PRÁCTICA SERIES TEMPORALES

Javier Gil Domínguez, David Lázaro Martín, Nicolás Vega Muñoz

2022-12-13

```
#install.packages(c("fUnitRoots","forecast","lmtest", "readxl"))
library(lmtest)
```

```
## Warning: package 'lmtest' was built under R version 4.1.3
```

```
## Loading required package: zoo
```

```
## Warning: package 'zoo' was built under R version 4.1.3
```

```
##
## Attaching package: 'zoo'
```

```
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
library(tseries)
```

```
## Warning: package 'tseries' was built under R version 4.1.3
```

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

```
library(forecast)
```

```
## Warning: package 'forecast' was built under R version 4.1.3
```

```
library(fUnitRoots)
```

```
## Warning: package 'fUnitRoots' was built under R version 4.1.3
```

```
library(readxl)
```

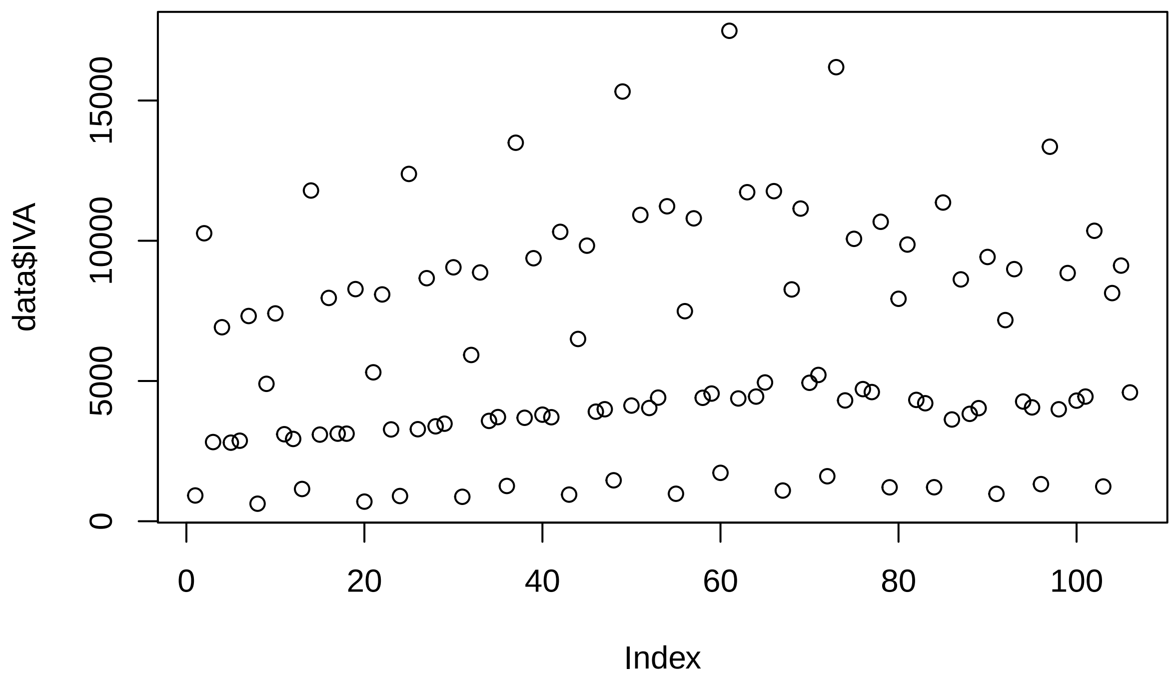
```
## Warning: package 'readxl' was built under R version 4.1.2
```

```r
data <- read_excel(file.choose())
#data = read_excel('IVA.xls')
```

a) Analizar la estacionariedad de la serie y transformarla en caso de fuese necesario para alcanzar la estacionariedad y tratar la estacionalidad.

Mostramos los datos de los que partimos

```r
plot(data$IVA)
```



Creamos un objeto data serie

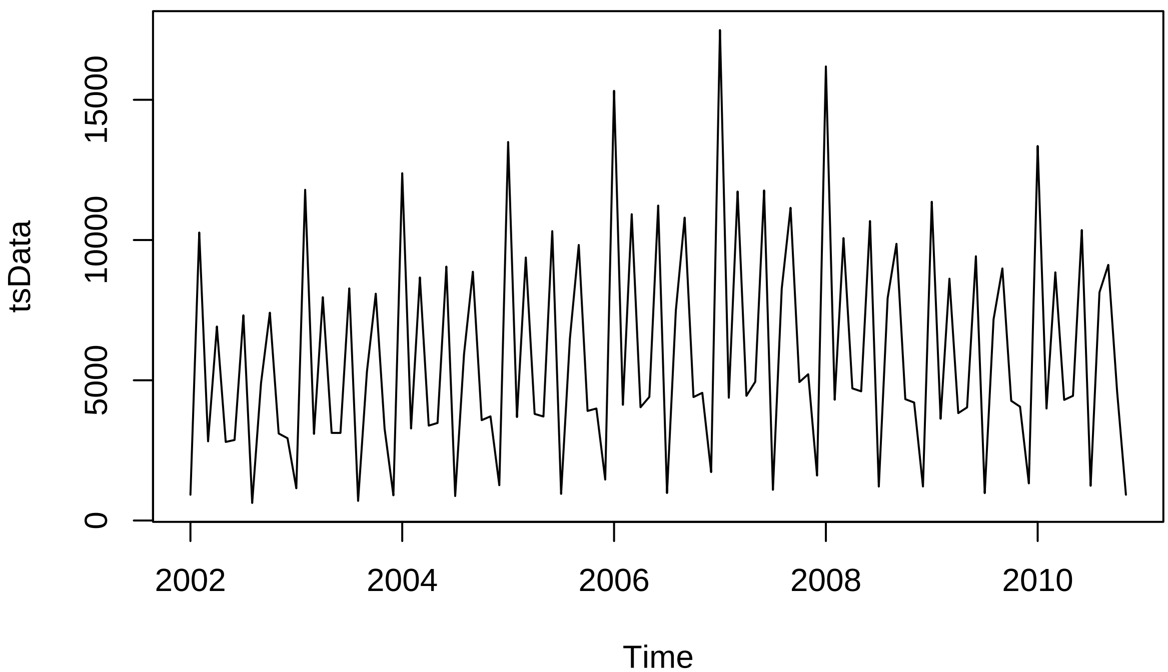
```r
tsData = ts(data, start = c(2002,1), frequency = 12, end=c(2010,11))
tsData
```

```
##          Jan     Feb     Mar    Apr    May     Jun    Jul    Aug     Sep
## 2002   917.9 10267.6  2822.8 6917.0 2801.8  2870.7 7316.8  625.6  4899.5
## 2003  1148.7 11791.9  3088.6 7961.5 3122.7  3121.8 8276.0  699.1  5308.5
## 2004 12383.2  3281.2  8666.6 3383.7 3477.3  9052.8  871.9 5927.7  8870.1
## 2005 13495.6  3689.2  9377.5 3799.8 3708.2 10317.9  949.3 6498.5  9825.0
## 2006 15321.3  4124.4 10921.7 4037.6 4407.8 11229.2  979.9 7488.7 10799.0
## 2007 17486.5  4376.8 11731.1 4445.1 4947.3 11768.4 1094.3 8265.2 11147.7
## 2008 16189.8  4306.8 10069.1 4710.6 4605.2 10677.0 1209.6 7931.3  9864.8
## 2009 11363.9  3628.2  8622.6 3828.5 4032.3  9422.2  977.5 7171.5  8988.2
## 2010 13352.9  3991.1  8848.7 4299.8 4444.2 10355.9 1238.9 8134.3  9115.3
```

```
##          Oct     Nov     Dec
## 2002  7408.9  3102.3  2934.1
## 2003  8086.2  3273.5   897.9
## 2004  3577.6  3713.9  1257.6
## 2005  3906.6  3993.2  1459.1
## 2006  4400.9  4552.4  1724.9
## 2007  4936.2  5216.1  1603.4
## 2008  4325.5  4207.4  1211.5
## 2009  4268.2  4058.3  1323.1
## 2010  4591.6   917.9
```

Dibujamos la serie temporal.

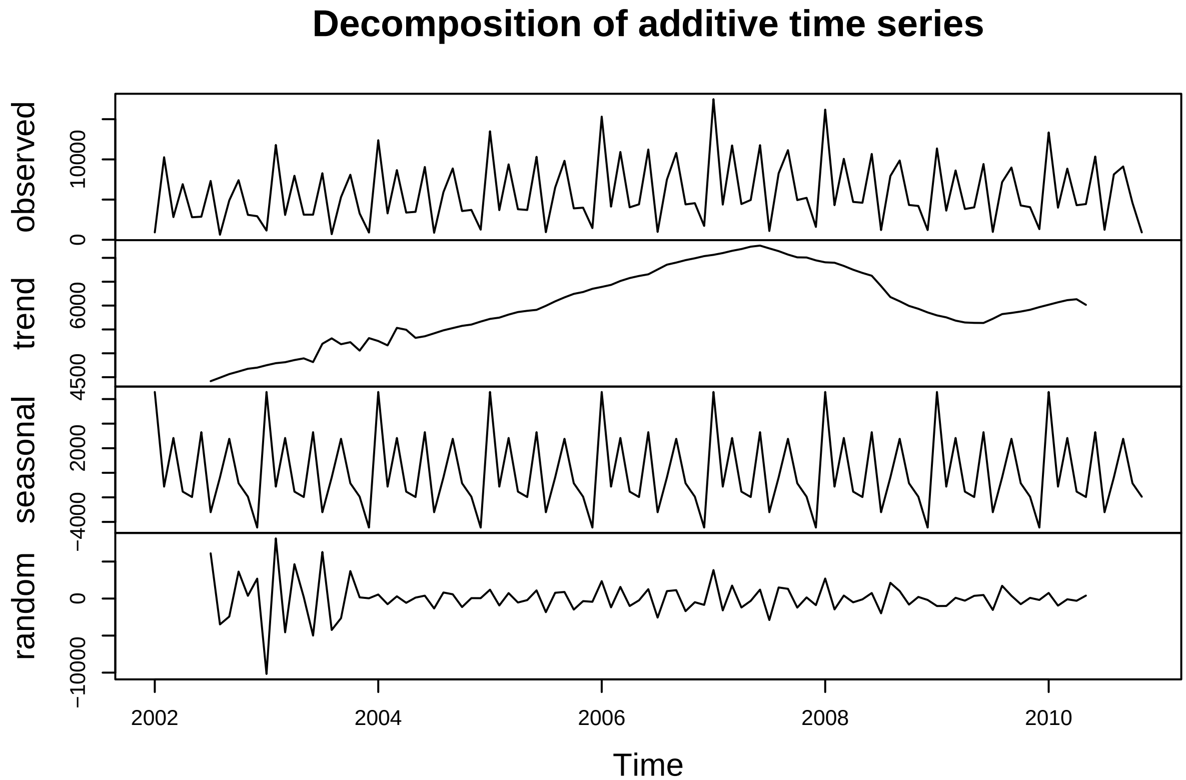
```
plot(tsData)
```



Viendo el gráfico, parece que se trata de una serie estacional, sin embargo no parece ser estacionaria en media (primero aumenta y luego disminuye), ni en varianza (aumenta conforme avanza el tiempo).

Estudiamos gráficamente la serie más a fondo. Primero calculamos sus componentes.

```
componentes.ts = decompose(tsData)
plot(componentes.ts)
```



Podemos ver que la serie no es estacionaria en media ya que aumenta hasta el año 2007 aproximadamente y luego disminuye. Por otro lado, se ve que es estacional ya que hay un patrón que se repite anualmente. Y por último, la serie tampoco es estacionaria en varianza ya que no es constante.

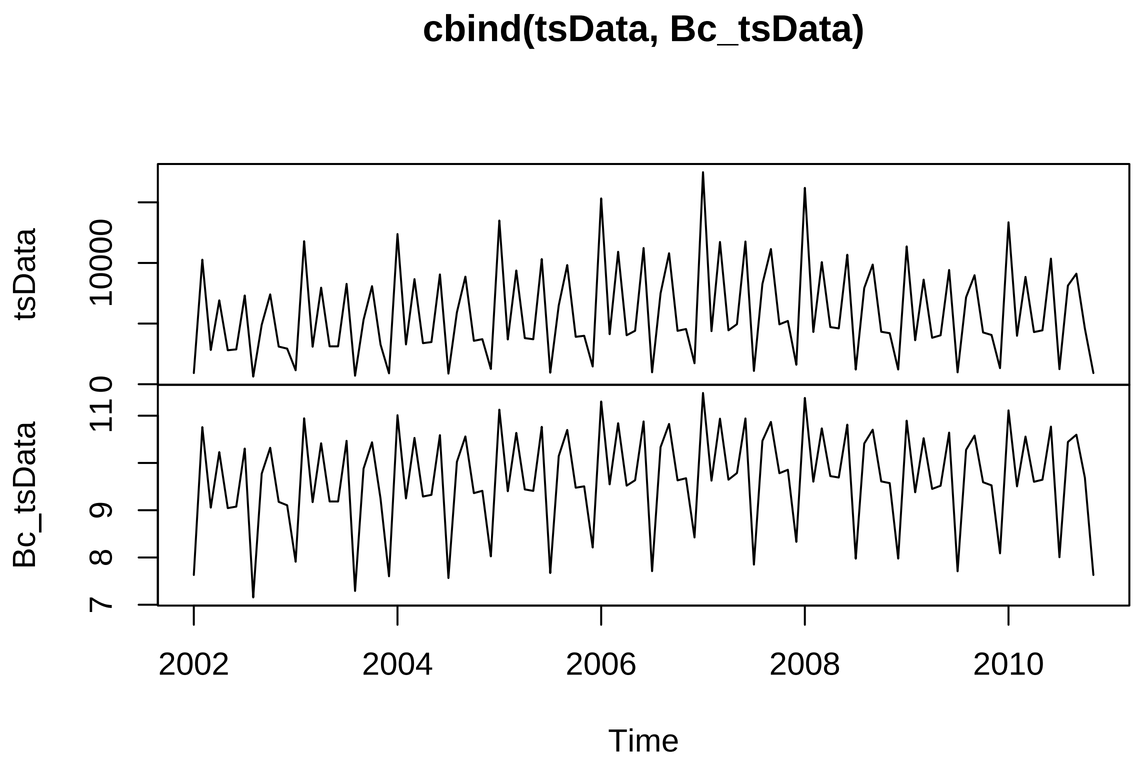
Empezamos estabilizando la varianza.

```
lambda <- BoxCox.lambda(tsData)
lambda
```

```
## [1] 0.03220788
```

Calcular esta lambda nos ayuda para poder realizar la transformación de Box-Cox que es la forma más general de estabilizar la varianza (aunque vemos que el valor de lambda es muy cercano a 0 lo que indica que esta transformación es prácticamente equivalente a aplicarle el logaritmo neperiano).

```
Bc_tsData <- BoxCox(tsData, lambda)
plot(cbind(tsData,Bc_tsData))
```



Vemos que la nueva serie es estacionaria en varianza ya que al compararla con la anterior, observamos que la varianza se ha estabilizado.

A continuación, vamos a tratar la estacionalidad. Ayudándonos de nsdiffs que nos indica las diferencias estacionales necesarias.

```
nsdiffs(Bc_tsData)
```

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

Al ser 1, aplicamos el operador de diferencia estacional de periodo 12 y orden 1 ya que tenemos datos mensuales.

```
tsstationary <- diff(Bc_tsData, lag=frequency(Bc_tsData))
```

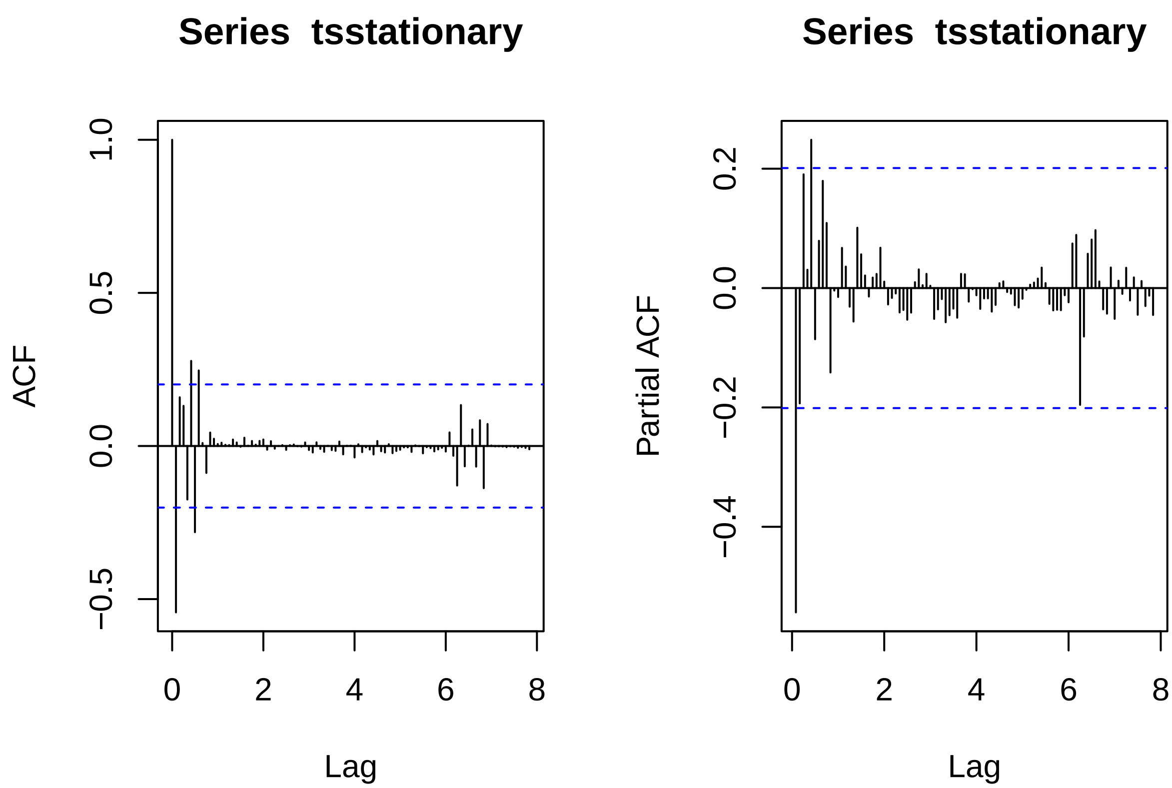
Volvemos a calcular nsdiffs para ver si es necesario hacer alguna diferenciación más.

```
nsdiffs(tsstationary)
```

```
## [1] 0
```

Vemos que ya es 0, por lo que no hace falta hacer ninguna transformación más. Además, lo podemos comprobar con el ACF y el PACF.

```
par(mfrow = c(1,2))
acf(tsstationary,lag.max=107)
pacf(tsstationary,lag.max=107)
```



Vemos que no existen valores significativos en múltiplos de 12 (valor de la frecuencia) así que podemos considerar a la serie como no estacional.

Por último, vamos a tratar la estacionariedad en media. Para ello vamos a usar ndiffs para saber el número de diferencias regulares necesarias para transfarmar la serie en una estacionaria.

```
ndiffs(tsstationary)
```

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

Vemos que necesita una diferenciación regular, así que seguimos el mismo proceso que antes.

```
Dtsstationary<-diff(tsstationary)
```

Volvemos a calcular ndiffs.

```
ndiffs(Dtsstationary)
```

```
## [1] 0
```

Esto ya nos indica que no son necesarias más transformaciones pero aún así, lo vamos a comprobar con los tests KPSS y ADF

```
kpss.test(Dtsstationary, null = "Trend")
```

```
## Warning in kpss.test(Dtsstationary, null = "Trend"): p-value greater than
## printed p-value
```

```
##
##  KPSS Test for Trend Stationarity
##
## data:  Dtsstationary
## KPSS Trend = 0.031815, Truncation lag parameter = 3, p-value = 0.1
```

```
adf.test(Dtsstationary)
```

```
## Warning in adf.test(Dtsstationary): p-value smaller than printed p-value
```

```
##
##  Augmented Dickey-Fuller Test
##
## data:  Dtsstationary
## Dickey-Fuller = -7.0535, Lag order = 4, p-value = 0.01
## alternative hypothesis: stationary
```

KPSS no rechaza la hipotesis de que la serie sea estacionaria al ser su p-valor $= 0.1 > 0.05$. Así que realizamos un segundo test pues sabemos que este es propenso a cometer errores de Tipo II.

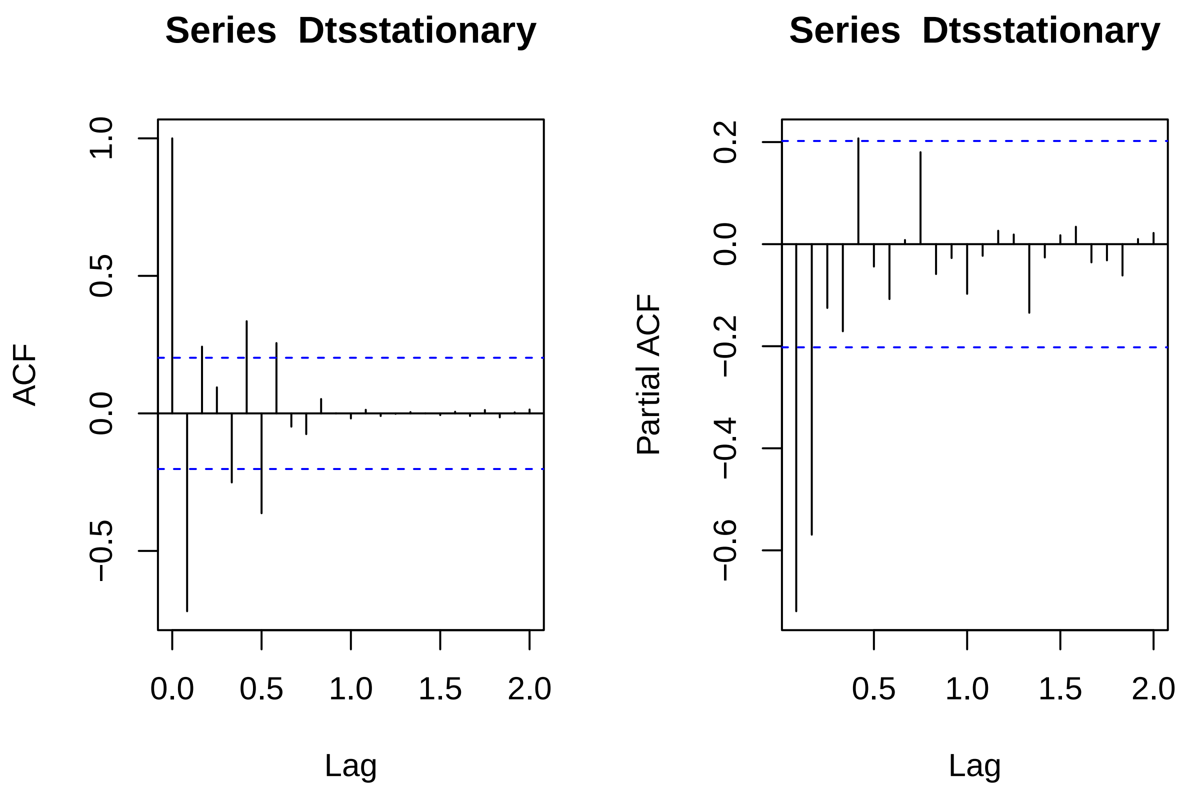
ADF rechaza la hipótesis de que la serie no sea estacionaria por lo tanto, es estacionaria.

Con la serie estacionaria en varianza y media con la estacionalidad tratada, vamos a calcular los modelos ARIMA que mejor se ajusten.

b) Analizar a partir del PCF y PACF los modelos ARIMA generalizados que mejor se ajustan y analizar sus residuos.

Observamos el ACF y PACF de la serie estacionaria para determinar los parámetros del modelo ARIMA generalizado que se aplicará a la serie original (observamos sólo los primeros 24 para que sea más f´cil y claro el análisis).

```
par(mfrow = c(1,2))
acf(Dtsstationary,lag.max=24)
pacf(Dtsstationary,lag.max=24)
```



Tomando como referencia la ACF vemos que hay tres términos significativos consecutivos por lo que la parte regular puede identificarse con los parámetros (0,1,2) (el uno viene de que la serie había sido diferenciada una vez en su parte regular). Si nos fijamos en los términos múltiplos de 12 vemos que no hay ningún término significativo consecutivo por lo que la parte estacional sería (0,1,0) (la serie había sido diferenciada una vez en su parte estacional). El candidato sería ARIMA(0,1,2)(0,1,0)12

Tomando como referencia la PACF vemos que tiene dos términos significativos consecutivos por lo que la parte regular puede identificarse con los parámetros (2,1,0). Si nos fijamos en los términos múltiplos de 12 vemos que no hay ningún término significativo consecutivo por lo que la parte estacional sería (0,1,0) (la serie había sido diferenciada una vez en su parte estacional). El candidato sería ARIMA(2,1,0)(1,1,0)12.

Finalmente, hay que combinar las partes regulares identificadas con todas las estacionales y viceversa, lo que nos lleva a los modelos modelo1: ARIMA(0,1,2)(0,1,0)12 modelo2: ARIMA(2,1,0)(0,1,0)12.

c) Identificar el modelo matemático que mejor se ajusta a la serie, mostrar gráficamente el ajuste del modelo con los datos y analizar sus residuos.

Modelo1 -> ARIMA(0,1,2)(0,1,0)12

```
modelo1<- arima(tsData, order=c(0,1,2),seasonal = list(order = c(0,1,0), period = 12), method="ML")
modelo1
```

```
##
## Call:
## arima(x = tsData, order = c(0, 1, 2), seasonal = list(order = c(0, 1, 0), period = 12),
##     method = "ML")
##
## Coefficients:
```

```
##           ma1     ma2
##       -1.3937  0.4856
## s.e.   0.1065  0.1347
##
## sigma^2 estimated as 3632246:  log likelihood = -844.66,  aic = 1695.32
```

Realizamos la diagnosis del modelo

```
coeftest(modelo1)
```

```
##
## z test of coefficients:
##
##     Estimate Std. Error  z value  Pr(>|z|)
## ma1 -1.39373    0.10646 -13.0917 < 2.2e-16 ***
## ma2  0.48562    0.13472   3.6046 0.0003126 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Todos los parámetros son significativos al tener un p-valor mayor que 0.05.

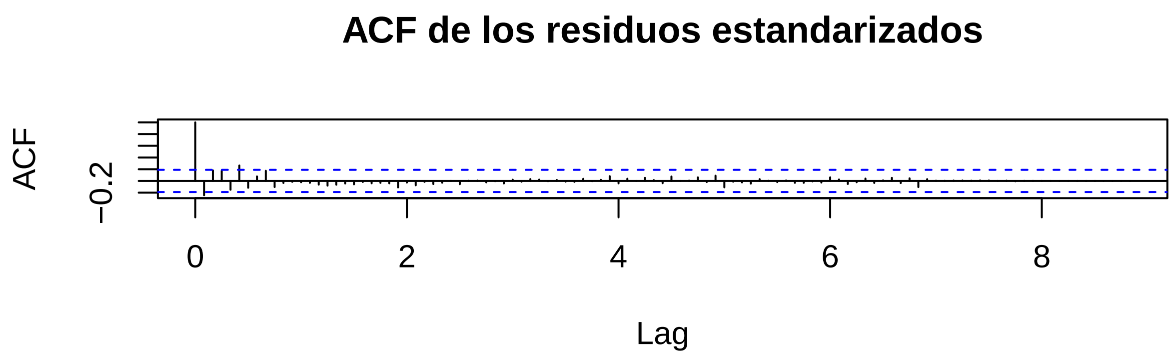
Obtenemos los intervalos de confianza al 95% para los parámetros.

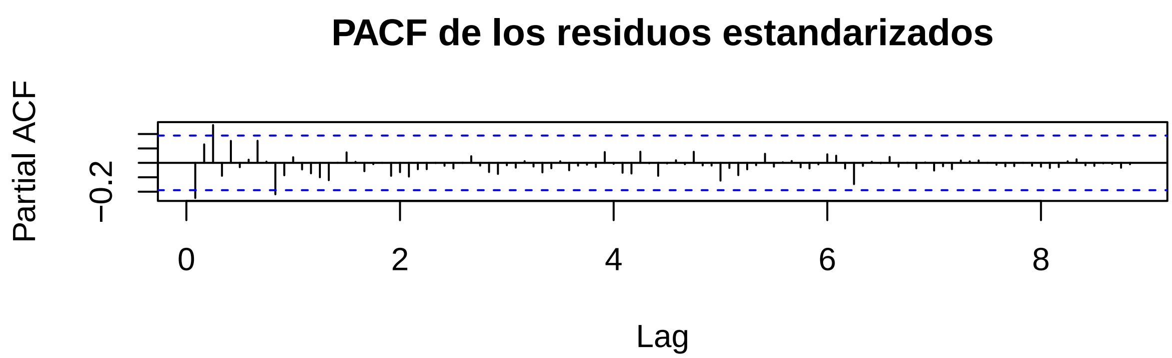
```
confint(modelo1, level = 0.95)
```

```
##          2.5 %     97.5 %
## ma1 -1.6023875 -1.1850755
## ma2  0.2215719  0.7496718
```

Visualizamos las funciones ACF y PACF de los residuos.

```
par(mfcol= c(2,1))
acf(modelo1$residuals, lag.max=107, main="ACF de los residuos estandarizados")
pacf(modelo1$residuals, lag.max=107, main="PACF de los residuos estandarizados")
```

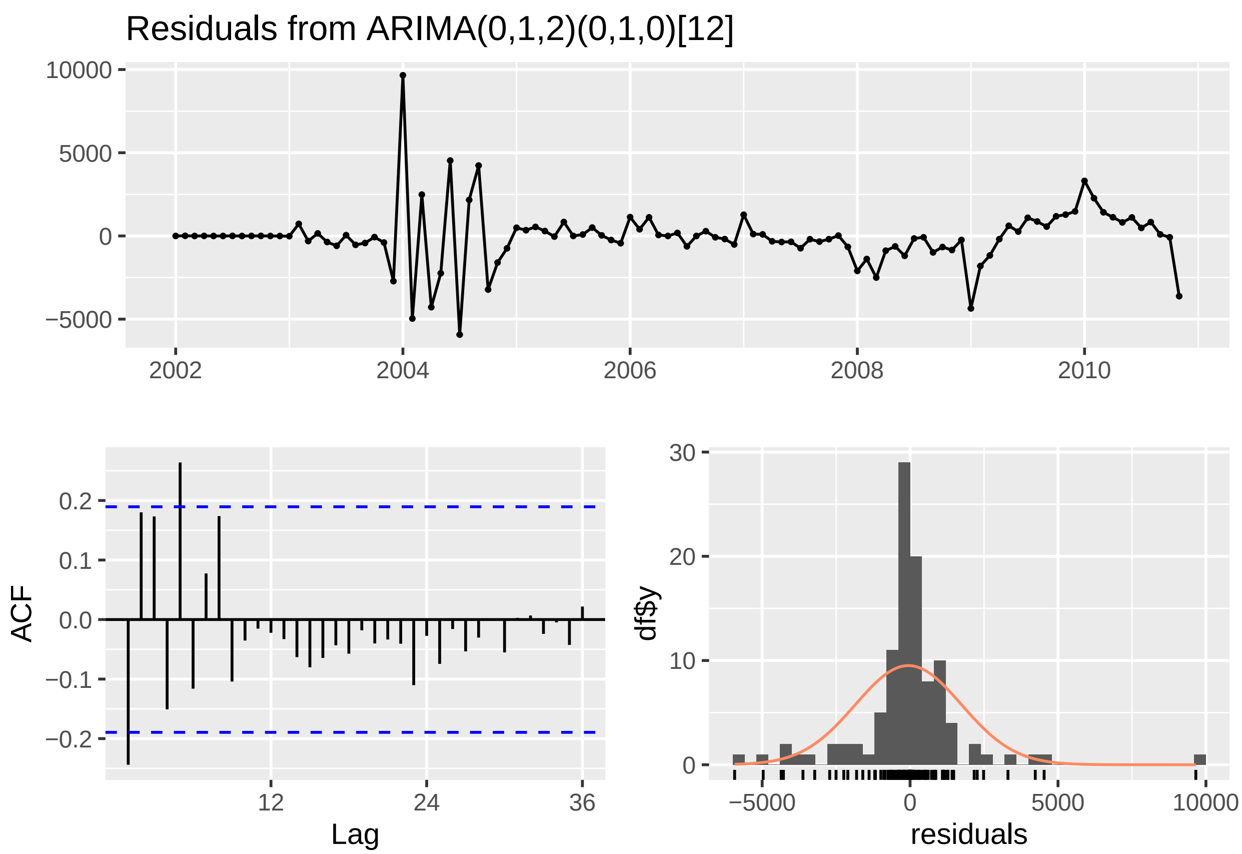




La ACF y la PACF de los residuos son muy parecidas, no muestran estructura y tienen casi todos los valores dentro de las bandas de confianza, por lo que podemos considerar que los residuos son aleatorios.

Realizamos el test de Ljung-Box

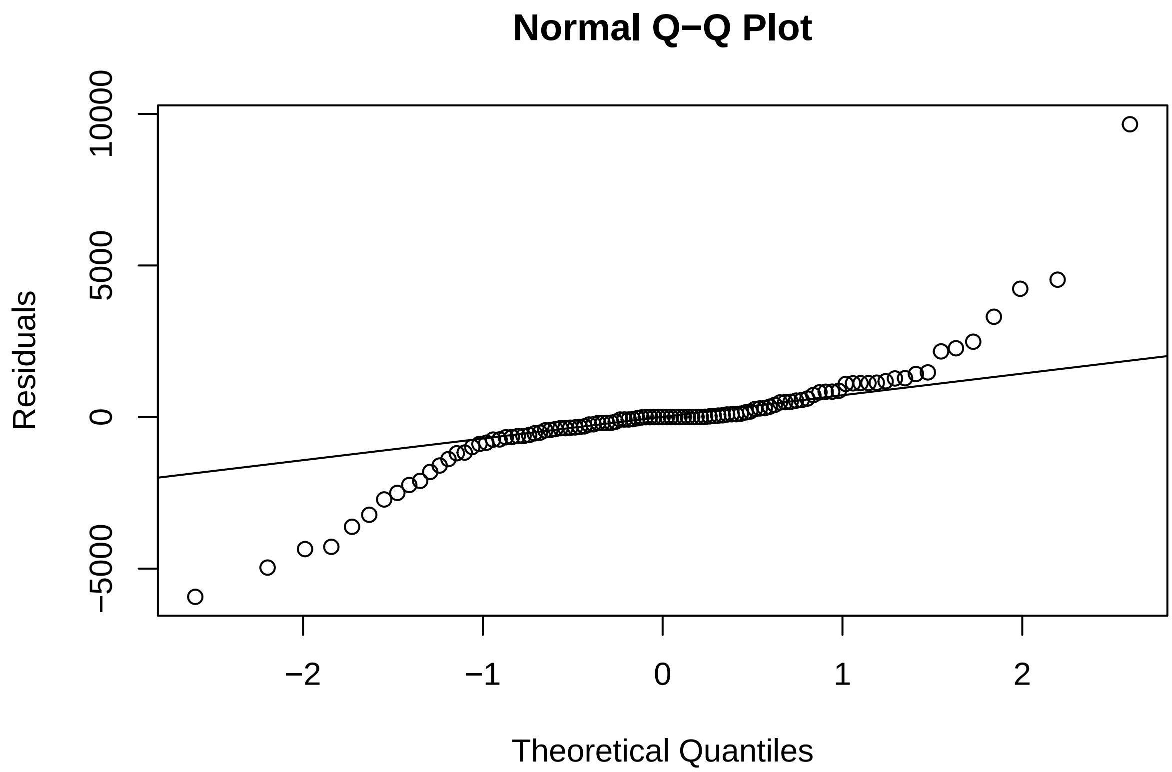
```
checkresiduals(modelo1)
```




```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(0,1,2)(0,1,0)[12]
## Q* = 34.457, df = 19, p-value = 0.01622
##
## Model df: 2.   Total lags used: 21
```

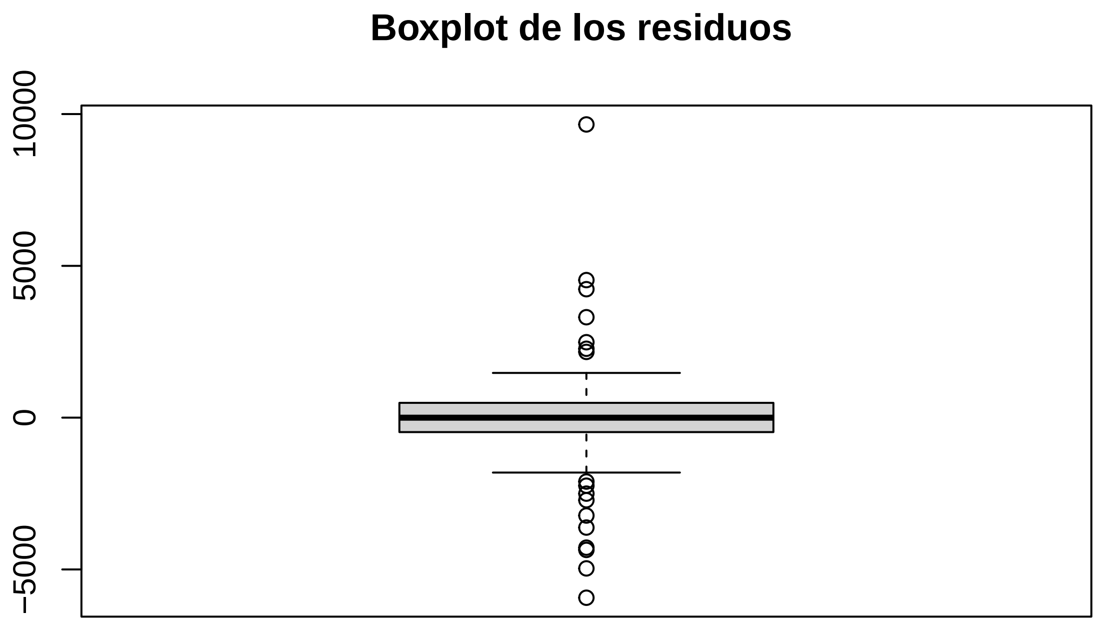
El p-valor de text de Ljung-Box es menor que 0.05 luego se puede rechazar que las primeras autocorrelaciones sean nulas, y no se puede asumir que los residuos sean ruido blanco. Además, también observamos una gran cantidad de datos atípicos.

```
qqnorm(modelo1$residuals, ylab="Residuals")
qqline(modelo1$residuals)
```



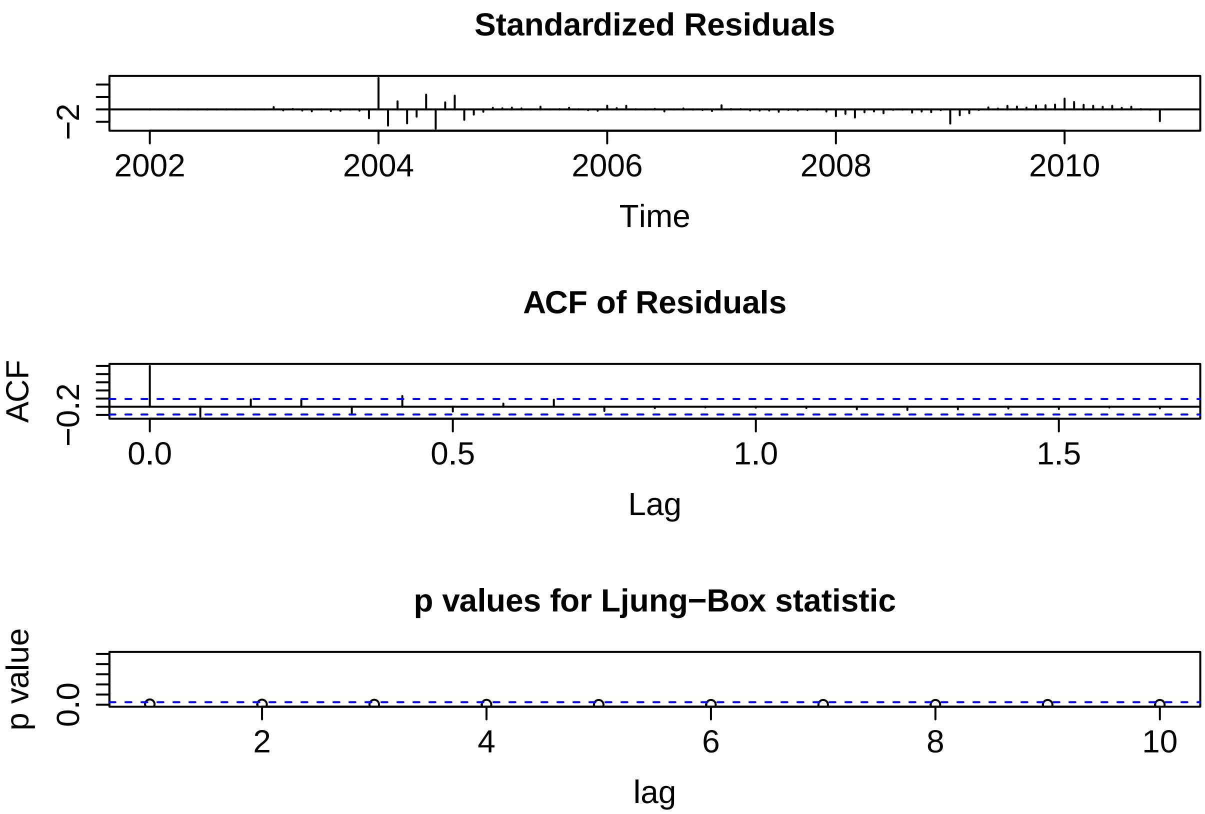
Podemos ver que los residuos siguen una distribución normal aproximadamente a pesar de la gran cantidad de valores atípicos.

```
boxplot(modelo1$residuals,main="Boxplot de los residuos ")
```



De igual forma, aquí se puede apreciar que hay un gran número de valores atípicos y también que la mayoría de los valores están en torno a 0.

```
par(cex.axis=1.5, cex.main=1.5, cex.lab=1.5)
tsdiag(modelo1)
```



Cogemos el siguiente modelo: Modelo2 -> ARIMA(2,1,0)(0,1,0)12.

```
modelo2<- arima(tsData, order=c(2,1,0),seasonal = list(order = c(0,1,0), period = 12), method="ML")
modelo2
```

```
##
## Call:
## arima(x = tsData, order = c(2, 1, 0), seasonal = list(order = c(0, 1, 0), period = 12),
##     method = "ML")
##
## Coefficients:
##           ar1      ar2
##       -1.2405  -0.6615
## s.e.   0.0762   0.0754
##
## sigma^2 estimated as 3815985:  log likelihood = -846.63,  aic = 1699.27
```

Realizamos la diagnosis

```
coeftest(modelo2)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error  z value  Pr(>|z|)
```

```
## ar1 -1.240469   0.076218 -16.2752 < 2.2e-16 ***
## ar2 -0.661513   0.075405  -8.7728 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Todos los paramétros son significativos ya que tienen un p-valor mayor que 0.05

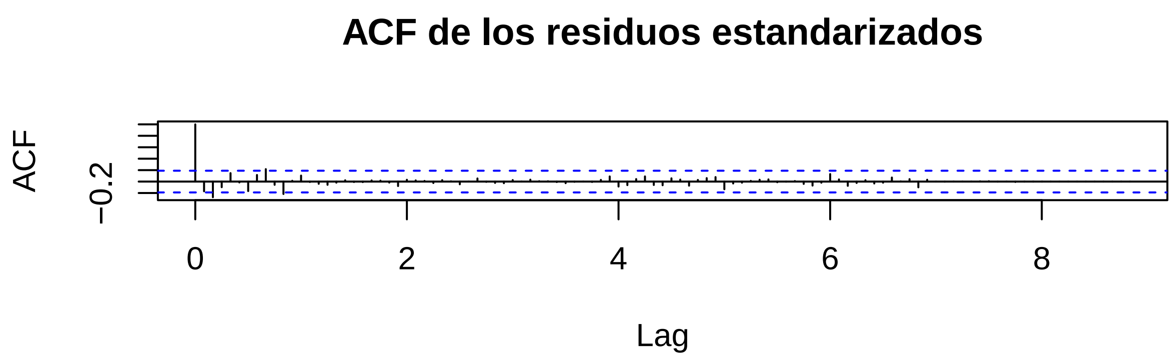
Obtenemos los intervalos de confianza al 95% para los parámetros.

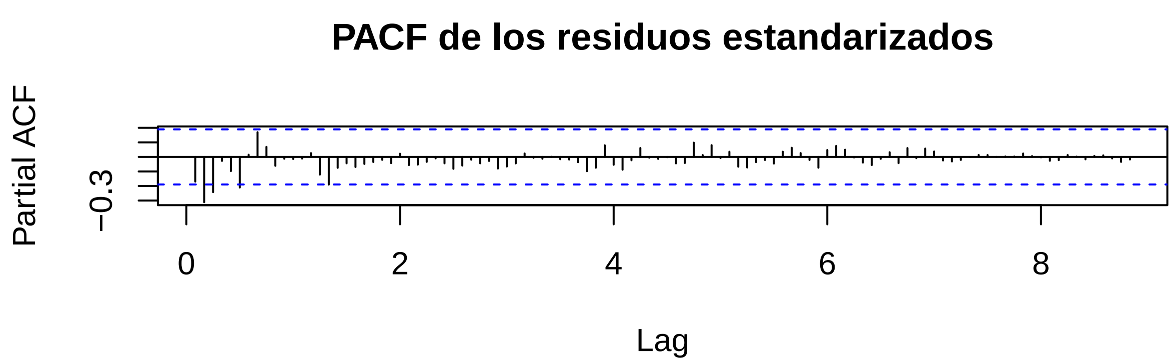
```r
confint(modelo1, level = 0.95)
```

```
##          2.5 %     97.5 %
## ma1 -1.6023875 -1.1850755
## ma2  0.2215719  0.7496718
```

Visualizamos las funciones ACF y PACF

```r
par(mfcol= c(2,1))
acf(modelo2$residuals, lag.max=275, main="ACF de los residuos estandarizados")
pacf(modelo2$residuals, lag.max=275, main="PACF de los residuos estandarizados")
```

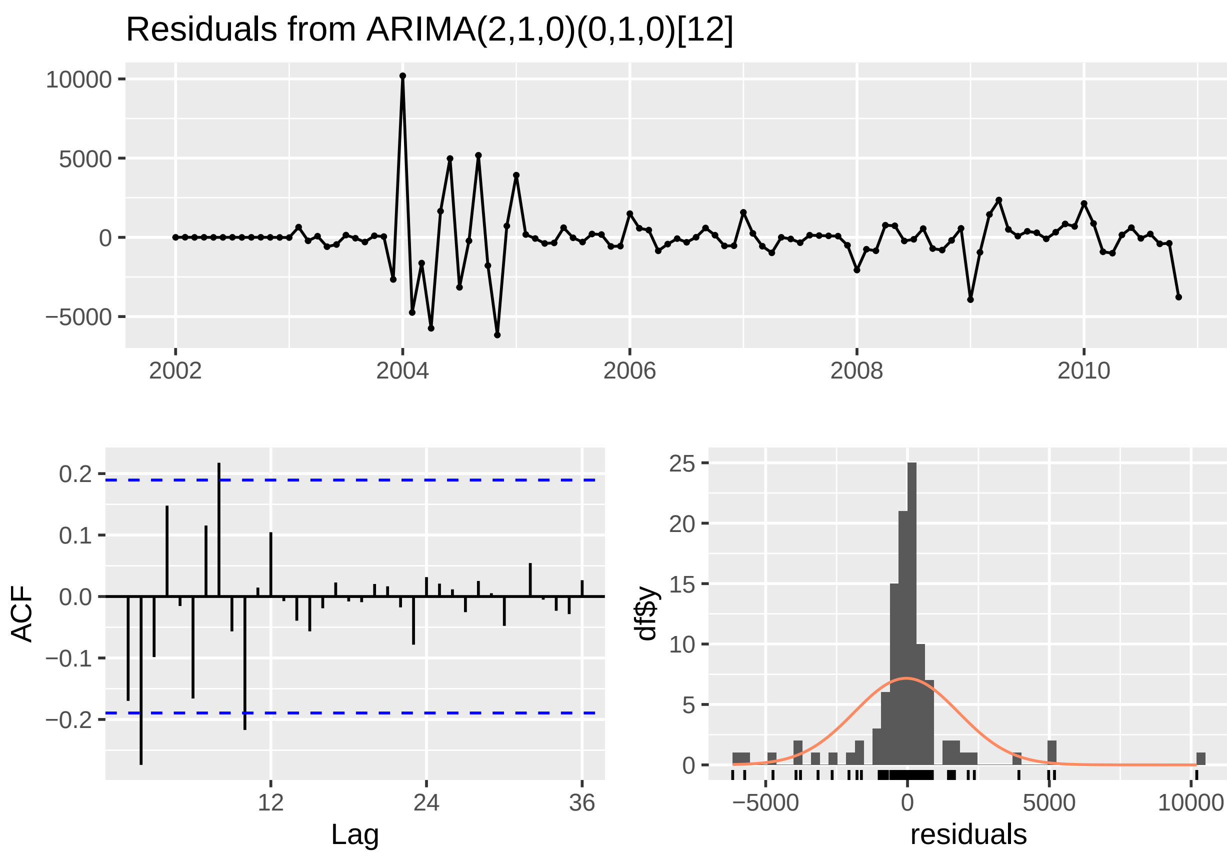




Vemos que también son muy parecidas y que no muestran estructura y casi todos los valores se encuentran dentro de las bandas así que podemos considerar que los residuos son aleatorios

Realizamos el test de Ljung-Box

```
checkresiduals(modelo2)
```

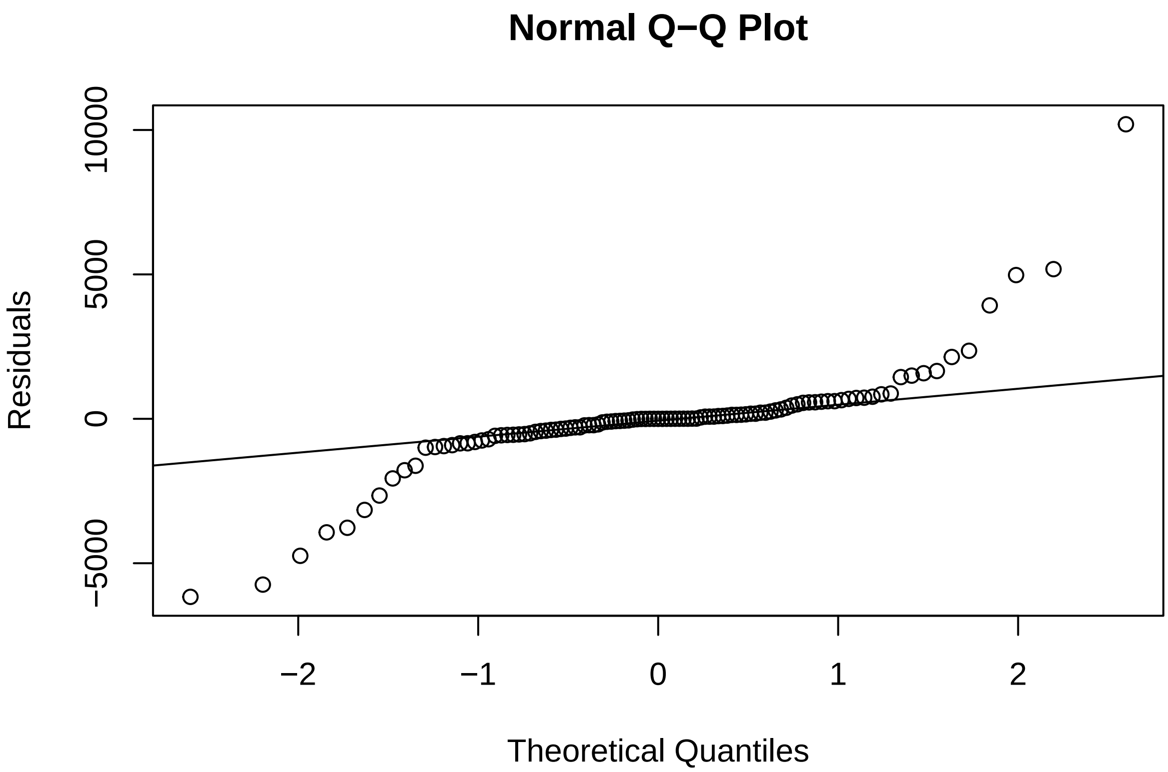



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,0)(0,1,0)[12]
## Q* = 33.66, df = 19, p-value = 0.02015
##
## Model df: 2.   Total lags used: 21
```

El p-valor de text de Ljung-Box es menor que 0.05 luego se puede rechazar que las primeras autocorrelaciones sean nulas, y no se puede asumir que los residuos sean ruido blanco.

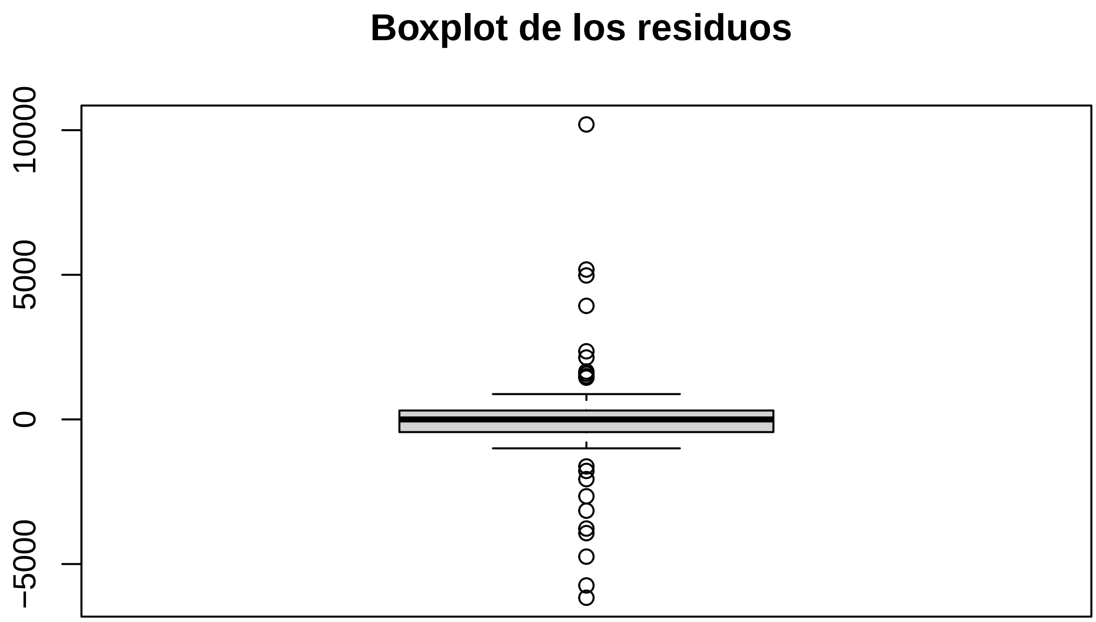
De igual forma que el modelo anterior, los residuos no parecen normales ya que el histograma no se aproxima a una campana de Gauss (también observamos muchos datos atípicos).

```
qqnorm(modelo2$residuals, ylab="Residuals")
qqline(modelo2$residuals)
```



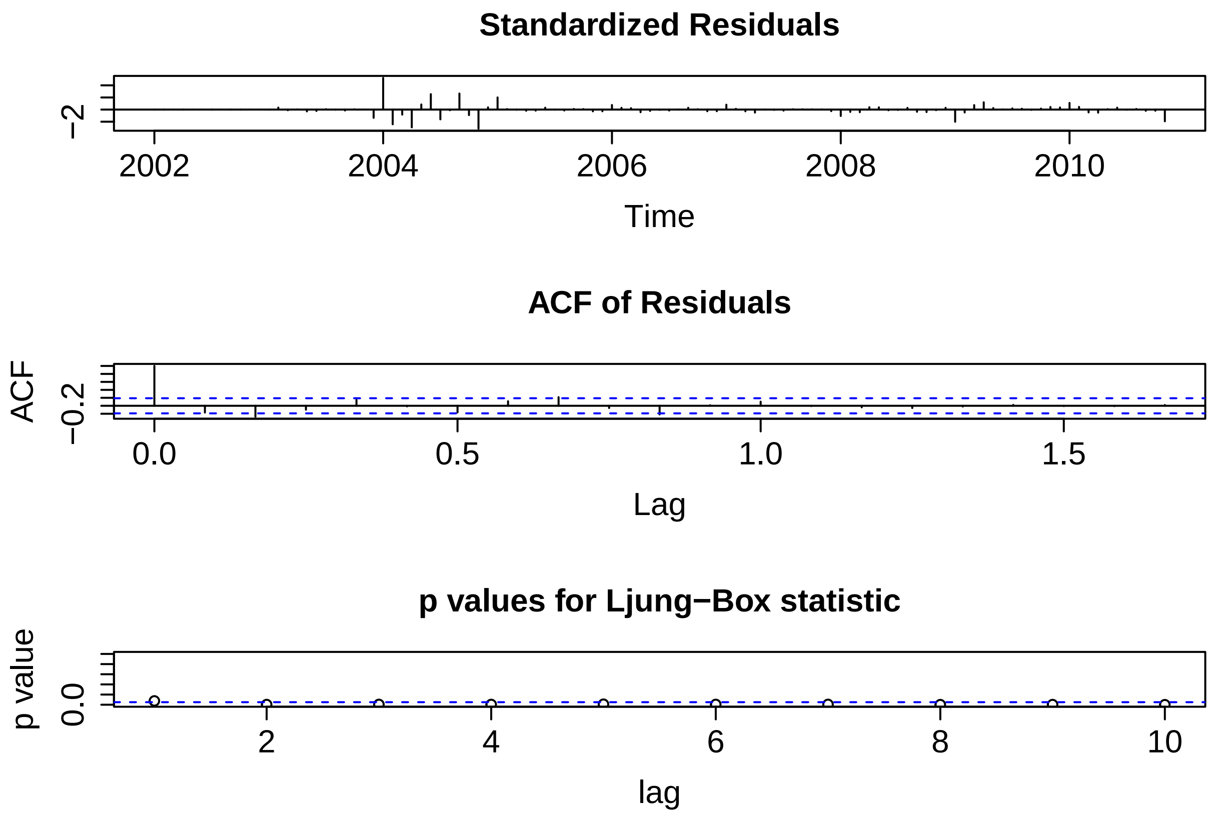
De igual forma, vemos que los valores atipicos afectan a la hora de la normalidad de los residuos.

```
boxplot(modelo2$residuals,main="Boxplot de los residuos ")
```



Esto también refleja que la mayoría de valores se encuentran en torno a 0 y que hay una gran cantidad de valores atípicos.

```
par(cex.axis=1.5, cex.main=1.5, cex.lab=1.5)
tsdiag(modelo2)
```



inguno de los modelos superan el diagnosis pero vamos a escoger el mejor de los dos. Para elegir uno de ellos tomamos como referencia el aic, que representa la capacidad predictiva del modelo y nos quedamos con el menor.

```
modelo1$aic
```

```
## [1] 1695.324
```

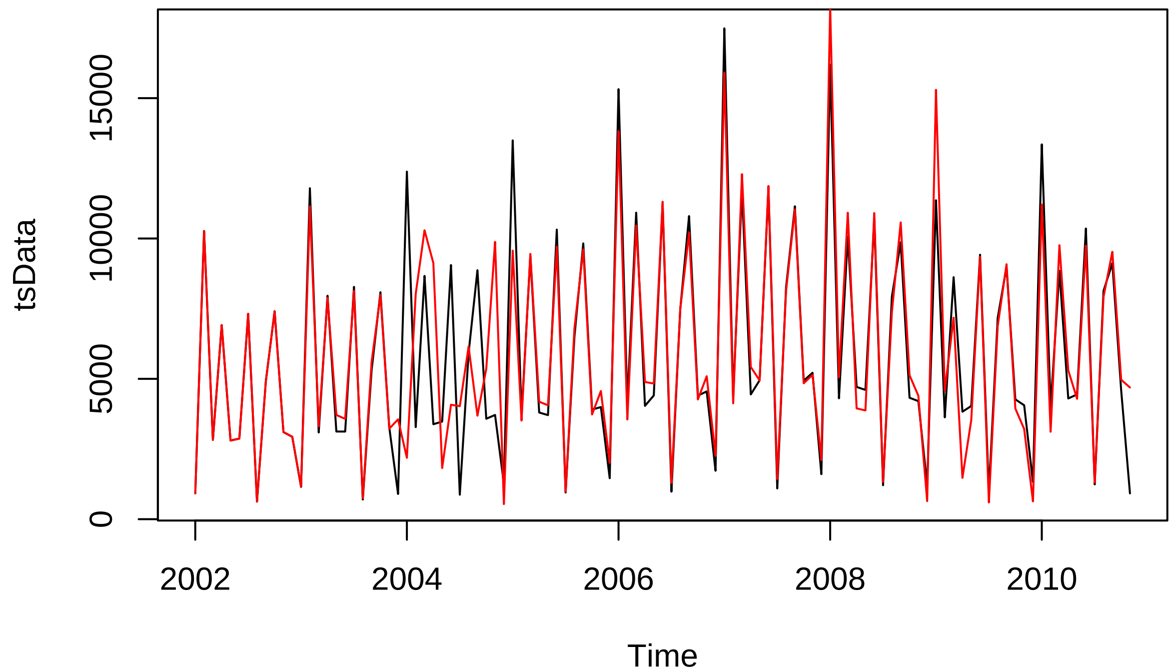
```
modelo2$aic
```

```
## [1] 1699.269
```

Por lo tanto, nos quedamos con el modelo ARIMA(0,1,2)(0,1,0)12, modelo1.

Veamos gráficamente la diferencia entre la serie original y el modelo ajustado (en rojo)

```
par(mfrow = c(1,1))
plot(tsData)
lines(tsData - modelo2$residuals,col="red")
```



Podemos ver como la línea roja ajusta en gran medida a nuestros datos quitando una pequeña zona en torno a 2004-2005. Esto nos indica que nuestro modelo es bastante acertado aunque el p-valor sea un poco justo.

```
predicciones<-forecast(modelo2,h=20,level=c(85,95),robust=TRUE,model="Arima")
```

```
## Warning in forecast.Arima(modelo2, h = 20, level = c(85, 95), robust = TRUE, :
## The non-existent robust arguments will be ignored.The non-existent model
## arguments will be ignored.
```

```
predicciones
```

```
##          Point Forecast      Lo 85     Hi 85     Lo 95     Hi 95
## Dec 2010     2349.58265  -462.4769  5161.642 -1479.118  6178.284
## Jan 2011    11501.84205  8609.6207 14394.063  7563.998 15439.686
## Feb 2011     2953.09488  -448.5864  6354.776 -1678.393  7584.583
## Mar 2011     8705.65899  5149.0989 12262.219  3863.299 13548.019
## Apr 2011     2508.73824 -1064.0136  6081.490 -2355.667  7373.144
## May 2011     4105.42684   147.1776  8063.676 -1283.845  9494.698
## Jun 2011     9305.79491  5319.3012 13292.289  3878.068 14733.522
## Jul 2011      110.47222 -4024.2857  4245.230 -5519.121  5740.065
## Aug 2011     7573.58461  3285.2085 11861.961  1734.836 13412.334
## Sep 2011     7902.16629  3560.9736 12243.359  1991.506 13812.826
## Oct 2011     3812.22195  -697.7132  8322.157 -2328.186  9952.630
## Nov 2011       32.04471 -4557.1935  4621.283 -6216.337  6280.426
## Dec 2011     1308.87400 -4625.2046  7242.953 -6770.547  9388.295
```

```
## Jan 2012    10723.66035  4776.4088 16670.912   2626.304 18821.016
## Feb 2012     1951.69410 -4557.1268  8460.515  -6910.255 10813.643
## Mar 2012     7807.48958   973.9027 14641.076  -1496.638 17111.617
## Apr 2012     1630.17585 -5324.5423  8584.894  -7838.875 11099.227
## May 2012     3134.25362 -4306.0858 10574.593  -6995.985 13264.492
## Jun 2012     8436.53228   848.5740 16024.491  -1894.694 18767.758
## Jul 2012     -823.94408 -8678.1027  7030.215 -11517.610  9869.721
```

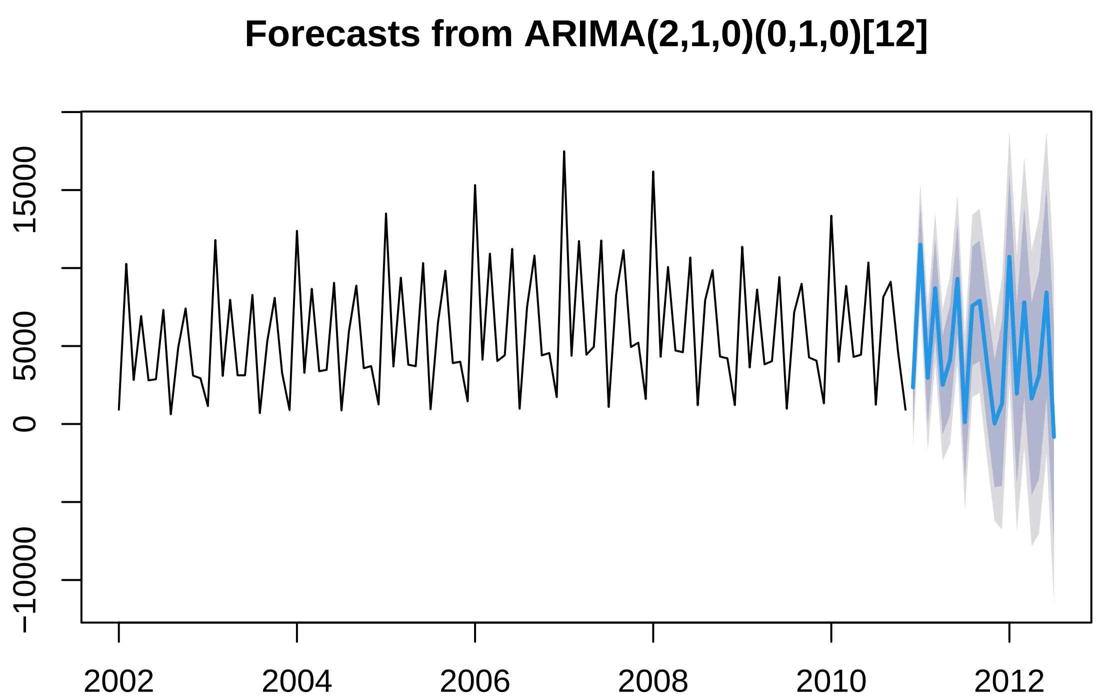
```r
accuracy(predicciones)
```

```
##                     ME     RMSE      MAE       MPE     MAPE      MASE
## Training set -42.51128 1830.946 937.4301 -12.90841 27.23178 0.7548431
##                    ACF1
## Training set -0.1697518
```

En estas medidas podemos estimar la precisión de nuestro modelo. Un error medio de -42 cuando nuestros datos están entre 500 y 15000 podríamos decir que no es un error malo, pero si nos fijamos en el MAE (error cuadrático medio) vemos que es bastante más elevado. Además el RMSE es bastante malo también por lo que no podemos decir que nuestro modelo sea demasiado bueno según estos errores. Por último, como hemos dicho, nuestros datos tienen un rango muy grande por lo que vamos a analizar el MAPE (error porcentual absoluto) ya que se olvida de la escala. En este caso tenemos un 27% que tampoco es un valor bueno.

Aun así, vamos a hacer un gráfico con nuestras predicciones para ver cómo quedarían.

```r
plot(forecast(modelo2,h=20))
```



Los pronósticos se muestran como una línea azul, con los intervalos de predicción del 80% como un área sombreada oscura, y los intervalos de predicción del 95% como un área sombreada clara.

Ahora vamos a buscar el mejor modelo con la función auto.arima() y ver si su AIC es mejor que el nuestro anterior

```r
fit<-auto.arima(tsData,  stepwise = FALSE, trace=TRUE)
```

```
##
##  ARIMA(0,1,0)(0,1,0)[12]                    : 1825.4
##  ARIMA(0,1,0)(0,1,1)[12]                    : 1827.481
##  ARIMA(0,1,0)(0,1,2)[12]                    : 1825.017
##  ARIMA(0,1,0)(1,1,0)[12]                    : 1827.48
##  ARIMA(0,1,0)(1,1,1)[12]                    : 1829.583
##  ARIMA(0,1,0)(1,1,2)[12]                    : Inf
##  ARIMA(0,1,0)(2,1,0)[12]                    : 1828.59
##  ARIMA(0,1,0)(2,1,1)[12]                    : 1830.623
##  ARIMA(0,1,0)(2,1,2)[12]                    : Inf
##  ARIMA(0,1,1)(0,1,0)[12]                    : 1728.075
##  ARIMA(0,1,1)(0,1,1)[12]                    : 1730.204
##  ARIMA(0,1,1)(0,1,2)[12]                    : 1731.219
##  ARIMA(0,1,1)(1,1,0)[12]                    : 1730.204
##  ARIMA(0,1,1)(1,1,1)[12]                    : Inf
##  ARIMA(0,1,1)(1,1,2)[12]                    : 1732.47
##  ARIMA(0,1,1)(2,1,0)[12]                    : 1731.953
##  ARIMA(0,1,1)(2,1,1)[12]                    : 1734.112
##  ARIMA(0,1,1)(2,1,2)[12]                    : Inf
##  ARIMA(0,1,2)(0,1,0)[12]                    : 1688.127
##  ARIMA(0,1,2)(0,1,1)[12]                    : 1690.178
##  ARIMA(0,1,2)(0,1,2)[12]                    : 1692.145
##  ARIMA(0,1,2)(1,1,0)[12]                    : 1690.185
##  ARIMA(0,1,2)(1,1,1)[12]                    : Inf
##  ARIMA(0,1,2)(1,1,2)[12]                    : Inf
##  ARIMA(0,1,2)(2,1,0)[12]                    : 1692.22
##  ARIMA(0,1,2)(2,1,1)[12]                    : 1701.786
##  ARIMA(0,1,3)(0,1,0)[12]                    : 1686.759
##  ARIMA(0,1,3)(0,1,1)[12]                    : 1688.947
##  ARIMA(0,1,3)(0,1,2)[12]                    : 1690.799
##  ARIMA(0,1,3)(1,1,0)[12]                    : 1688.951
##  ARIMA(0,1,3)(1,1,1)[12]                    : Inf
##  ARIMA(0,1,3)(2,1,0)[12]                    : 1690.899
##  ARIMA(0,1,4)(0,1,0)[12]                    : 1686.879
##  ARIMA(0,1,4)(0,1,1)[12]                    : 1689.144
##  ARIMA(0,1,4)(1,1,0)[12]                    : 1690.075
##  ARIMA(0,1,5)(0,1,0)[12]                    : 1687.865
##  ARIMA(1,1,0)(0,1,0)[12]                    : 1752.077
##  ARIMA(1,1,0)(0,1,1)[12]                    : 1754.183
##  ARIMA(1,1,0)(0,1,2)[12]                    : 1753.973
##  ARIMA(1,1,0)(1,1,0)[12]                    : 1754.181
##  ARIMA(1,1,0)(1,1,1)[12]                    : 1756.33
##  ARIMA(1,1,0)(1,1,2)[12]                    : 1755.394
##  ARIMA(1,1,0)(2,1,0)[12]                    : 1755.823
##  ARIMA(1,1,0)(2,1,1)[12]                    : Inf
##  ARIMA(1,1,0)(2,1,2)[12]                    : Inf
```

```
##  ARIMA(1,1,1)(0,1,0)[12]                    : 1693.314
##  ARIMA(1,1,1)(0,1,1)[12]                    : 1695.482
##  ARIMA(1,1,1)(0,1,2)[12]                    : 1697.607
##  ARIMA(1,1,1)(1,1,0)[12]                    : 1695.482
##  ARIMA(1,1,1)(1,1,1)[12]                    : 1697.717
##  ARIMA(1,1,1)(1,1,2)[12]                    : Inf
##  ARIMA(1,1,1)(2,1,0)[12]                    : 1697.638
##  ARIMA(1,1,1)(2,1,1)[12]                    : 1699.733
##  ARIMA(1,1,2)(0,1,0)[12]                    : 1686.101
##  ARIMA(1,1,2)(0,1,1)[12]                    : 1688.259
##  ARIMA(1,1,2)(0,1,2)[12]                    : 1690.081
##  ARIMA(1,1,2)(1,1,0)[12]                    : 1688.264
##  ARIMA(1,1,2)(1,1,1)[12]                    : Inf
##  ARIMA(1,1,2)(2,1,0)[12]                    : 1690.184
##  ARIMA(1,1,3)(0,1,0)[12]                    : 1688.15
##  ARIMA(1,1,3)(0,1,1)[12]                    : 1690.396
##  ARIMA(1,1,3)(1,1,0)[12]                    : 1690.399
##  ARIMA(1,1,4)(0,1,0)[12]                    : 1690.285
##  ARIMA(2,1,0)(0,1,0)[12]                    : 1699.536
##  ARIMA(2,1,0)(0,1,1)[12]                    : 1700.808
##  ARIMA(2,1,0)(0,1,2)[12]                    : 1702.511
##  ARIMA(2,1,0)(1,1,0)[12]                    : 1700.767
##  ARIMA(2,1,0)(1,1,1)[12]                    : 1702.924
##  ARIMA(2,1,0)(1,1,2)[12]                    : 1703.379
##  ARIMA(2,1,0)(2,1,0)[12]                    : 1702.851
##  ARIMA(2,1,0)(2,1,1)[12]                    : 1705.256
##  ARIMA(2,1,1)(0,1,0)[12]                    : 1683.293
##  ARIMA(2,1,1)(0,1,1)[12]                    : 1685.484
##  ARIMA(2,1,1)(0,1,2)[12]                    : 1687.438
##  ARIMA(2,1,1)(1,1,0)[12]                    : 1685.486
##  ARIMA(2,1,1)(1,1,1)[12]                    : 1687.705
##  ARIMA(2,1,1)(2,1,0)[12]                    : 1687.519
##  ARIMA(2,1,2)(0,1,0)[12]                    : 1684.022
##  ARIMA(2,1,2)(0,1,1)[12]                    : 1686.246
##  ARIMA(2,1,2)(1,1,0)[12]                    : 1686.249
##  ARIMA(2,1,3)(0,1,0)[12]                    : 1685.778
##  ARIMA(3,1,0)(0,1,0)[12]                    : 1696.201
##  ARIMA(3,1,0)(0,1,1)[12]                    : 1697.613
##  ARIMA(3,1,0)(0,1,2)[12]                    : 1699.862
##  ARIMA(3,1,0)(1,1,0)[12]                    : 1697.595
##  ARIMA(3,1,0)(1,1,1)[12]                    : 1699.853
##  ARIMA(3,1,0)(2,1,0)[12]                    : 1699.841
##  ARIMA(3,1,1)(0,1,0)[12]                    : 1685.047
##  ARIMA(3,1,1)(0,1,1)[12]                    : 1687.224
##  ARIMA(3,1,1)(1,1,0)[12]                    : 1687.23
##  ARIMA(3,1,2)(0,1,0)[12]                    : 1686.066
##  ARIMA(4,1,0)(0,1,0)[12]                    : 1687.323
##  ARIMA(4,1,0)(0,1,1)[12]                    : 1689.602
##  ARIMA(4,1,0)(1,1,0)[12]                    : 1689.602
##  ARIMA(4,1,1)(0,1,0)[12]                    : 1685.039
##  ARIMA(5,1,0)(0,1,0)[12]                    : 1687.215
##
##
##
```

```
##  Best model: ARIMA(2,1,1)(0,1,0)[12]
```

El mejor modelo es el ARIMA(2,1,1)(0,1,0)12.

```
fit
```

```
## Series: tsData
## ARIMA(2,1,1)(0,1,0)[12]
##
## Coefficients:
##           ar1      ar2      ma1
##       -0.9126  -0.4108  -0.7189
## s.e.   0.1094   0.1068   0.0859
##
## sigma^2 = 3208476:  log likelihood = -837.42
## AIC=1682.84   AICc=1683.29   BIC=1693.02
```

El AIC de este modelo es 1683.29 que es menor que el de nuestro modelo anterior (1695.324)

Hacemos los diagnosis de los parámetros del modelo Analizamos la significatividad individual de los parámetros del modelo

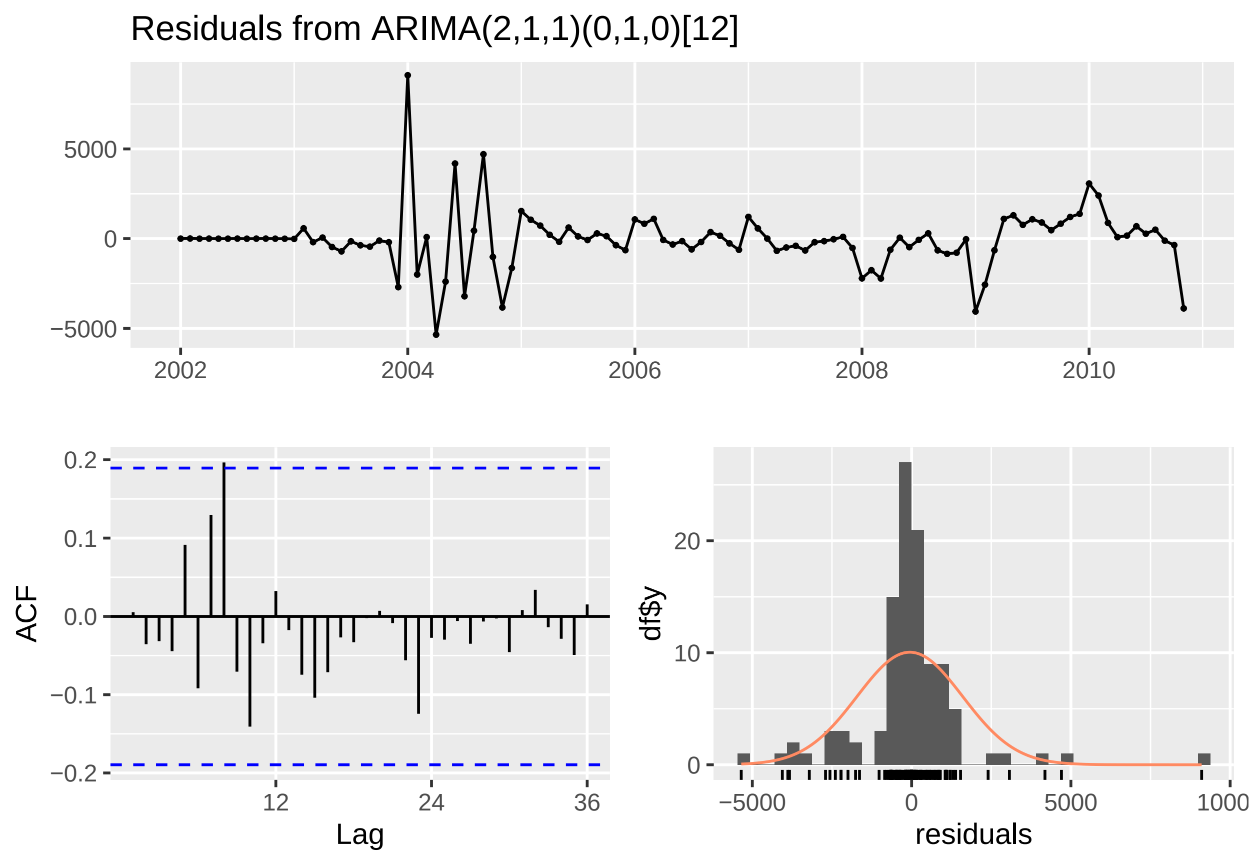
```
coeftest(fit)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ar1 -0.912604   0.109433 -8.3394 < 2.2e-16 ***
## ar2 -0.410788   0.106818 -3.8457 0.0001202 ***
## ma1 -0.718917   0.085907 -8.3685 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Todos los p-valores son menores que 0.05 y por tanto, son todos significativos

Hacemos la diagnosis de los residuos

```
checkresiduals(fit)
```

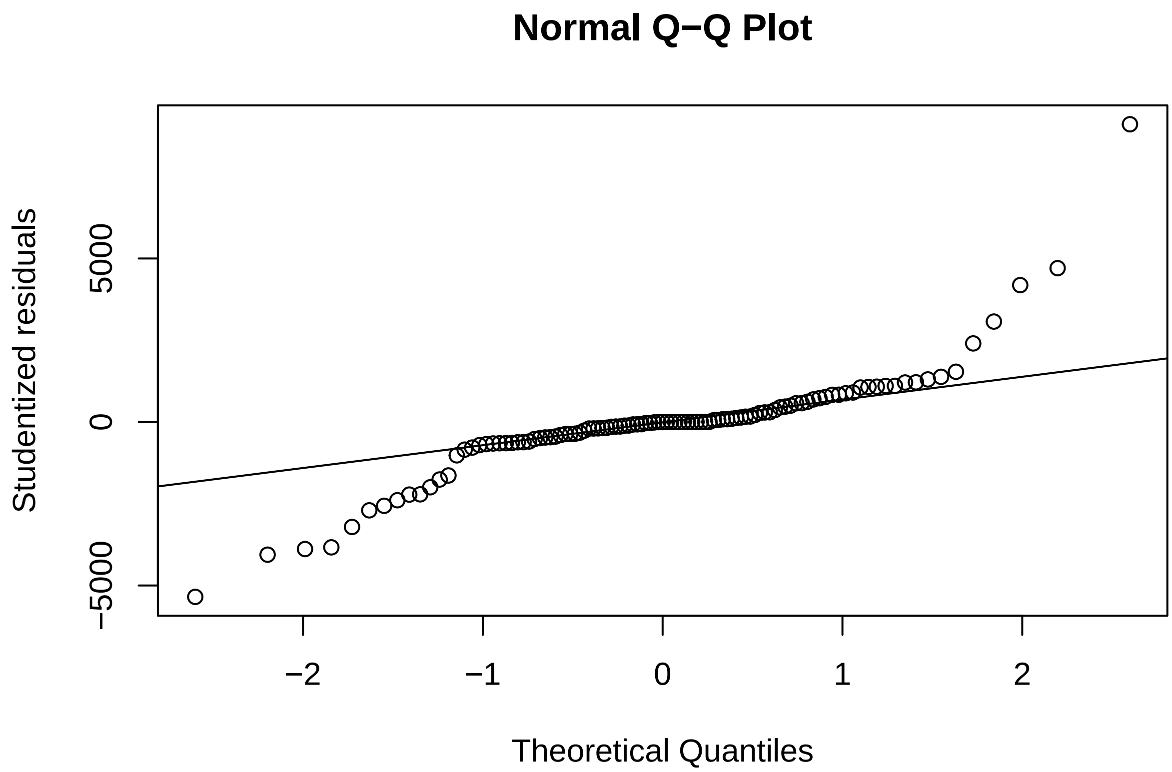



```
##
##  Ljung-Box test
##
## data:  Residuals from ARIMA(2,1,1)(0,1,0)[12]
## Q* = 15.183, df = 18, p-value = 0.6494
##
## Model df: 3.   Total lags used: 21
```

El p-valor de text de Ljung-Box es mayor que el de nuestro modelo anterior y también mayor que 0.05 (0.6494) lo cual concuerda con el ACF ya que no hay casi ningún valor que salga de las bandas de confianza aunque observamos que tenemos muchos datos atípicos.

Analizamos la normalidad con el qq-plot y vemos como se ajusta mucho más a la normal.

```
qqnorm(fit$residuals, ylab="Studentized residuals")
qqline(fit$residuals)
```



Vemos que siguen una distribución normal a pesar de la gran cantidad de valores atípicos.

d) Predecir la recaudación para los próximos 5 meses

Hacemos la predicción para los 5 próximos meses estableciendo la h a 5

```r
predicciones<-forecast(fit,h=5)
predicciones
```

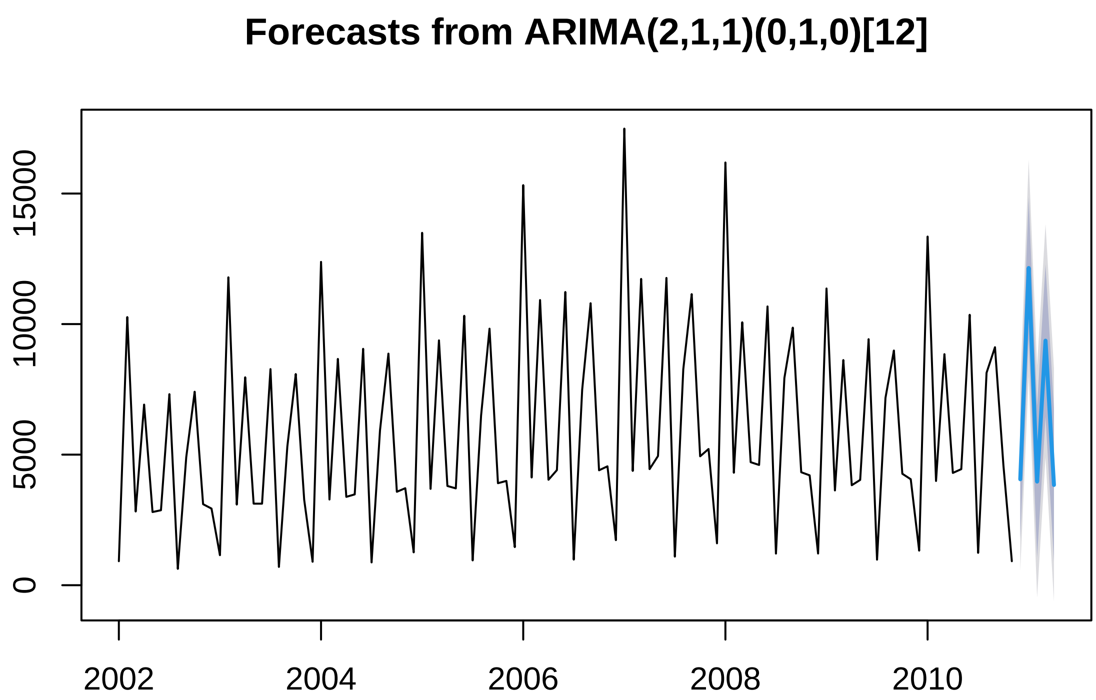
```
##          Point Forecast     Lo 80     Hi 80     Lo 95     Hi 95
## Dec 2010       4056.935 1761.3918  6352.478  546.2045  7567.666
## Jan 2011      12148.768 9433.7896 14863.747 7996.5668 16300.970
## Feb 2011       3967.711 1065.6022  6869.821 -470.6817  8406.105
## Mar 2011       9365.428 6447.3243 12283.531 4902.5736 13828.282
## Apr 2011       3838.581  919.9708  6757.190 -625.0479  8302.209
```

```r
accuracy(predicciones)
```

```
##                    ME     RMSE      MAE       MPE     MAPE      MASE      ACF1
## Training set -55.4848 1651.879 929.2924 -14.08587 29.44954 0.7482904 0.0052732
```

Si lo comparamos con el modelo que sacamos anteriormente, vemos que da un modelo parecido a grandes rasgos, ya que tanto el RMSE como el MAE disminuyen pero el MPE y el MAPE aumentan.

```
plot(forecast(fit,h=5))
```



Según estos resultados, el mejor modelo para predecir el IVA es el ARIMA(0,1,2)(0,1,0)12 aunque no sea del todo preciso al tener errores bastante grandes.