALSE)
ggtsdisplay(residuals(sarima2.fit),lag.max = 100)
```



Tras crear el nuevo modelo comprobamos algunas métricas relevantes como en el caso del modelo entregado.

```
coeftest(sarima2.fit)
```

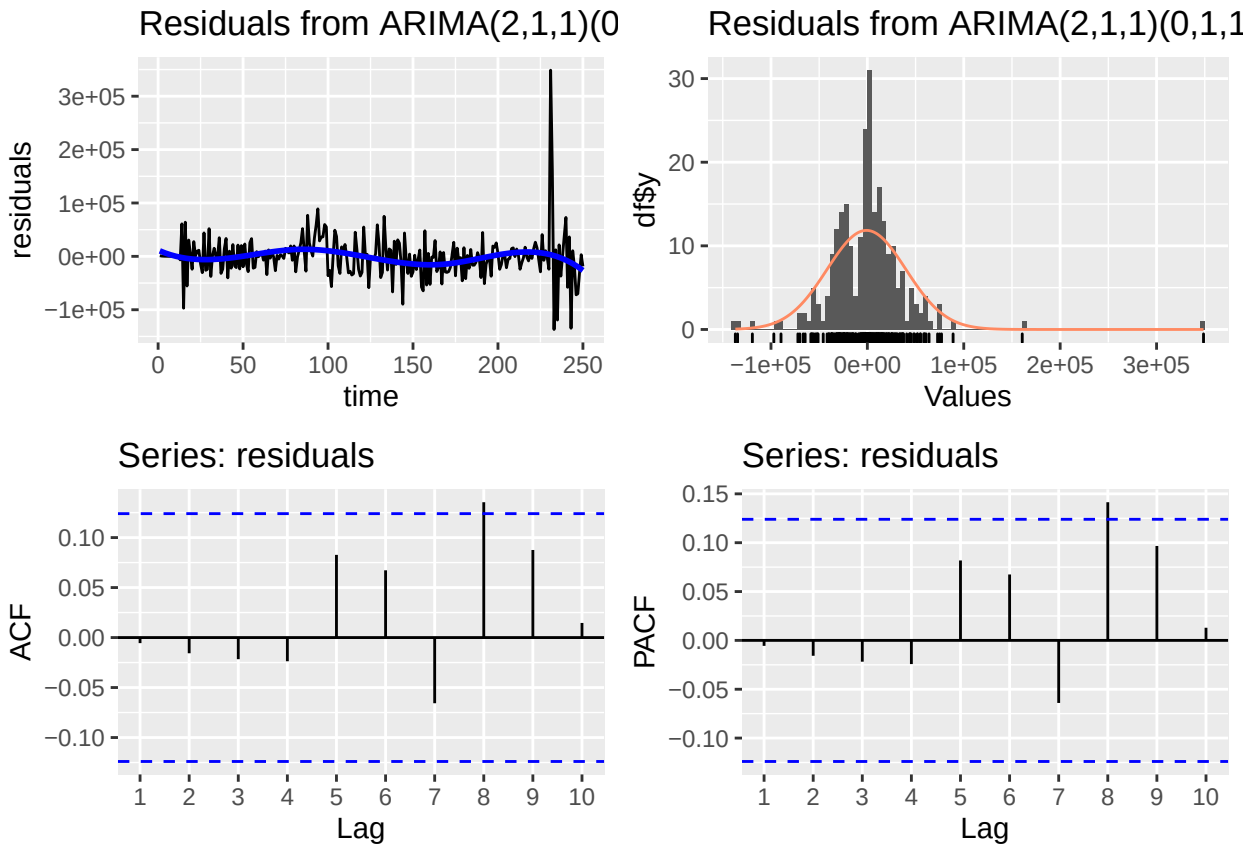
```
##
## z test of coefficients:
##
##       Estimate Std. Error  z value  Pr(>|z|)
## ar1  -0.230650   0.154521  -1.4927    0.1355
## ar2   0.607363   0.098842   6.1448 8.008e-10 ***
## ma1   0.856134   0.162073   5.2824 1.275e-07 ***
## sma1 -0.723409   0.068744 -10.5232 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
summary(sarima2.fit)
```

```
## Series: y
## ARIMA(2,1,1)(0,1,1)[12]
##
## Coefficients:
##           ar1     ar2     ma1     sma1
##       -0.2307  0.6074  0.8561  -0.7234
## s.e.   0.1545  0.0988  0.1621   0.0687
##
## sigma^2 estimated as 1.787e+09:  log likelihood=-2863.4
## AIC=5736.8   AICc=5737.06   BIC=5754.14
##
## Training set error measures:
##                      ME     RMSE      MAE         MPE      MAPE      MASE
## Training set -920.8681 40813.86 25549.44 0.007740555 0.8067537 0.4579112
##                       ACF1
## Training set -0.005591997
```

Tras ello comprobamos el comportamiento de los residuos:

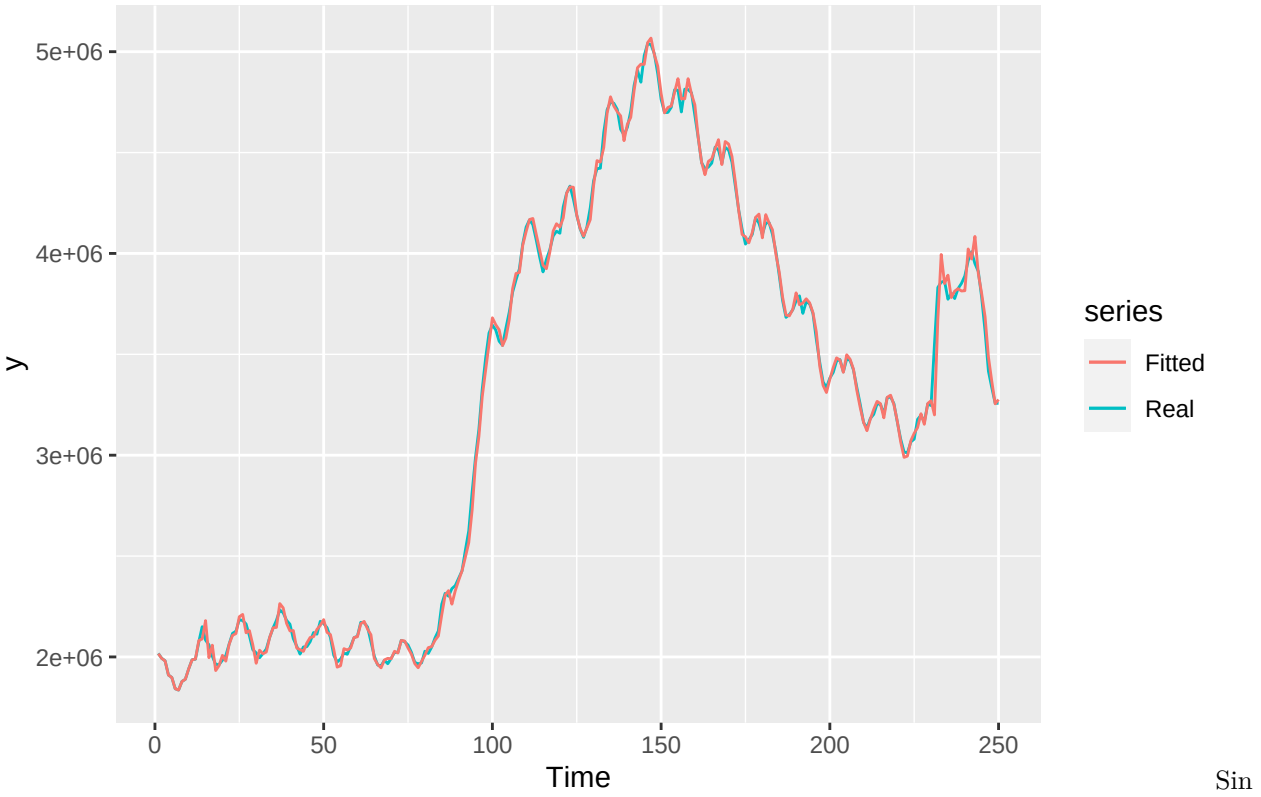
```
CheckResiduals.ICAI(sarima2.fit, bins = 100)
```



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,1)(0,1,1)[12]
## Q* = 11.211, df = 6, p-value = 0.08206
##
## Model df: 4.   Total lags used: 10
```

Por último comprobamos que tal ajusta nuestro modelo con los datos:

```
autoplot(y, series = "Real")+
  forecast::autolayer(sarima2.fit$fitted, series = "Fitted")
```



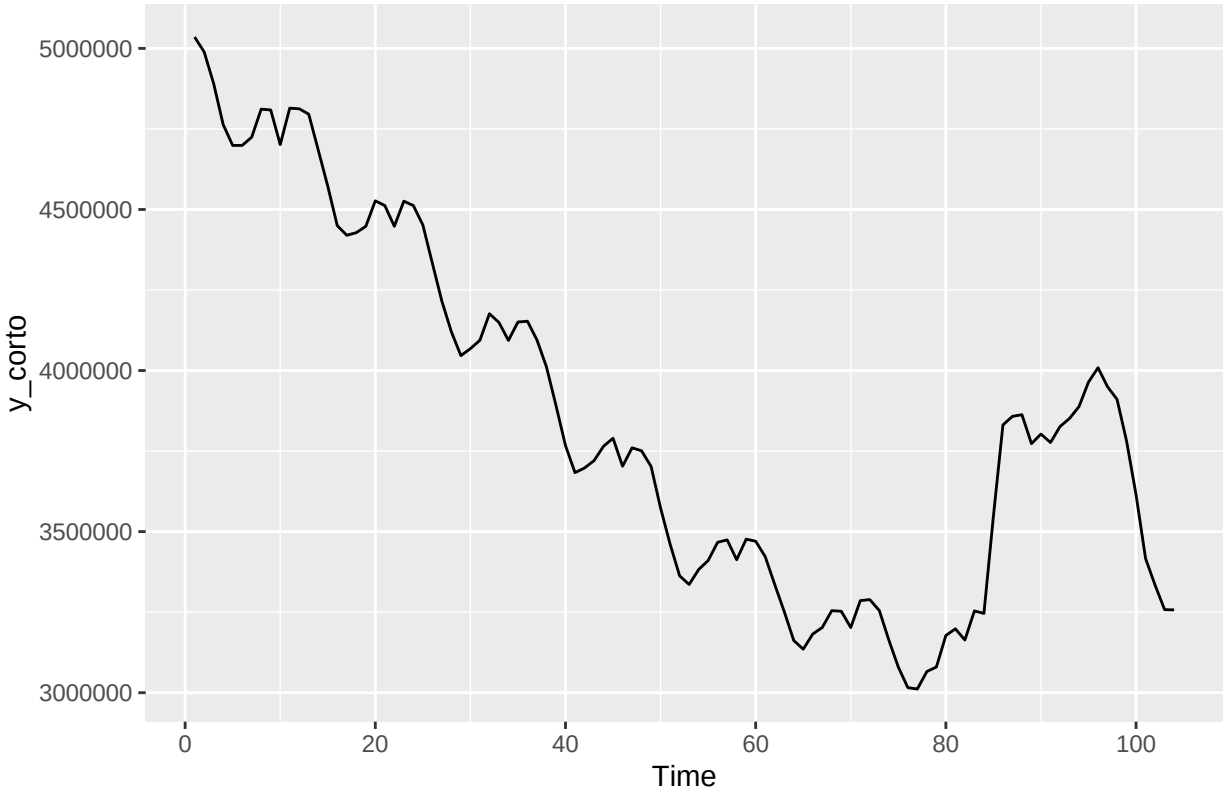
Sin embargo, estos resultados a pesar de que generan ruido blanco y la mayoría de variables aparecen muy significativas, generaba resultados ligeramente peores que los del modelo anterior. Pero esto lo veremos en mayor profundidad en un apartado posterior donde compararemos todos nuestros modelos.

```
y_est <- forecast(sarima2.fit, h=1)
y_est
```

```
##     Point Forecast   Lo 80   Hi 80   Lo 95   Hi 95
## 251        3220246 3166067 3274426 3137386 3303107
```

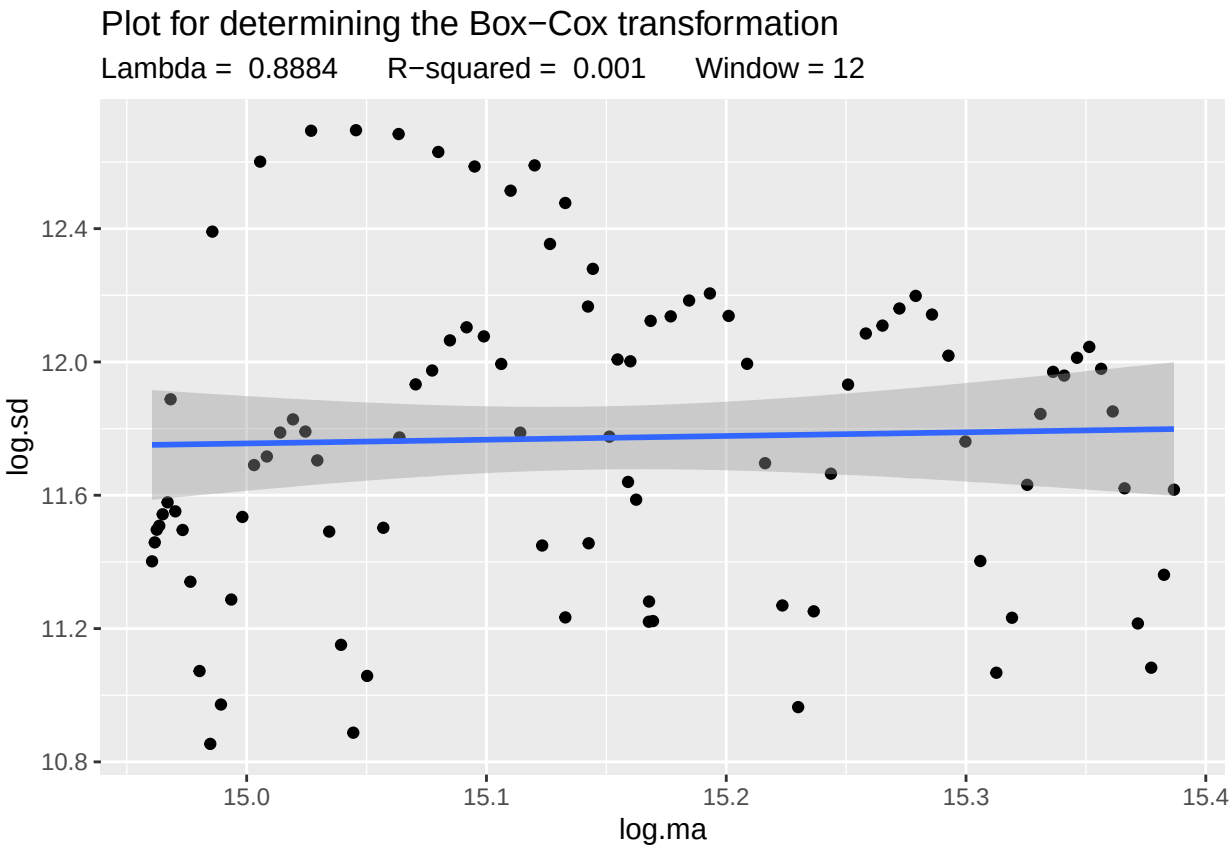
Nuevos modelos

Para buscar una predicción más acertada a la realidad, hemos creado varios modelos nuevos. La línea de trabajo que hemos adoptado ha sido analizar la serie temporal después de la crisis de 2008, es decir; cuando los datos del paro empiezan a decrecer. De esta forma intentamos conseguir un modelo que no esté mal influido por los datos previos donde se producía un aumento del paro.



Una vez tenemos estos datos, empezamos procediendo de igual forma. Analizamos si la serie es estacionaria en varianza.

```
## `geom_smooth()` using formula 'y ~ x'
```



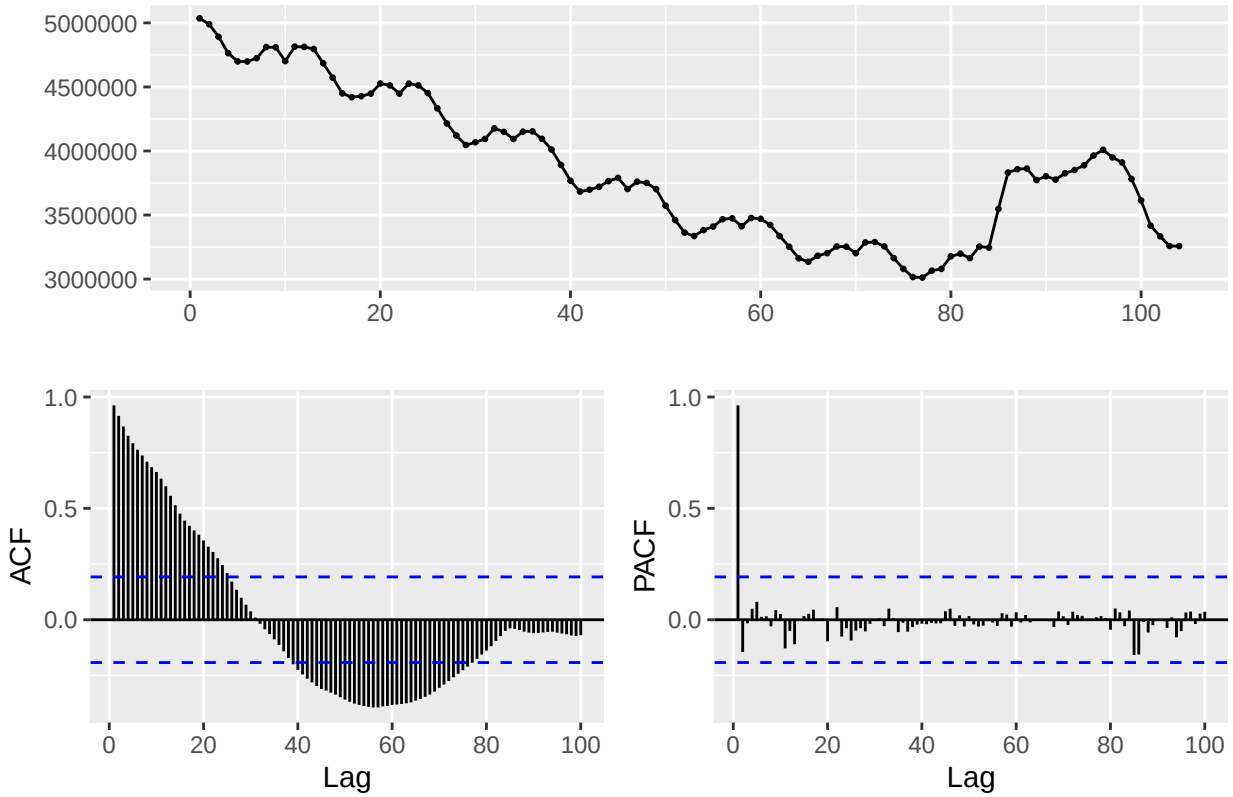

```
##    log.ma
## 0.8884236
```

Que como se puede ver lo es ya que la línea azul es práctiacmente horizontal y R-squared = 0.001. Por lo tanto, no es necesario realizar una estabilización de varianza.

Seasonal Arima

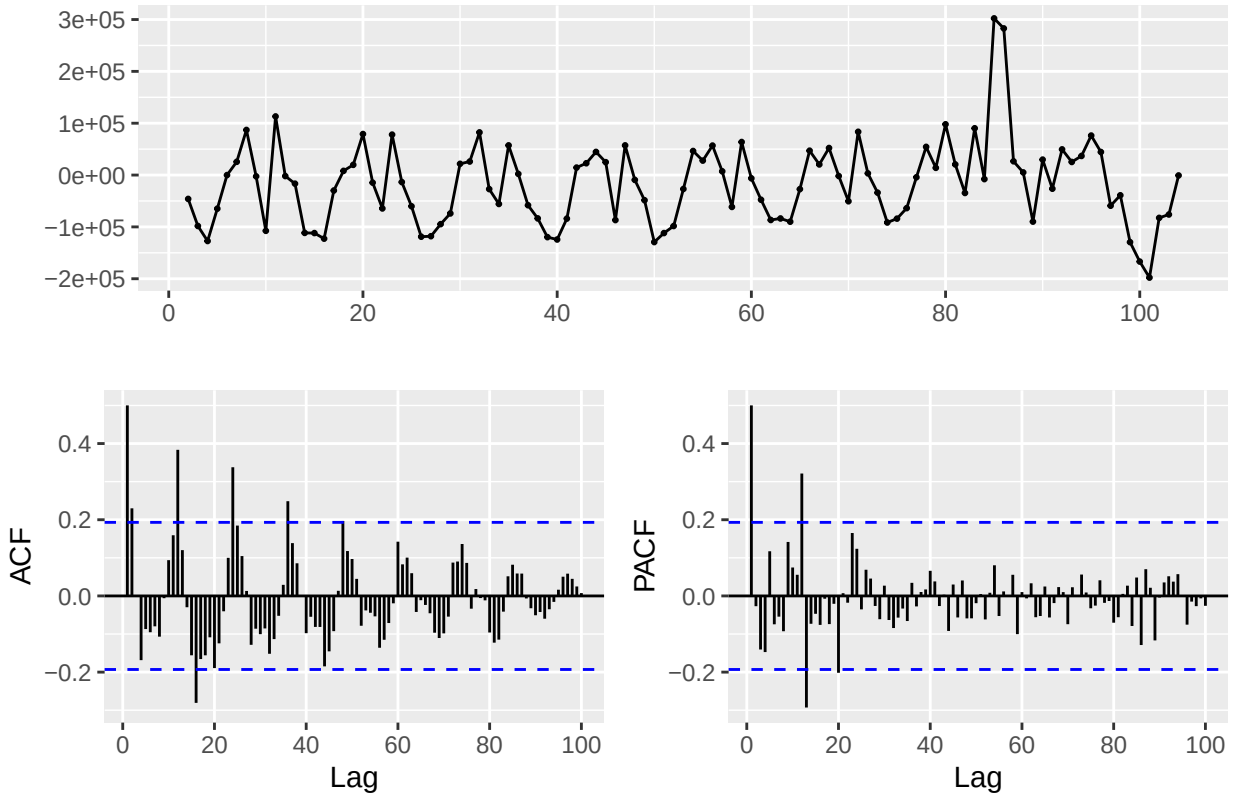
Empezamos por tanto representando la serie temporal junto con su ACF y PACF para inspeccionar la serie regular y estacional

```
ggtsdisplay(y_corto,lag.max = 100)
```



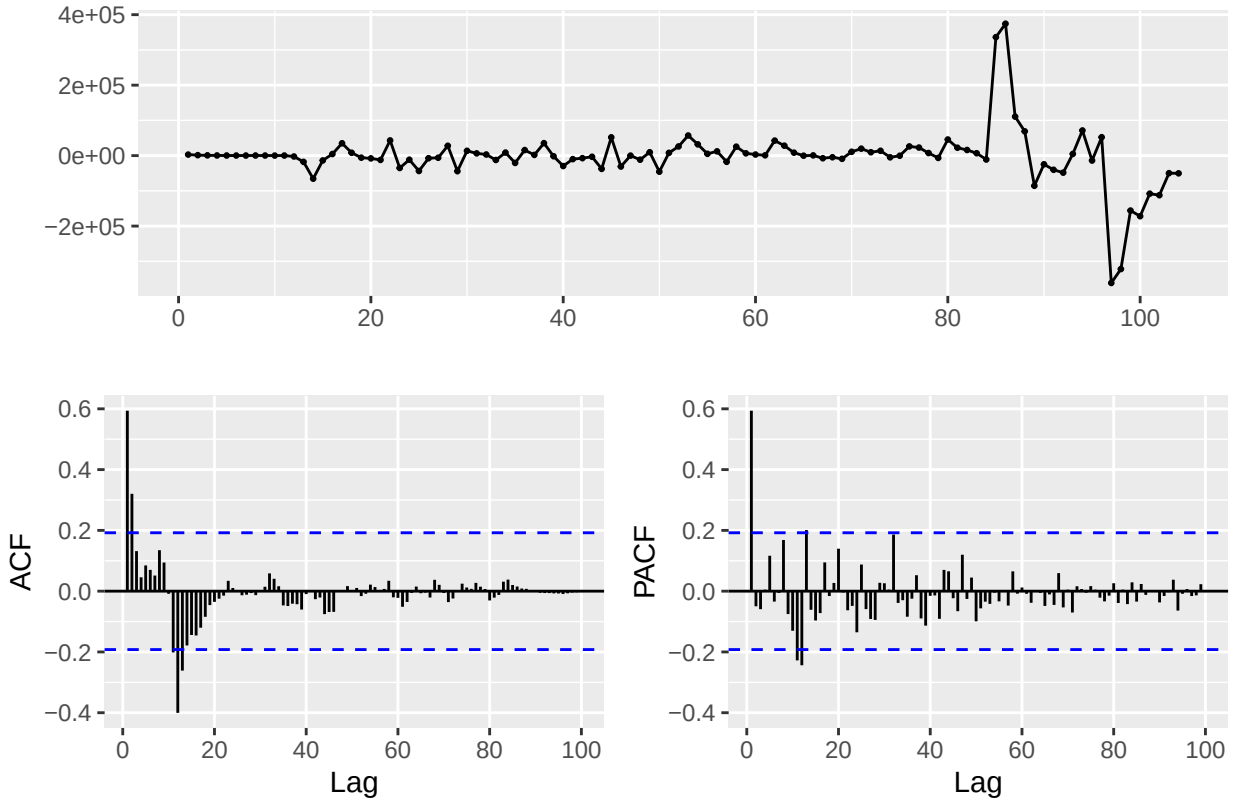
Vemos que la serie necesita una diferenciación en la parte regular ya que el ACF dismunye progresivamente por el tiempo y volvemos a mostrar el ACF y el PACF

```
Bz <- diff(y_corto,differences = 1)
ggtsdisplay(Bz,lag.max = 100)
```



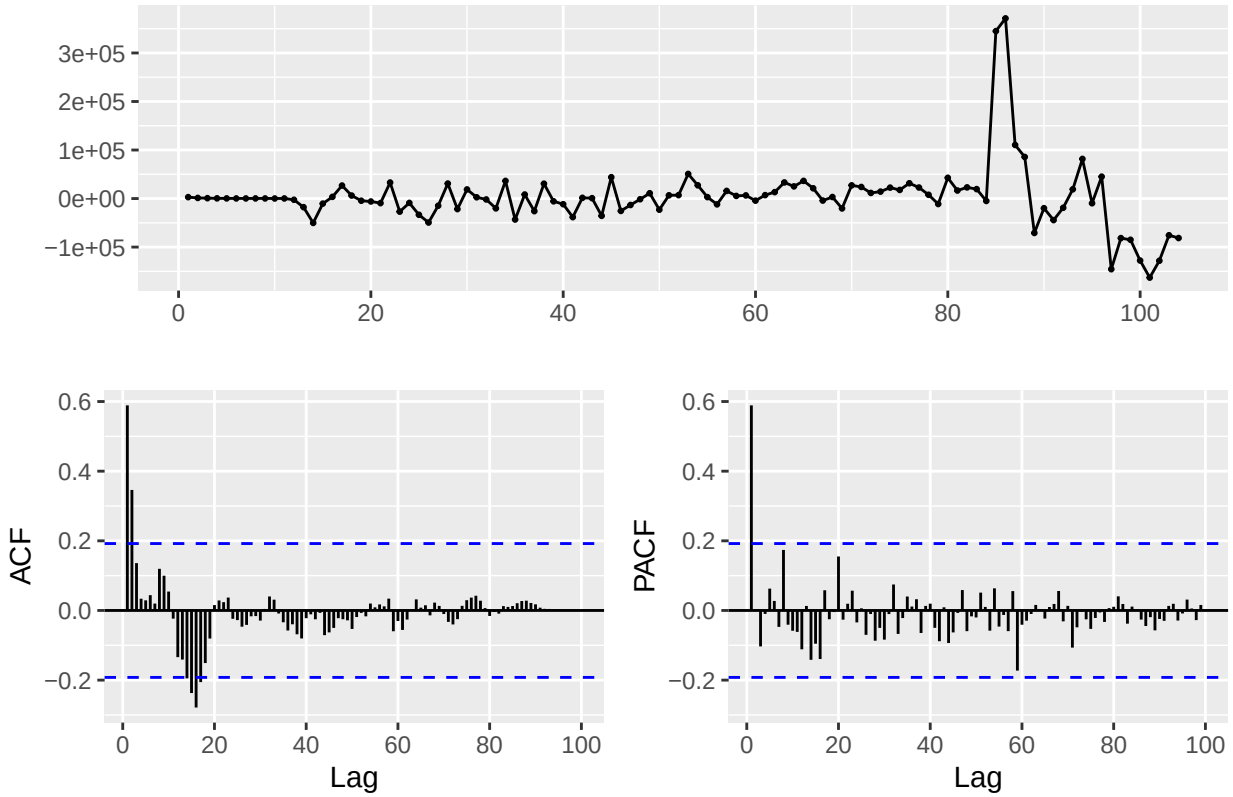
Vemos que cada 12 puntos aparece un pico y por tanto deducimos que tenemos que diferenciar en la parte estacional también.

```
arimaCorto.fit <- Arima(y_corto,
                    order=c(0,1,0),
                    seasonal = list(order=c(0,1,0),period=12),
                    include.constant = FALSE)
ggtsdisplay(residuals(arimaCorto.fit),lag.max = 100)
```



Ahora vemos que en el punto 12, el primero de la parte estacional se produce un residuo significativo. Como estos residous van decreciendo exponencialmente, aplicamos un RA(1) y volvemos a representar los ACF y PACF.

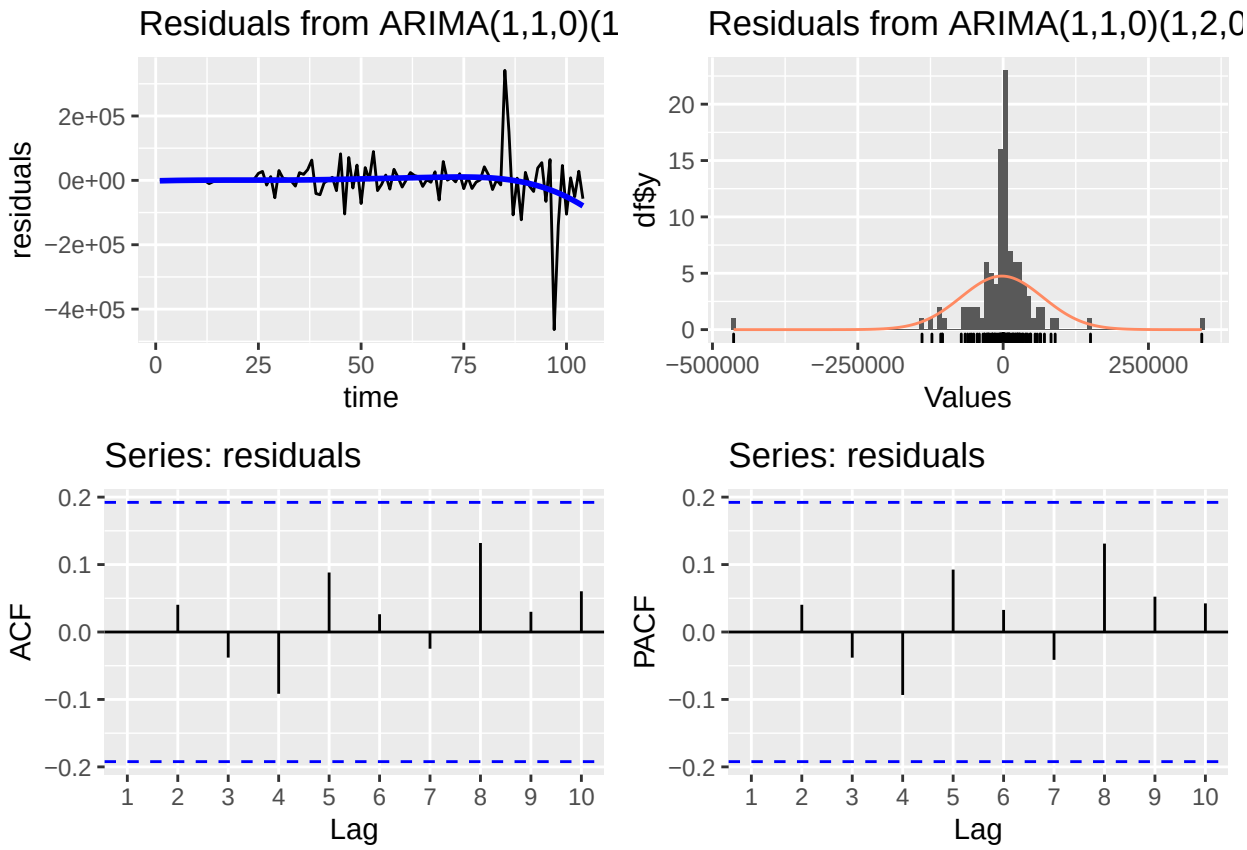
```
arimaCorto.fit <- Arima(y_corto,
                    order=c(0,1,0),
                    seasonal = list(order=c(1,1,0),period=12),
                    include.constant = FALSE)

ggtsdisplay(residuals(arimaCorto.fit),lag.max = 100)
```



Ahora podemos ver que la parte estacional parece ajustada más o menos, sin embargo; la parte regular quiere una corrección. Como va decreciendo poco a poco en ACF y vemos un gran valor en el PACF, aplicamos un AR(1). Comprobamos los residuos.

```
arimaCorto.fit <- Arima(y_corto,
                    order=c(1,1,0),
                    seasonal = list(order=c(1,2,0),period=12),
                    include.constant = FALSE,
                    method="ML")
```

```
CheckResiduals.ICAI(arimaCorto.fit, bins = 100)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(1,1,0)(1,2,0)[12]
## Q* = 4.8036, df = 8, p-value = 0.7783
##
## Model df: 2.   Total lags used: 10
```

```
summary(arimaCorto.fit) # summary of training errors and estimated coefficients
```

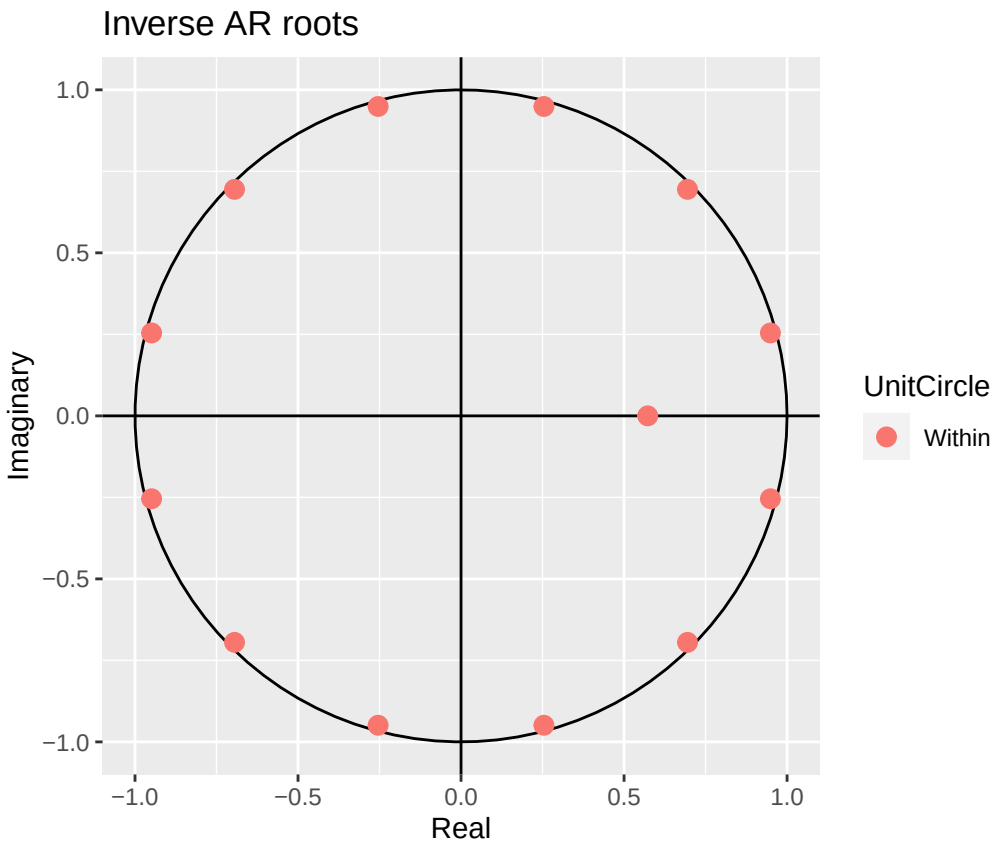
```
## Series: y_corto
## ARIMA(1,1,0)(1,2,0)[12]
##
## Coefficients:
##          ar1     sar1
##       0.5726  -0.8084
## s.e.  0.0921   0.0734
##
## sigma^2 estimated as 6.63e+09:  log likelihood=-1014.67
## AIC=2035.34   AICc=2035.66   BIC=2042.45
##
## Training set error measures:
```

```
##                      ME     RMSE      MAE         MPE     MAPE      MASE
## Training set -2184.298 70064.61 34056.08 -0.04978799 0.924992 0.5450673
##                      ACF1
## Training set 0.001045982
```

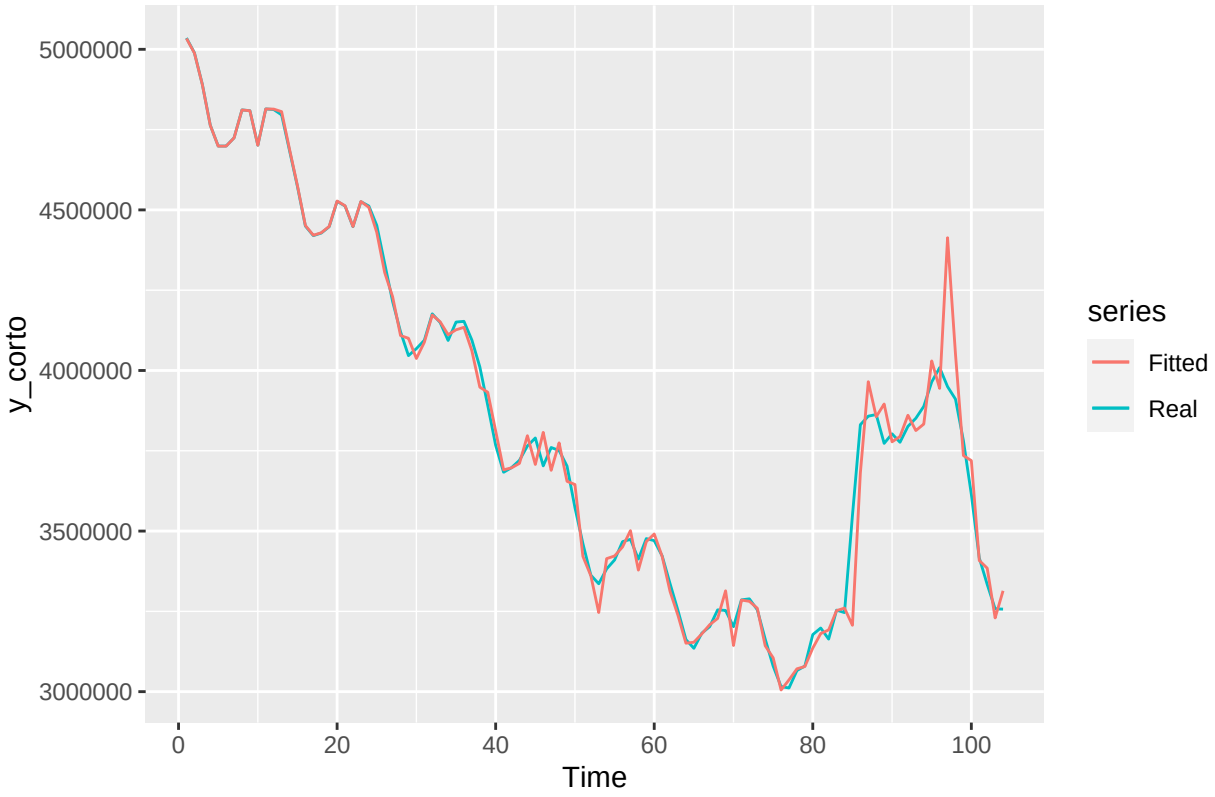
```
coeftest(arimaCorto.fit) # statistical significance of estimated coefficients
```

```
##
## z test of coefficients:
##
##       Estimate Std. Error  z value  Pr(>|z|)
## ar1   0.572592   0.092115   6.2161 5.097e-10 ***
## sar1 -0.808434   0.073380 -11.0170 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
autoplot(arimaCorto.fit) # root plot
```




```
autoplot(y_corto, series = "Real")+
  forecast::autolayer(arimaCorto.fit$fitted, series = "Fitted")
```




```
y_est <- forecast(arimaCorto.fit, h=1)
y_est[("mean")]
```

```
## $mean
## Time Series:
## Start = 105
## End = 105
## Frequency = 1
## [1] 3256655
```