

Documentación sobre autoajuste de modelos ARIMAX

Ana Xiangning Pereira Ezquerro

Versión 25 abril, 2022

Índice

1	Función de auto-ajuste de modelos ARIMAX (<code>auto.fit.arima</code> en <code>auto_fit_arima.R</code>)	2
2	Función de selección automática de múltiples variables y retardos en modelos ARIMAX (<code>auto.fit.arima.regression</code> en <code>automatic_selection.R</code>)	9
3	Funciones auxiliares	13
3.1	Ajuste de los coeficientes de un modelo (<code>fit.coefficients()</code> de <code>auto_fit_arima.R</code>)	13
3.2	Ajuste de un ARIMA vía múltiples optimizadores (<code>fit.model()</code> de <code>auto_fit_arima.R</code>)	13
3.3	Selección del retardo óptimo (<code>select.optimal.lag()</code> de <code>automatic_selection.R</code>)	14
4	Predicciones puntuales a horizonte h e intervalos de confianza (<code>forecasting_model()</code> de <code>forecasting.R</code>)	15
5	Comprobación con ejemplos	17
5.1	Evolución de la gripe en Cataluña	17

1 Función de auto-ajuste de modelos ARIMAX (`auto.fit.arima` en `auto_fit_arima.R`)

Descripción: Obtiene el ajuste de un modelo válido para una serie temporal y, opcionalmente, una o varias variables regresoras. En el ajuste obtenido todos los parámetros son estadísticamente significativos y se verifica que se cumplen las hipótesis de independencia y media nula sobre sus residuos. Este ajuste es escogido por un criterio de información que se introduce como argumento.

Devuelve:

- Ajuste para la serie temporal (objeto Arima) si lo hay y se puede optimizar. En caso de que no exista, devuelve NA.
- Si `plot_result = TRUE` y se ha conseguido ajustar un modelo válido para la serie, devuelve un objeto de tipo list donde se encuentra el ajuste (`$ajuste`), el gráfico de la serie (`$fig_serie`) y el gráfico de los residuos del ajuste (`$fig_residuales`).

```
auto.fit.arima(serie, xregs = NULL, seasonal = TRUE, ic = c("aicc", "aic", "bic"),  
              d = NA, D = NA, alpha = 0.05, show_info = TRUE, plot_result = FALSE)
```

Argumentos:

- `serie [ts]`: Serie temporal sobre la que se quiere obtener un ajuste válido de un modelo ARIMAX.
- `xregs [ts]`: Se pueden introducir series de tiempo que actuarán como variables regresoras sobre `serie`. Por defecto, `xregs=NULL`, i.e. no hay variables regresoras.
- `ic [character]`: Criterio de información para escoger modelos.
 - "aicc": Criterio de Información de Akaike Corregido (por defecto).
 - "aic": Criterio de Información de Akaike.
 - "bic": Criterio de Información Bayesiano.
- `d [numeric]`: Orden de diferenciación regular de `serie` sobre el que se limita la búsqueda de modelos. Si no se introduce ningún valor el valor máximo de la búsqueda es `d=4`.
- `D [numeric]`: Orden de **de** diferenciación estacional de `serie` sobre el que se limita la búsqueda de modelos. Si no se introduce ningún valor el valor máximo de la búsqueda es `D=3`.
- `alpha [numeric]`: Valor entre 0 y 1 que indica el nivel de significación de los tests para chequear:
 - La significación de los parámetros de los ajustes.
 - La validez del modelo a partir del test de independencia de residuos y el test de media nula de los residuos.
- `show_info [boolean]`: Indica si se muestra la información de la búsqueda del mejor ajuste o no. Por defecto `TRUE`.
- `plot_results [boolean]`: Indica si se deben devolver los gráficos de la serie temporal y los residuos del modelo obtenido. Por defecto `FALSE`.

Ejemplo de uso: Evolución de la gripe en Cataluña.

```
dat <- read.csv("data/evolucion_gripe_covid.csv")  
gripe <- ts(dat$sdgripal, start=c(2020, 40), frequency=52)
```

```
result_gripe <- auto.fit.arima(gripe, plot_result = TRUE)
```

Series: serie

ARIMA(2,0,1) with non-zero mean

Coefficients:

	ar1	ar2	ma1	mean
	1.7693	-0.8833	-0.7882	251.3508
s.e.	0.1032	0.0946	0.1667	18.7800

$\sigma^2 = 3824$: log likelihood = -331.74

AIC=673.48 AICc=674.59 BIC=683.95

Falla la hipótesis de normalidad sobre los residuos.

El modelo es válido pero los intervalos de predicción basados en la dist. asintótica no son válidos

MODELO FINAL

Series: serie

ARIMA(2,0,1) with non-zero mean

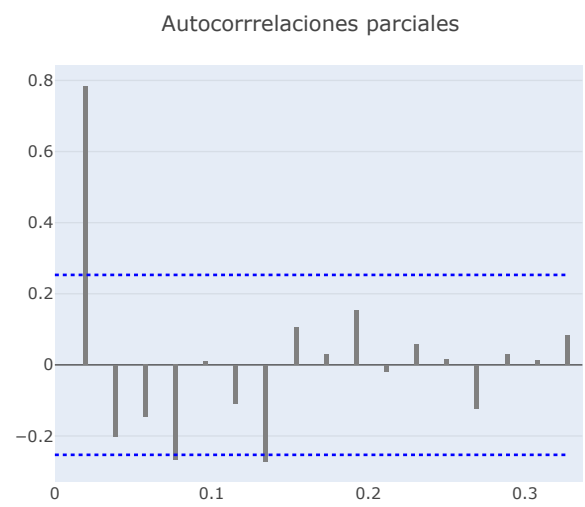
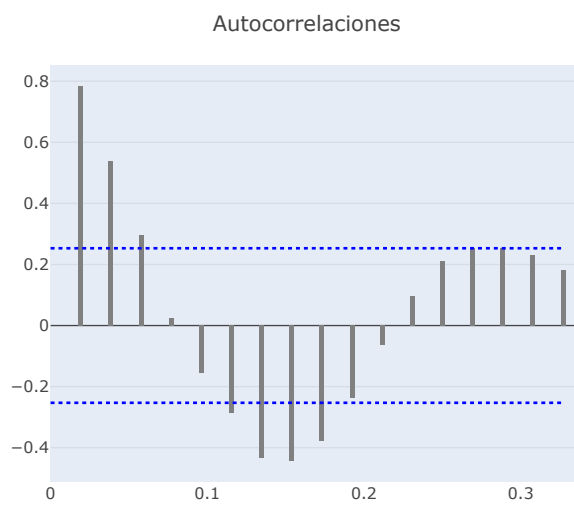
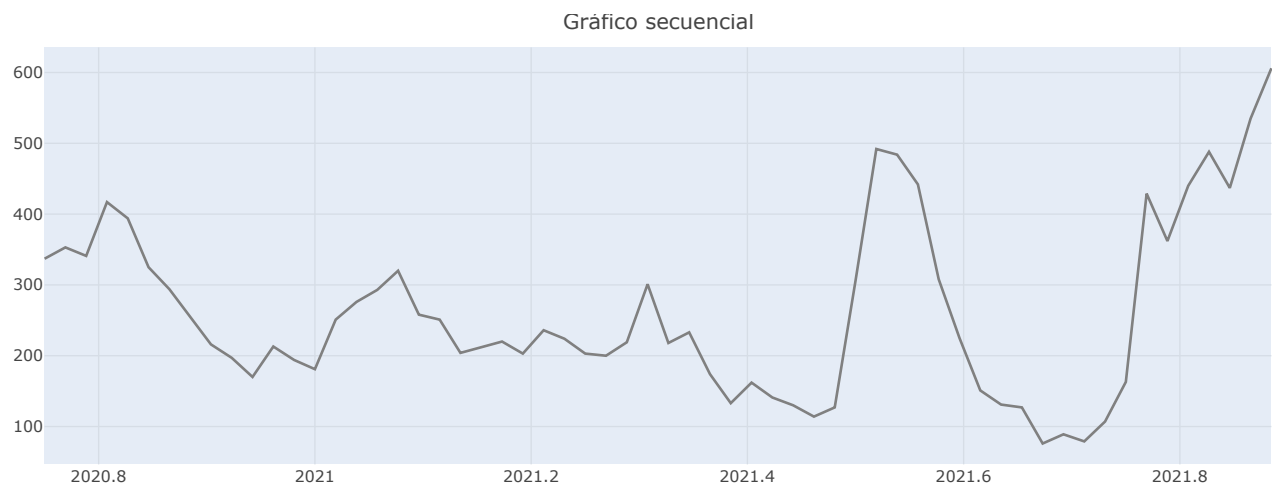
Coefficients:

	ar1	ar2	ma1	mean
	1.7693	-0.8833	-0.7882	251.3508
s.e.	0.1032	0.0946	0.1667	18.7800

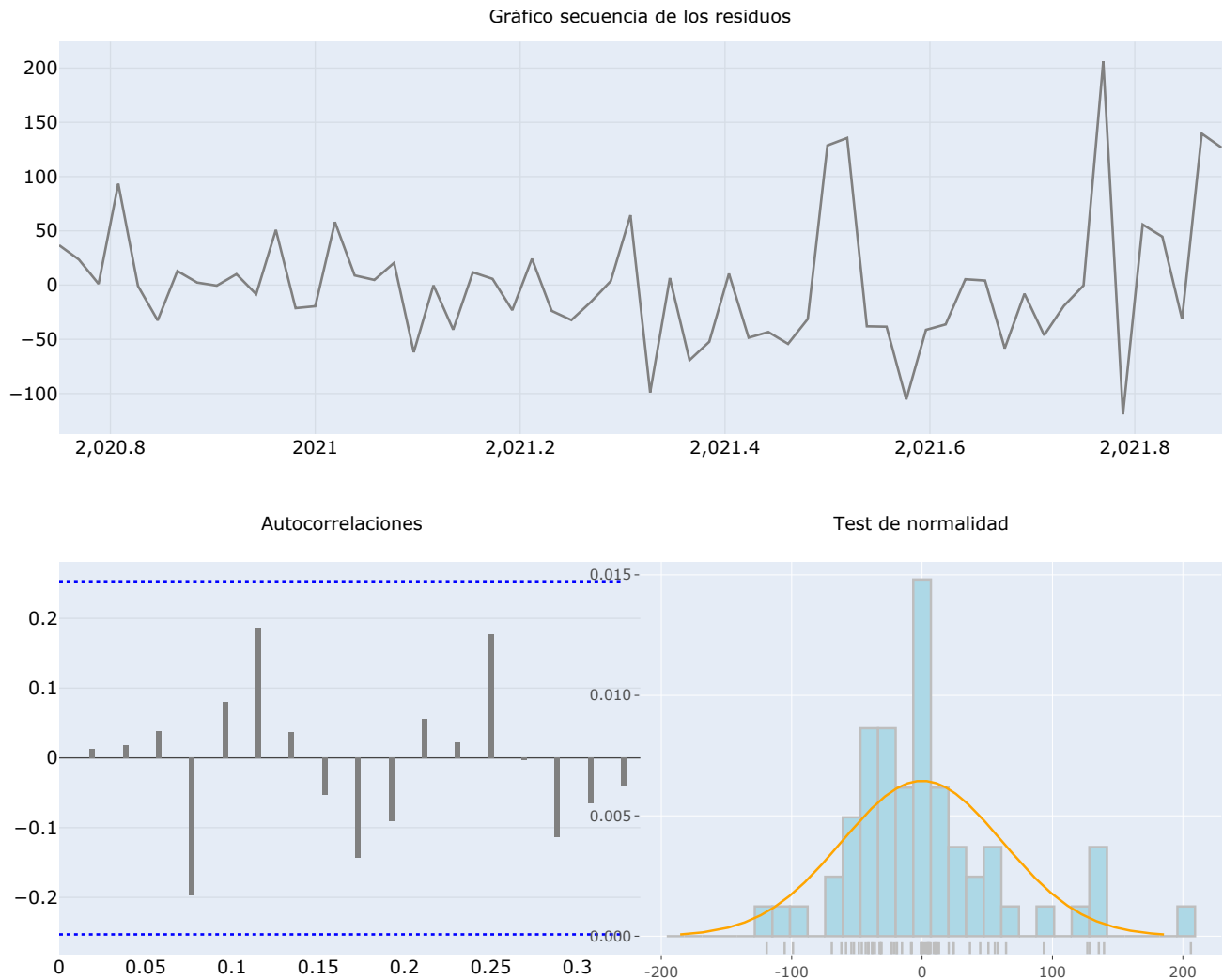
$\sigma^2 = 3824$: log likelihood = -331.74

AIC=673.48 AICc=674.59 BIC=683.95

```
display(result_gripe$fig_serie, "serie gripe", width=1000, height=800)
```



```
display(result_gripe$fig_residuals, "residuals gripe", width=1000, height=800)
```



Nivel mensual de dióxido de carbono (Co2) medido en el Observatorio de Mauna Loa (Hawaii). La serie comienza en Marzo de 1958.

```
co2 <- ts(scan('data/co2MaunaLoa.dat'), start=c(1958, 3), frequency=12)
result_co2 <- auto.fit.arima(co2, ic="aicc", plot_result=TRUE)
```

Series: serie

ARIMA(1,1,1)(0,1,1)[12]

Coefficients:

	ar1	ma1	sma1
	0.1645	-0.5210	-0.8684
s.e.	0.1048	0.0909	0.0208

sigma² = 0.09136: log likelihood = -135.78

AIC=279.57 AICc=279.63 BIC=297.23

Es necesario retirar del modelo el parámetro: ar1

Series: serie

ARIMA(0,1,1)(0,1,1)[12]

Coefficients:

	ma1	sma1
	-0.3783	-0.8684
s.e.	0.0415	0.0209

sigma^2 = 0.09155: log likelihood = -136.93
AIC=279.87 AICc=279.91 BIC=293.11

Falla la hipótesis de normalidad sobre los residuos.
El modelo es válido pero los intervalos de predicción basados en la
dist. asintótica no son válidos

MODELO FINAL

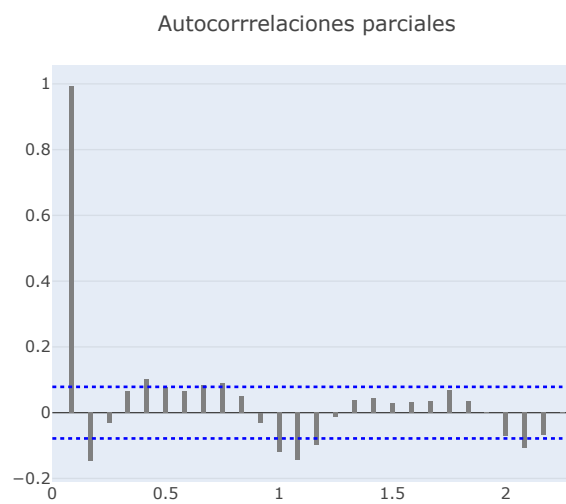
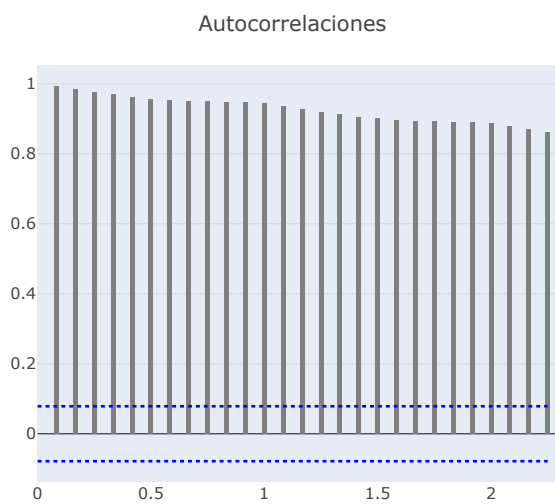
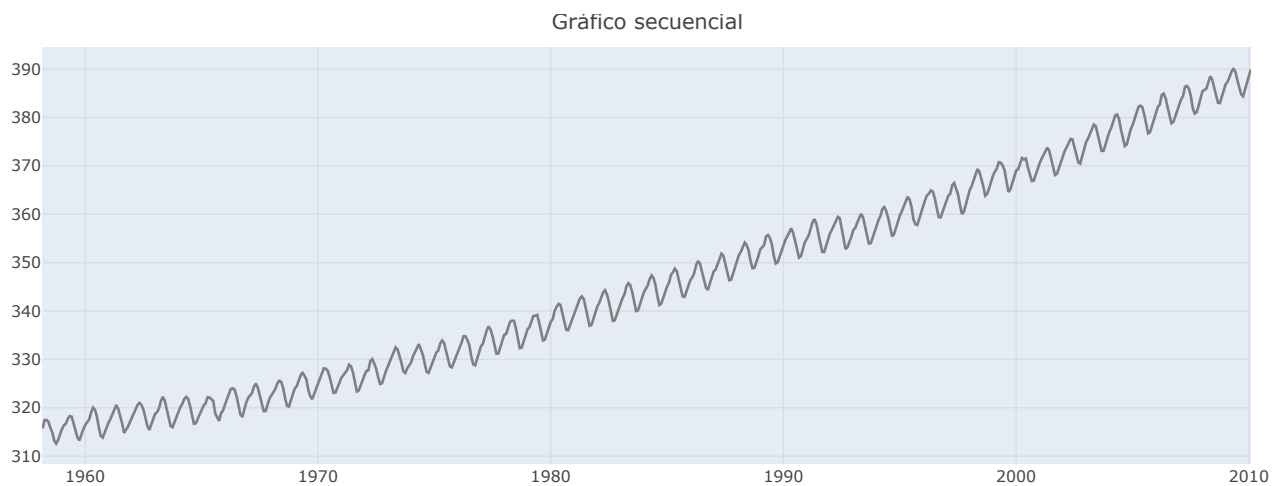
Series: serie
ARIMA(0,1,1)(0,1,1)[12]

Coefficients:

	ma1	sma1
	-0.3783	-0.8684
s.e.	0.0415	0.0209

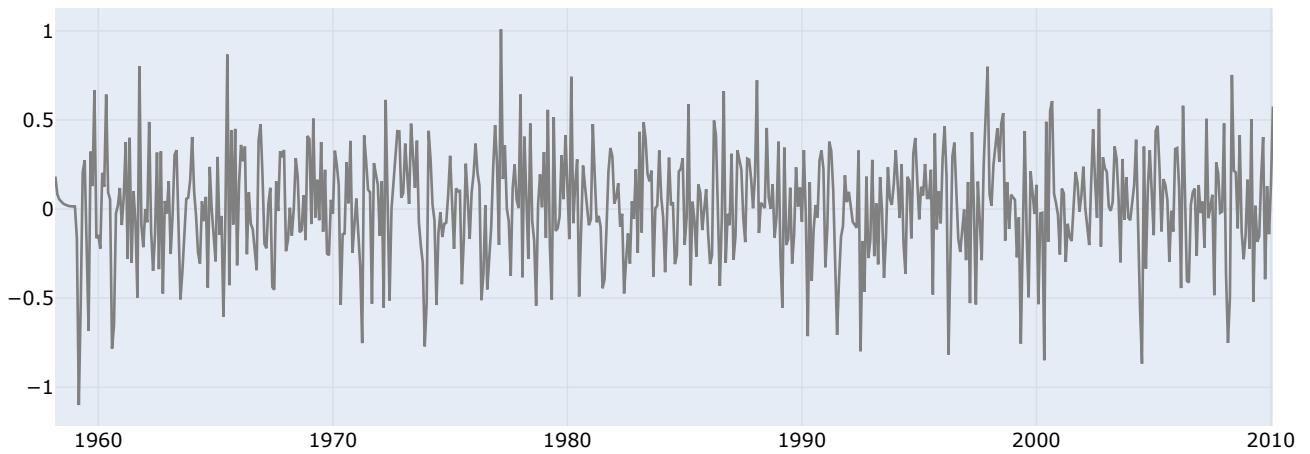
sigma^2 = 0.09155: log likelihood = -136.93
AIC=279.87 AICc=279.91 BIC=293.11

display(result_co2\$fig_serie, "serie co2", width=1000, height=800)

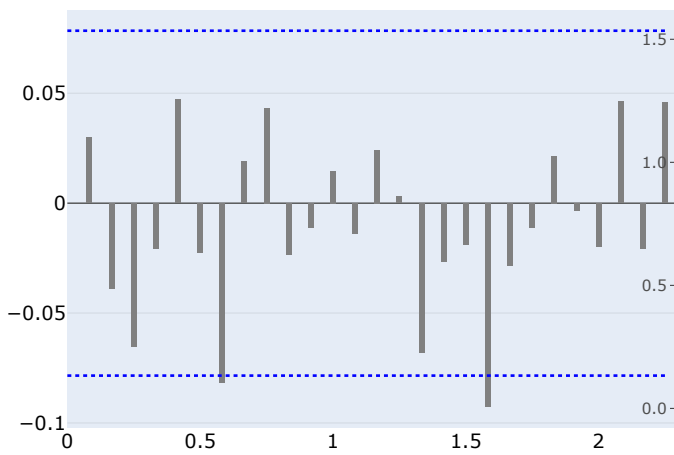


```
display(result_co2$fig_residuais, "residuals co2", width=1000, height=800)
```

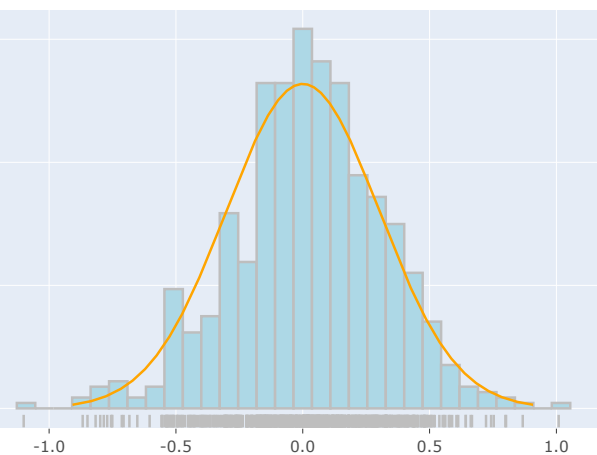
Gráfico secuencia de los residuos



Autocorrelaciones



Test de normalidad



Consideraciones:

- Para chequear la independencia de residuos se utiliza el contraste de Ljung-Box (`Box.test`). El número de retardos se escoge en base a la estacionalidad de la serie (si la hay) y la longitud de la misma (función `ljungbox_lag`).
- Para chequear la media nula de los residuos se utiliza el `t.test`.
- Para chequear la normalidad de los residuos se utilizar el test de Jarque-Bera (`jarque.bera.test`) y el de Shapiro-Wilks (`shapiro.test`).
- Los modelos considerados tendrán siempre un orden de diferenciación regular igual o inferior a 3 ($d \leq 3$) y un orden de diferenciación estacional menor o igual a 2 ($D \leq 2$).




2 Función de selección automática de múltiples variables y retardos en modelos ARIMAX (auto.fit.arima.regression en automatic_selection.R)

Descripción: Método de selección las variables regresoras y sus respectivos retardos (óptimos) para una serie de tiempo en base al método propuesto por Cryer y Chan (2008).

Devuelve: Ajuste de un modelo válido de todas las variables regresoras que se han seleccionado para modelar la variable respuesta.

```
auto.fit.arima.regression(serie, xregs, ic = c("aicc", "aic", "bic"),
                          alpha = 0.05, stationary_method='auto.arima',
                          show_info = TRUE, ndiff=0)
```

Argumentos:

- **serie [ts]:** Serie temporal que funciona como variable respuesta en el modelo de regresión dinámico sobre el que se realiza la selección de variables regresoras.
- **xregs [data.frame]:** Dataframe con las series temporales que actuarán como variables regresoras de serie. Es importante que los nombres de las columnas tengan un significado de cara a identificar las variables regresoras.
- **alpha [numeric]:** Valor entre 0 y 1 que indica el nivel de significación de los tests para chequear:
 - La significación de los parámetros de los ajustes. La validez del modelo a partir del test de independencia de residuos y el test de media nula de residuos. 
 - La selección de retardos óptimos.
 - La comprobación de tendencia de las series.
- **stationary_method [character]:** Método utilizado para chequear la estacionariedad de una serie temporal. Si `stationary_method = 'auto.arima'`, se utiliza la función `forecast::auto.arima` para ajustar un modelo ARIMA(p,d,q) y chequear si $d > 0$ (si se cumple esta condición se asume que la serie no es estacionaria). Si `stationary_method = 'adf.test'` se usa el test Dickey-Fuller (`tseries::adf.test`) para chequear la estacionariedad de una serie temporal. 
- **show_info [boolean]:** Indica si se muestra la información de la selección de variables o no.
- **ndiff [numeric]:** Parámetro interno del programa (no utilizar!) para diferenciar todas las variables cuando no se pueda ajustar un modelo válido y mantener un registro del número de diferenciaciones que se están realizando. 

Nota: No se mostrará la información del ajuste de cada modelo para cada variable regresora.

Ejemplo de uso: Logaritmo de las ventas semanales y el precio de patatas fritas *Bluebird* de Nueva Zelanda. El período de observación es de 104 semanas (desde el 20 de Septiembre de 1988 hasta el 10 de Septiembre de 2000).

```
load("data/patatas.dat")
Y <- patatas[,1]
X <- patatas[,2]
ajuste_patatas <- auto.fit.arima.regression(Y, data.frame(X))
```

Iteración 1 del algoritmo

Se ha probado con la variable X [ic=-71.4371307570267, lag=0]

Se ha añadido la variable regresora X [aicc=-71.4371307570267]

Series: serie

Regression with ARIMA(0,0,4) errors

Coefficients:

	ma1	ma2	ma3	ma4	intercept	xreg
	0	0.2884	0	0.5416	15.8559	-2.4682
s.e.	0	0.0794	0	0.1167	0.1909	0.1100

sigma^2 = 0.02728: log likelihood = 41.02

AIC=-72.05 AICc=-71.44 BIC=-58.83

No se añaden más variables

Histórico de variables añadidas al modelo

var	lag	ic
X	0	-71.4371307570267

Ejemplo de uso: Modelización de la serie de tiempo de muertes en España debido al COVID19, considerando como posibles variables regresoras:

- Los casos confirmados y curados en España.
- Los casos confirmados y muertes en Francia.
- Los casos confirmados y muertes en Inglaterra.

```
confirmed <- read.csv("data/covid-global-confirmed-bycountry.csv")
deaths <- read.csv("data/covid-global-deaths-bycountry.csv")
recovered <- read.csv("data/covid-global-recovered-bycountry.csv")
```

```
confirmed_spain <- ts(confirmed$Spain, frequency=7)
```

```
deaths_spain <- ts(deaths$Spain, frequency=7)
```

```
recovered_spain <- ts(recovered$Spain, frequency=7)
```

```
confirmed_france <- ts(confirmed$France, frequency=7)
```

```
confirmed_england <- ts(confirmed$United.Kingdom, frequency=7)
```

```
deaths_france <- ts(deaths$France, frequency=7)
```

```
deaths_england <- ts(deaths$United.Kingdom, frequency=7)
```

```
regresoras <- data.frame(confirmed_spain, recovered_spain)
```

```
ajuste <- auto.fit.arima.regression(deaths_spain, regresoras)
```

Iteración 1 del algoritmo

Se ha probado con la variable confirmed_spain [ic=5252.42532484484, lag=0]

Se ha probado con la variable recovered_spain [ic=5274.29687878351, lag=-7]

Se ha añadido la variable regresora confirmed_spain [aicc=5252.42532484484]

Series: serie

Regression with ARIMA(0,2,1)(1,0,1)[7] errors

Coefficients:

	ma1	sar1	sma1	xreg
	-0.7885	0.9106	-0.7903	0.0074
s.e.	0.0307	0.0790	0.1233	0.0011

sigma^2 = 35300: log likelihood = -2621.14

AIC=5252.27 AICc=5252.43 BIC=5272.15

Iteración 2 del algoritmo

Saltamos confirmed_spain

Se ha probado con la variable recovered_spain [ic=5165.93402575897, lag=-3]

Se ha añadido la variable regresora recovered_spain [aicc=5165.93402575897]

Series: serie

Regression with ARIMA(1,1,1)(1,0,1)[7] errors

Coefficients:

	ar1	ma1	sar1	sma1	recovered_spain	confirmed_spain
	0.9726	-0.8241	0.9166	-0.7856	0.0334	0.0072
s.e.	0.0160	0.0395	0.0627	0.1017	0.0180	0.0011

sigma^2 = 34438: log likelihood = -2575.82

AIC=5165.64 AICc=5165.93 BIC=5193.37

Iteración 1 del algoritmo



Se ha probado con la variable confirmed_spain [ic=5265.07233211036, lag=0]

Se ha probado con la variable recovered_spain [ic=5286.85279482587, lag=-7]

Se ha añadido la variable regresora confirmed_spain [aicc=5265.07233211036]

Series: serie

Regression with ARIMA(0,1,1)(1,0,1)[7] errors

Coefficients:

	ma1	sar1	sma1	xreg
--	-----	------	------	------

```

-0.7864  0.9057  -0.7825  0.0074
s.e.    0.0307  0.0832   0.1287  0.0011

```

```

sigma^2 = 35246:  log likelihood = -2627.46
AIC=5264.92  AICc=5265.07  BIC=5284.81

```

```

-----
Iteración 2 del algoritmo
-----

```

Saltamos confirmed_spain

Se ha probado con la variable recovered_spain [ic=5148.7977905626, lag=-4]

Se ha añadido la variable regresora recovered_spain [aicc=5148.7977905626]

Series: serie

Regression with ARIMA(0,1,1)(1,0,1)[7] errors

Coefficients:

```

      ma1      sar1      sma1  recovered_spain  confirmed_spain
-0.8589  0.9203  -0.7995           0.0521           0.0071
s.e.    0.0291  0.0657   0.1055           0.0170           0.0011

```

```

sigma^2 = 34219:  log likelihood = -2568.29


```

```

AIC=5148.58  AICc=5148.8  BIC=5172.33

```

No se añaden más variables

Histórico de variables añadidas al modelo			
	var	lag	ic
confirmed_spain	0	5265.07233211036	
recovered_spain	-4	5148.7977905626	

3 Funciones auxiliares

3.1 Ajuste de los coeficientes de un modelo (`fit.coefficients()` de `auto_fit_arima.R`)

Descripción: Elimina de forma incremental los coeficientes no significativos en un modelo.

Devuelve: Ajuste de un modelo donde todos sus coeficientes son significativamente distintos de cero.

```
fit.coefficients(ajuste, orders, alpha=0.05, show_info=T)
```

Argumentos:



- `ajuste` [Arima]: Ajuste de un modelo ARIMA sobre el que se deben eliminar los coeficientes no significativos.
- `orders` [list]: Objeto de tipo lista donde se especifica información sobre los órdenes regulares y estacionales del modelo. El formato es el siguiente:
 - `orders$regular = c(p, d, q)` [numeric]: Especifica los órdenes regulares.
 - `orders$seasonal = c(P, D, Q)` [numeric]: Especifica los órdenes estacionales.
 - `orders$include_constant` [boolean]: Especifica si se debe incluir constante en el ajuste.
- `alpha` [numeric]: Valor entre 0 y 1 que especifica el nivel de significación para retirar parámetros del modelo. Por defecto es 0.05%.
- `show_info` [boolean]: Indica si se debe mostrar información sobre los parámetros que se van retirando del ajuste o no. Por defecto, va mostrando esta información en consola.

3.2 Ajuste de un ARIMA vía múltiples optimizadores (`fit.model()` de `auto_fit_arima.R`)


Descripción: Ajuste de un modelo ARIMA dados sus órdenes sobre una serie temporal, manejando posibles errores de optimización y probando con otros métodos en caso de que el que viene dado por defecto provoque errores. Los optimizadores con los que prueba son, en este orden: BFGS, Nelder-Mead, CG, L-BFGS-B, SANN y Brent.

Devuelve: Modelo ARIMA para los parámetros y serie temporal dada o NA en caso de que no haya sido posible ajustar ningún modelo por problemas de optimización.

```
fit.model(serie, orders, xregs=NULL, fixed=NULL)
```


Argumentos:

- `serie` [Arima]: Serie temporal sobre la que se ajusta el modelo ARIMA.
- `orders` [list]: Objeto de tipo lista donde se especifica información sobre los órdenes regulares y estacionales del modelo. El formato es el siguiente:
 - `orders$regular = c(p, d, q)` [numeric]: Especifica los órdenes regulares.
 - `orders$seasonal = c(P, D, Q)` [numeric]: Especifica los órdenes estacionales.
 - `orders$include_mean` [boolean]: Especifica si se debe incluir la media en el ajuste ($d = 0$)

y $D = 0$). 

- `xregs [ts]`: Matriz de posibles variables regresoras.
- `fixed`: Vector de valores fijos para los coeficientes del modelo ARIMA que se quiere ajustar.

3.3 Selección del retardo óptimo (`select.optimal.lag()` de `automatic_selection`)

Descripción: Selección del retardo significativo y óptimo de dos series (asumiendo que una funciona como variable explicativa y otra como variable respuesta en un modelo de regresión con componente temporal). Esta selección se realiza siguiendo el procedimiento descrito por Cryer y Chan (2008) usando las funciones `tseries::adf.test()` o `auto.arima` para chequear estacionariedad, `seastests::isSeasonal` para chequear presencia de estacionalidad y `TSA::prewhiten()` para aplicar el preblanqueado sobre las dos series. 

Devuelve: El retardo óptimo de las dos series o NA en caso de que ningún retardo sea significativo.

```
select.optimal.lag(serie, xreg, alpha=0.05, max_lag=NA)
```

Argumentos:

- `serie [ts]`: Serie temporal que funciona como variable respuesta.
- `xreg [ts]`: Variable regresora de `serie`.
- `alpha [numeric]`: Valor entre 0 y 1 que indica el nivel de significación para aceptar o no la hipótesis nulas en los contrastes de significación, estacionariedad y estacionalidad.
- `max_lag [numeric o NA]`: Opcionalmente, se puede añadir un valor que limite el valor del retardo óptimo tal que su valor absoluto siempre sea menor que `max_lag`.

4 Predicciones puntuales a horizonte h e intervalos de confianza (forecast_model() de forecasting.R)

Descripción: A partir del ajuste de un ARIMAX realiza predicciones puntuales a horizonte h de cada variable regresora para introducirlas en las predicciones puntuales de la variable respuesta.

Devuelve: Objeto forecast con las predicciones puntuales y los intervalos de confianza.

```
forecast_model(ajuste, h, mode=c('bootstrap', 'norm'), levels=c(80, 90))
```

Arguments:

- ajuste [Arima]: Ajuste de un modelo de regresión con series temporales sobre el que se quieren hacer predicciones puntuales e intervalos de confianza.
- h [numeric]: Valor horizonte de las predicciones.
- mode [character]: Modo de realizar las predicciones: basadas en normalidad sobre los residuos (norm) o a través de bootstrap (bootstrap).
- levels [vector]: Vector numérico de los niveles a los que se quieren hacer los intervalos de predicción.

Ejemplo de uso:

```
load("data/patatas.dat")
Y <- ts(patatas[,1])
X <- ts(patatas[,2])
ajuste_patatas <- auto.fit.arima.regression(Y, data.frame(X=X))
```

Iteración 1 del algoritmo

Se ha probado con la variable X [ic=-71.4371307570267, lag=0]

Se ha añadido la variable regresora X [aicc=-71.4371307570267]

Series: serie

Regression with ARIMA(0,0,4) errors

Coefficients:

	ma1	ma2	ma3	ma4	intercept	xreg
	0	0.2884	0	0.5416	15.8559	-2.4682
s.e.	0	0.0794	0	0.1167	0.1909	0.1100

sigma^2 = 0.02728: log likelihood = 41.02

AIC=-72.05 AICc=-71.44 BIC=-58.83

No se añaden más variables

Histórico de variables añadidas al modelo

var	lag	ic
-----	-----	----

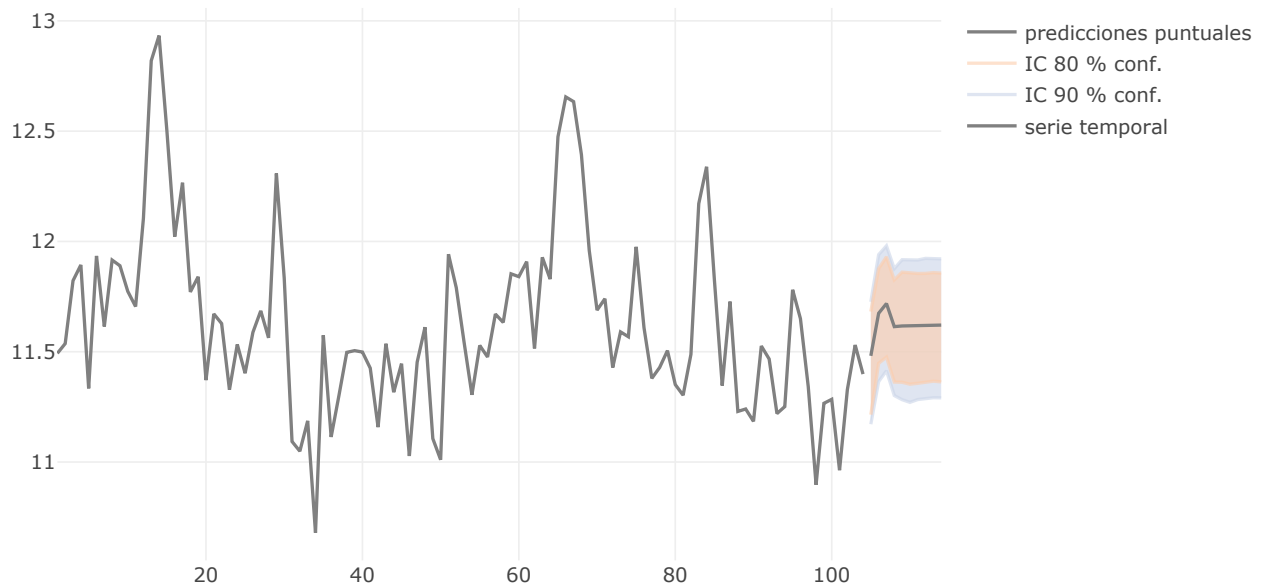
```
X 0 -71.4371307570267
```

```
ajuste_patatas$xreg <- cbind(X=ts(ajuste_patatas$xreg))
```

```
# Calculamos las predicciones puntuales
```

```
preds <- forecast_model(ajuste_patatas, h=10, mode='bootstrap')
```

```
display(plot_forecast(preds), name='preds_patatas')
```



5 Comprobación con ejemplos

5.1 Evolución de la gripe en Cataluña

```
# Carga de datos
```

```
cataluna <- read.csv("data/evolucion_gripe_covid.csv")  
str(cataluna)
```

```
'data.frame': 60 obs. of 39 variables:  
 $ fecha          : chr  "2020-40" "2020-41" "2020-42" "2020-43" ...  
 $ sdgripal       : int   337 353 341 417 394 325 294 254 216 197 ...  
 $ sarscov2       : int   71 133 133 218 220 169 161 135 87 60 ...  
 $ sdgripal.edad04 : int   52 35 40 65 48 49 43 33 32 28 ...  
 $ sarscov2.edad04 : int    6 5 7 13 7 8 14 3 2 1 ...  
 $ sdgripal.edad1544 : int  89 115 113 138 129 88 77 61 60 49 ...  
 $ sarscov2.edad1544 : int  21 47 50 82 81 51 39 33 26 11 ...  
 $ sdgripal.edad4564 : int  37 66 48 80 75 72 42 44 40 49 ...  
 $ sarscov2.edad4564 : int  15 30 22 51 55 43 25 23 19 22 ...  
 $ sdgripal.edad514 : int 139 103 108 105 108 87 78 51 44 54 ...  
 $ sarscov2.edad514 : int  25 38 39 55 57 48 46 25 16 20 ...  
 $ sdgripal.edad65 : int  20 34 32 29 34 29 54 65 40 17 ...  
 $ sarscov2.edad65 : int   4 13 15 17 20 19 37 51 24 6 ...  
 $ pob04          : int 5450 5450 5450 5450 5441 5438 5443 5436 5425 5426 ...  
 $ pob514         : int 14270 14270 14270 14270 14262 14268 14266 14248 14242 14235 ...  
 $ pob1544        : int 18428 18428 18428 18428 18431 18402 18386 18334 18328 18311 ...  
 $ pob4564        : int 13825 13825 13825 13825 13841 13848 13854 13816 13817 13821 ...  
 $ pob65          : int 8991 8991 8991 8991 8998 8995 8997 8982 8988 8987 ...  
 $ pob            : int 60964 60964 60964 60964 60973 60951 60946 60816 60800 60780 ...  
 $ sdgripal.BARCELONA : int  95 93 92 123 125 94 86 59 61 58 ...  
 $ sarscov2.BARCELONA : int  14 27 39 50 51 40 36 18 15 10 ...  
 $ sdgripal.CANTALUNYA : int  37 49 50 56 35 34 31 23 21 26 ...  
 $ sarscov2.CANTALUNYA : int  11 21 20 27 26 21 23 19 13 12 ...  
 $ sdgripal.GIRONA : int  16 27 30 36 28 24 17 12 23 9 ...  
 $ sarscov2.GIRONA : int   4 11 12 25 21 16 14 6 8 3 ...  
 $ sdgripal.LLEIDA : int  27 27 37 24 31 32 26 16 18 15 ...  
 $ sarscov2.LLEIDA : int   5 11 13 10 18 18 14 8 11 9 ...  
 $ sdgripal.METR_NORD : int  24 29 24 34 29 29 27 31 20 12 ...  
 $ sarscov2.METR_NORD : int   8 16 10 26 19 15 17 16 7 5 ...  
 $ sdgripal.METR_SUD : int  20 23 14 24 32 22 20 8 9 13 ...  
 $ sarscov2.METR_SUD : int   3 9 5 20 15 11 5 1 0 6 ...  
 $ sdgripal.PIRINEU : int  11 16 14 10 10 12 39 72 32 29 ...  
 $ sarscov2.PIRINEU : int   1 8 9 3 7 6 32 57 26 7 ...  
 $ sdgripal.TARRAGONA : int  35 48 44 67 62 44 28 18 22 18 ...  
 $ sarscov2.TARRAGONA : int  10 11 12 29 34 18 11 8 7 3 ...  
 $ sdgripal.TERRES_EBRE: int  21 14 15 11 8 16 4 4 4 5 ...  
 $ sarscov2.TERRES_EBRE: int   7 9 9 7 5 9 3 0 0 1 ...  
 $ sdgripal.VALLES : int  51 27 21 32 34 18 16 10 6 12 ...  
 $ sarscov2.VALLES : int   8 10 4 21 24 15 6 2 0 4 ...
```

El dataset `evolucion_gripe_covid.csv` contiene información sobre la evolución de la gripe y el COVID19 en las distintas áreas sanitarias de Cataluña y en toda la comunidad a lo largo del tiempo. Cada dato recogido representa el número de casos confirmados (de gripe y COVID19) en una semana (desde la 40ª semana de 2020 hasta la 46ª semana de 2021).

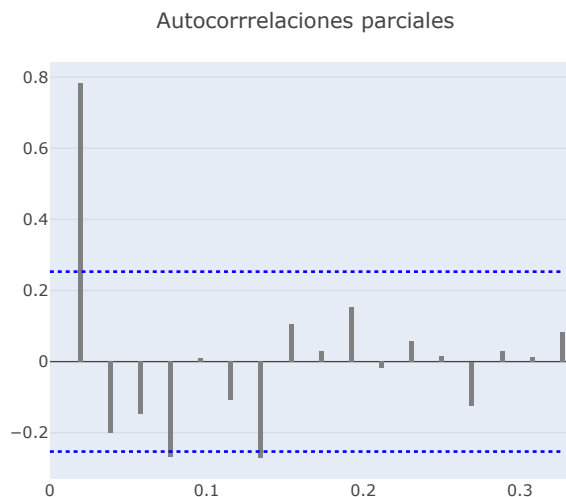
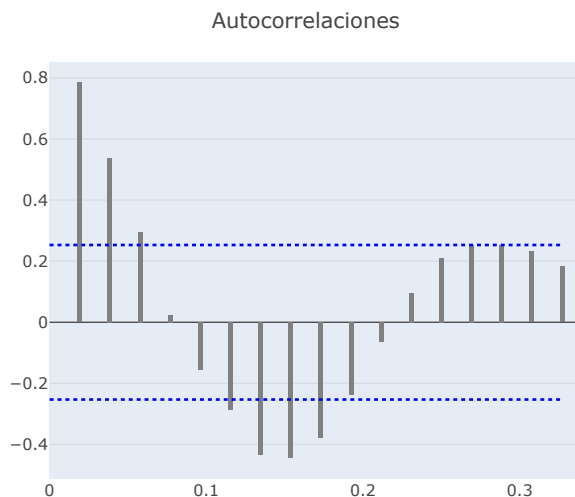
Vamos a intentar modelizar la evolución de la gripe con un ARIMA a través de los siguientes métodos:

- Usando la función `auto.arima` y ajustando los coeficientes para obtener un ajuste válido.
- Usando la función `auto.fit.arima` que realiza todo el proceso.

```
# Los datos ya están ordenados temporalmente
gripe <- ts(cataluna$sdgripal, start=c(2020, 40), frequency=52)
```

Analizamos el gráfico secuencial y la fas y fap muestral:

```
display(result_gripe$fig_serie, "serie gripe", width=1000, height=800)
```



A continuación, usamos la función `auto.arima`:

```
ajuste <- auto.arima(gripe, stepwise=FALSE, approximation=FALSE, trace=TRUE)
```

ARIMA(0,0,0)	with zero mean	: 852.0233
ARIMA(0,0,0)	with non-zero mean	: 751.2141
ARIMA(0,0,1)	with zero mean	: 788.6201
ARIMA(0,0,1)	with non-zero mean	: 707.8
ARIMA(0,0,2)	with zero mean	: 754.1405

ARIMA(0,0,2)	with non-zero mean	: 695.2859
ARIMA(0,0,3)	with zero mean	: 730.1785
ARIMA(0,0,3)	with non-zero mean	: 678.1347
ARIMA(0,0,4)	with zero mean	: 711.7497
ARIMA(0,0,4)	with non-zero mean	: 676.3749
ARIMA(0,0,5)	with zero mean	: 706.3197
ARIMA(0,0,5)	with non-zero mean	: 678.4754
ARIMA(1,0,0)	with zero mean	: 681.8079
ARIMA(1,0,0)	with non-zero mean	: 679.3566
ARIMA(1,0,1)	with zero mean	: 681.9605
ARIMA(1,0,1)	with non-zero mean	: 678.4626
ARIMA(1,0,2)	with zero mean	: 682.7
ARIMA(1,0,2)	with non-zero mean	: 678.4742
ARIMA(1,0,3)	with zero mean	: 683.6291
ARIMA(1,0,3)	with non-zero mean	: 675.201
ARIMA(1,0,4)	with zero mean	: 682.3024
ARIMA(1,0,4)	with non-zero mean	: 677.6556
ARIMA(2,0,0)	with zero mean	: 681.4824
ARIMA(2,0,0)	with non-zero mean	: 676.982
ARIMA(2,0,1)	with zero mean	: 683.4355
ARIMA(2,0,1)	with non-zero mean	: 674.5936
ARIMA(2,0,2)	with zero mean	: 686.3982
ARIMA(2,0,2)	with non-zero mean	: 678.6417
ARIMA(2,0,3)	with zero mean	: 684.4164
ARIMA(2,0,3)	with non-zero mean	: 677.6966
ARIMA(3,0,0)	with zero mean	: 683.1577
ARIMA(3,0,0)	with non-zero mean	: 676.741
ARIMA(3,0,1)	with zero mean	: 685.9773
ARIMA(3,0,1)	with non-zero mean	: 677.0667
ARIMA(3,0,2)	with zero mean	: Inf
ARIMA(3,0,2)	with non-zero mean	: 679.2844
ARIMA(4,0,0)	with zero mean	: 685.5207
ARIMA(4,0,0)	with non-zero mean	: 677.8352
ARIMA(4,0,1)	with zero mean	: 686.9957
ARIMA(4,0,1)	with non-zero mean	: 679.5731
ARIMA(5,0,0)	with zero mean	: 684.1011
ARIMA(5,0,0)	with non-zero mean	: 679.7597

Best model: ARIMA(2,0,1) with non-zero mean

Y comprobamos que el mejor modelo (siguiendo el AICc) es un ARIMA(2, 0, 1) con media.

ajuste

Series: gripe

ARIMA(2,0,1) with non-zero mean

Coefficients:

	ar1	ar2	ma1	mean
	1.7693	-0.8833	-0.7882	251.3508
s.e.	0.1032	0.0946	0.1667	18.7800

sigma^2 = 3824: log likelihood = -331.74

AIC=673.48 AICc=674.59 BIC=683.95

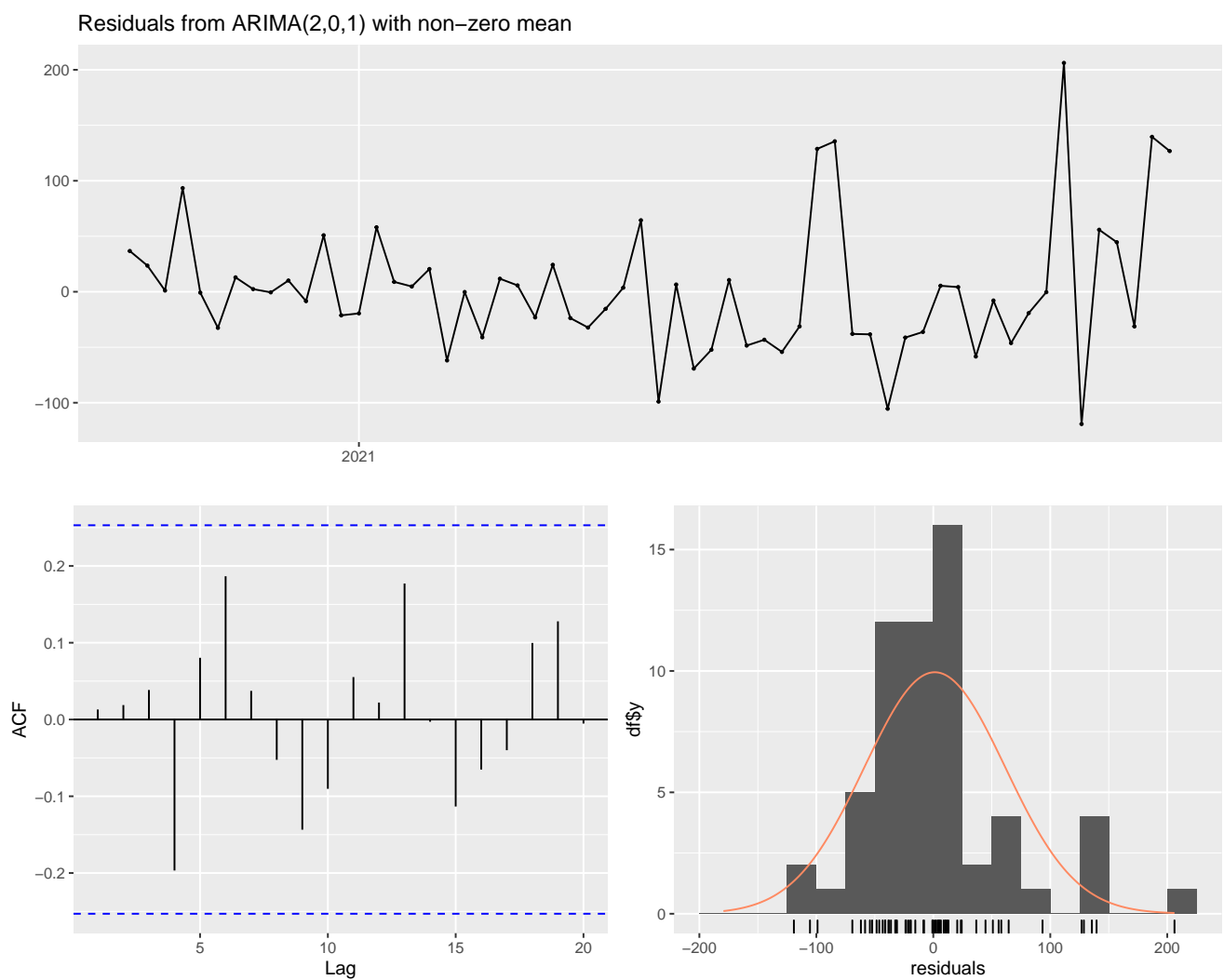
A continuación, comprobamos qué parámetros *no* son significativos:

```
alpha <- 0.05; stat <- qnorm(1-0.05/2)
abs(ajuste$coef) < stat*sqrt(diag(ajuste$var.coef))
```

ar1	ar2	ma1	intercept
FALSE	FALSE	FALSE	FALSE

En este caso, **todos** los parámetros son significativos y por tanto se trata de un ajuste válido. Finalmente, realizamos el análisis de residuos para chequear las hipótesis de independencia y media nula.

```
checkresiduals(ajuste)
```



Ljung-Box test

data: Residuals from ARIMA(2,0,1) with non-zero mean
Q* = 8.2096, df = 8, p-value = 0.4133

Model df: 4. Total lags used: 12

```
t.test(ajuste$residuals, mu=0)
```

One Sample t-test

```
data: ajuste$residuals
t = 0.1638, df = 59, p-value = 0.8704
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 -14.28613 16.83357
sample estimates:
mean of x
 1.273721
```

El test de independencia de Ljung-Box y el test de media nula nos dicen que los residuos sí son independientes y tienen media cero, por tanto se puede considerar que el ajuste es válido para modelizar la evolución de la gripe.

El objetivo de la función `auto.fit.arima` es realizar todo este proceso de forma automática. El resultado que nos devuelve debe ser el mismo que el que hemos obtenido haciendo los cálculos paso a paso:

```
ajuste <- auto.fit.arima(gripe)
```

```
-----
Series: serie
ARIMA(2,0,1) with non-zero mean

Coefficients:
      ar1      ar2      ma1      mean
    1.7693 -0.8833 -0.7882 251.3508
s.e.  0.1032  0.0946  0.1667  18.7800

sigma^2 = 3824: log likelihood = -331.74
AIC=673.48 AICc=674.59 BIC=683.95
-----
```

Falla la hipótesis de normalidad sobre los residuos.
El modelo es válido pero los intervalos de predicción basados en la dist. asintótica no son válidos

```
-----
|                                     MODELO FINAL                                     |
-----
Series: serie
ARIMA(2,0,1) with non-zero mean

Coefficients:
      ar1      ar2      ma1      mean
    1.7693 -0.8833 -0.7882 251.3508
s.e.  0.1032  0.0946  0.1667  18.7800

sigma^2 = 3824: log likelihood = -331.74
AIC=673.48 AICc=674.59 BIC=683.95
```

Adicionalmente, la función `auto.fit.arima` nos avisa de que los **residuos** no siguen una distribución normal, por tanto tendremos que tener cuidado al hacer predicciones sobre la serie.



Referencias

Cryer, Jonathan D y Kung-Sik Chan (2008). *Time series analysis: with applications in R*. Vol. 2. Springer.