Series Temporales Avanzadas - Modelos Heterocedásticos

Una aplicación en la estimación de rendimientos financieros para acciones, utilizando el software estadístico R.

Jorge Andrés Paguay Ortiz

Universidad de Granada

Departamento de Estadística e Investigación Operativa

Granada, España

2017

Series Temporales Avanzadas - Modelos Heterocedásticos

Una aplicación en la estimación de rendimientos financieros para acciones, utilizando el software estadístico R.

Jorge Andrés Paguay Ortiz

Trabajo presentado como requisito parcial para optar al título de:

Máster en Estadística Aplicada

Director:

Dr. Francisco Javier Alonso Morales

Línea de Investigación:

Análisis de series de tiempo, Aplicaciones a riesgos financieros, Inversiones

Universidad de Granada

Departamento de Estadística e Investigación Operativa

Granada, España

2017

Dedicatoria

A mi familia: Adriel y Yannira, por inspirarme a no desmayar.

A mis padres, por el regalo de la vida.

A mi hermano, por su entusiasmo.

A la UGR, por permitirme cursar este enriquecedor programa de estudios.

Agradecimientos

Quiero dar gracias a todas las personas que estuvieron pendientes de mi progreso en esta etapa de mi vida.

Quiero agradecer a los profesores del Máster en Estadística Aplicada de la UGR, por su dedicación a esta rama de la ciencia. En especial a Javier Alonso y Andrés González.

CONTENIDO

1. Conceptos teóricos.

En el presente capítulo se explican los conceptos teóricos necesarios para comprender la estimación de series de tiempo, centrándose en la estimación de rendimientos financieros.

1.1. Series de Tiempo y Procesos Estocásticos

Una serie de tiempo es una secuencia de valores ordenados cronológicamente, que representan un concepto: ventas, ingresos, egresos, etc. Además, sabemos que una secuencia de variables aleatorias es un proceso estocástico y sirve como modelo para observar una serie de tiempo, entonces definiremos una serie de tiempo como un proceso estocástico.

$$\{Y_t : t \in Z\}$$

Por otro lado, sabemos que gran parte de la información de este proceso se puede describir en términos de medias, varianzas y covarianzas, por tanto concentraremos nuestra atención en estos momentos.

1.1.1. Función de Medias

Para una serie de tiempo $\{Y_t\}$, la función de medias está definida por:

$$\mu_t = E(Y_t);\ t \in Z$$

μ_t es el valor esperado del proceso en el tiempo t, por tanto este puede cambiar en cada momento del tiempo.

1.1.2. Función de Auto-covarianzas

Para una serie de tiempo $\{Y_t\}$, la función de auto-covarianzas está definida por:

$$\gamma_{t,s} = Cov(Y_t, Y_s);\ t, s \in Z$$

Donde:

$$Cov(Y_t, Y_s) = E[(Y_t - \mu_t)(Y_s - \mu_s)]$$

Es importante, tener clara la siguiente propiedad:

$$\gamma_{t,t} = Var(Y_t)$$

$$\gamma_{t,s} = \gamma_{s,t}$$

$$|\gamma_{t,s}| \leq \sqrt{\gamma_{t,t}\gamma_{s,s}}$$

1.1.3. Función de Auto-correlaciones

Para una serie de tiempo $\{Y_t\}$, la función de auto-correlaciones está definida por:

$$\rho_{t,s} = Corr(Y_t, Y_s);\ t \in Z$$

Donde:

$$Corr(Y_t, Y_s) = \frac{Cov(Y_t, Y_s)}{\sqrt{Var(Y_t)Var(Y_s)}} = \frac{\gamma_{t,s}}{\sqrt{\gamma_{t,t}\gamma_{s,s}}}$$

Es importante, tener clara la siguiente propiedad:

$$\rho_{t,t} = 1$$

$$\rho_{t,s} = \rho_{s,t}$$

$$|\rho_{t,s}| \leq 1$$

1.1.4. Función de Auto-correlaciones Parciales

Para una serie de tiempo normalmente distribuida $\{Y_t\}$, la función de auto-correlaciones parciales está definida por:

$$\phi_{kk} = Corr(Y_t, Y_{t-k} | Y_{t-1}, Y_{t-2}, \dots, Y_{t-k+1});\ t \in Z;\ k \geq 0$$

ϕ_{kk}: es la correlación en la distribución bivariada de Y_t y Y_{t-k} condicionada en $Y_{t-1}, Y_{t-2}, \dots, Y_{t-k+1}$.

Es importante, tener claro lo siguiente:

$$\phi_{kk} = 1; k = 1$$

1.1.5. Componentes de una serie de tiempo

De acuerdo al modelo de descomposición clásico, la serie de tiempo está formada por las siguientes componentes: tendencia, estacionalidad y aleatoriedad, mismas que pueden unirse de forma aditiva o multiplicativa. A continuación una breve descripción de las mismas:

Tendencia

Podemos decir que la tendencia constituye los aumentos o disminuciones que sufre la media de la serie a lo largo del tiempo. Diremos que:

- La serie tiene tendencia creciente, si los valores aumentan a medida que el tiempo aumenta.
- La serie tiene tendencia decreciente, si los valores disminuyen a medida que el tiempo aumenta.
- La serie no tiene tendencia, si los valores no tienen variaciones significativas a medida que el tiempo aumenta.

Estacionalidad

Podemos decir que la estacionalidad constituye las variaciones que la serie sufre con cierta periodicidad, es común que estas periodicidades sean trimestrales, semestrales o anuales.

Aleatoriedad

Esta componente no describe ninguna característica de la serie, corresponde a los factores externos y su ocurrencia se atribuye al azar.

1.2. Procesos Estocásticos Estacionarios

En el presente trabajo, asumiremos que un proceso estocástico es una familia de variables aleatorias, denotada por:

$$\{Y_t : t \in Z\}$$

1.2.1. Procesos Estocásticos Estrictamente Estacionarios

Un proceso estocástico $\{Y_t : t \in Z\}$, es estricta o fuertemente estacionario si:

$$ley\left(Y_{t_1+h}, \dots, Y_{t_k+h}\right) = ley\left(Y_{t_1}, \dots, Y_{t_k}\right);\ t, k, h \in Z$$

1.2.2. Procesos Estocásticos Débilmente Estacionarios

Un proceso estocástico $\{Y_t : t \in Z\}$, es débilmente estacionario si:

1. $\{Y_t : t \in Z\}$ es real y de segundo orden.
2. $\mu_t = E(Y_t);\ t \in Z$ es constante.
3. $Cov(Y_t, Y_s) = Cov(Y_{t+h}, Y_{s+h});\ t, k, h \in Z$

Al cual se llamará estacionario en adelante.

1.3. Procesos Lineales

En series de tiempo es habitual ver, que la información presente depende de la información pasada y esta idea permite buscar una combinación lineal que explique la relación existente entre el presente y el pasado. Esta idea se puede representar mediante Procesos Lineales, a continuación su definición:

Un proceso estocástico $\{Y_t : t \in Z\}$ Centrado, de segundo orden. Es un Proceso Lineal, si se puede expresar como:

$$Y_t = \sum_{j=0}^{\infty} \Psi_j u_{t-j}\ ; t \in Z$$

Donde:

$\Psi_0 = 1$

$\sum_{j=0}^{\infty}\left|\Psi_j\right|^2 < \infty;\ \Psi_j \in R$

$\{u_t : t \in Z\}$ Es ruido blanco de varianza σ^2

1.3.1. Proceso Lineal Inversible

Un proceso lineal $\{Y_t : t \in Z\}$ es inversible si:

$$u_t = \sum_{j=0}^{\infty} \pi_j' Y_{t-j}\ ;t \in Z$$

Donde:

$\pi_j' = 1$

$\sum_{j=0}^{\infty}\left|\pi_j'\right|^2 < \infty;\ \pi_j' \in R$

1.3.2. Procesos Autoregresivos, AR(p)

Llamaremos Proceso Autoregresivo de orden (p), AR(p). A todo proceso lineal estacionario $\{Y_t : t \in Z\}$, que satisface:

$$Y_t = \sum_{j=1}^{p} \phi_j Y_{t-j} + u_t; t \in \mathrm{Z}$$

Donde:

$\phi_p \neq 0$

$\{u_t : t \in Z\}$ es un Ruido Blanco[1].

[1] Proceso estacionario de variables aleatorias idénticamente distribuidas con media cero, varianza sigma cuadrado.

$u_t \perp u_{t-1}; t \in Z$

1.3.3. Procesos Medias Móviles, MA(q)

Llamaremos Proceso Medias Móviles de orden (q), MA(q). A todo proceso lineal estacionario $\{Y_t: t \in Z\}$, tal que:

$$Y_t = u_t - \sum_{j=1}^{p} \theta_j u_{t-j}; t \in \mathrm{Z}$$

Donde:

$\theta_p \neq 0$

$\{u_t: t \in Z\}$ es un Ruido Blanco.

$u_t \perp u_{t-1}; t \in Z$

1.3.4. Procesos Medias Móviles - Autorregresivos, ARMA(p,q)

Llamaremos Procesos Medias Móviles - Autoregresivos de orden (p,q), ARMA(p,q) a todo proceso lineal estacionario $\{Y_t: t \in Z\}$, que satisface:

$$Y_t - \sum_{j=1}^{p} \phi_j Y_{t-j} = u_t - \sum_{j=1}^{q} \theta_j u_{t-j}$$

Donde:

$\phi_p \neq 0$

$\theta_p \neq 0$

$\{u_t: t \in Z\}$ es un Ruido Blanco.

$\Phi \ y \ \Theta$ los polinomios característicos[2] no tienen raíces de módulo 1.

1.3.5. Procesos Medias Móviles - Autorregresivos Integrados, ARIMA (p,d,q)

Llamaremos Procesos Medias Móviles - Autorregresivos Integrados de orden (p,d,q), ARIMA(p,d,q) a todo proceso lineal no estacionario $\{Y_t : t \in Z\}$, que satisface:

$$\Phi(B)(1-B)^d Y_t = \Theta(B)u_t ; t \in Z$$

Donde:

$\{u_t : t \in Z\}$ es un Ruido Blanco.

$\Phi \ y \ \Theta$ los polinomios característicos tienen sus raíces fuera del círculo unidad.

1.3.6. Procesos Heterocedásticos Condicionados Autorregresivos ARCH(r)

Llamaremos Procesos Heterocedásticos Autorregresivos Condicionados de orden (r), ARCH(r) a todo proceso lineal estacionario $\{Y_t : t \in Z\}$, que satisface:

$$Y_t = \sqrt{h_t u_t}$$

$$h_t = \alpha_0 + \alpha_1 Y^2{}_{t-1} + \cdots + \alpha_r Y^2{}_{t-r}$$

h_t es la varianza condicionada de Y_t dado $\{Y_s ; s < t\}$.

Donde:

$\{u_t : t \in Z\}$ es un Ruido Blanco de variables independientes e idénticamente distribuidas.

$\alpha_0 > 0$

2

$\alpha_i \geq 0; i > 0$

1.3.7. Procesos Heteroscedásticos Condicionados Autorregresivos Generalizados GARCH(r,s)

Llamaremos Procesos Heterocedásticos Autorregresivos Condicionados Generalizado de orden (r,s), GARCH(r,s) a todo proceso lineal estacionario $\{Y_t : t \in Z\}$, que satisface:

$$Y_t = \sqrt{h_t u_t}$$

$$h_t = \alpha_0 + \sum_{j=1}^{r} \alpha_i Y_{t-j}^2 + \sum_{j=1}^{s} \beta_j h_{t-j}$$

h_t es la varianza condicionada de Y_t dado $\{Y_r; r, s < t\}$.

Donde:

$\{u_t : t \in Z\}$ es un Ruido Blanco de variables independientes e idénticamente distribuidas.

$\alpha_0 > 0$

$\alpha_i \geq 0; i > 0$

$\beta_i \geq 0; i > 0$

$\sum_{j=1}^{r} \alpha_j + \sum_{j=1}^{s} \beta_j < 1$

2. Utilización de R para la estimación de series de tiempo.

Para realizar los cálculos necesarios en el presente trabajo, se ha elegido el Software Estadístico de uso libre R. Este software permite manejar la información de una serie de tiempo fácilmente, así como también se puede utilizar paquetes, funciones y código propio para estimar el mejor modelo que se ajuste a la serie de tiempo en estudio.

En adelante se asumirá que el lector tiene conocimientos básicos sobre la instalación y uso del Software Estadístico R (Si el lector no está familiarizado con R, se recomienda seguir el siguiente curso gratuito: https://www.coursera.org/learn/data-scientists-tools , mismos que le permitirá familiarizarse con R).

En este capítulo se utilizará, la información de cotización diaria de las acciones de Amazon, comprendida en el periodo del 04/Ago/2016 hasta el 01/Ago/2017. Esta información ha sido obtenida de la plataforma: Google Finance y alojada en Github en la siguiente dirección:

https://raw.githubusercontent.com/jorgepaguay86/EstadisticaAplicada_TFM_UGR/master/SerieDeTiempo_Amazon_Ago2016Ago2017.csv

Para facilitar la reproducción de los Códigos de Ejemplo, planteados en el presente capítulo.

2.1. Lectura de datos

En R podemos leer archivos .csv, utilizaremos la función read.csv() y visualizar el resultado obtenido con función View(). A continuación el detalle de estas funciones:

```
read.csv(file, header = TRUE, sep = ",", quote = "\"",
         dec = ".", fill = TRUE, comment.char = "", ...)
```

Argumentos[3]:

- file.- El nombre del archivo que contiene los datos, también puede ser una URL completa.
- header.- Indicador T/F que define si el archivo contiene el nombre de los campos en la primera línea del archivo o no.
- sep.- El carácter separador de campos.
- quote.- El conjunto de caracteres que citan.
- dec.- El carácter utilizado en el archivo para los puntos decimales.
- fill.- Indicador T/F en caso de que las filas tengan una longitud desigual, se añaden implícitamente campos en blanco.
- comment.char.- Un vector de caracteres de longitud uno que contiene un solo carácter o una cadena vacía.

```
View(x, title)
```

Argumentos:

- x.- Objeto R con valores distintos de cero, que puede ser representado en filas y columnas.
- title.- Título para la ventana del visor.

Código de Ejemplo:

```
ejemplo_data <-
read.csv('https://raw.githubusercontent.com/jorgepaguay86/EstadisticaAplicada_TFM_UGR/master/SerieDeTiempo_Amazon_Ago2016Ago2017.csv')
View(ejemplo_data,'SerieDeTiempo_Amazon_Ago2016Ago2017')
```

[3] Cada vez que revisemos los argumentos de una función. Estos corresponderán a la descripción tomada de la ayuda de la función en R e interpretada al español por el autor del presente trabajo.

2.2. Transformación de los Datos en un Objeto Serie de Tiempo

Una vez que hemos leído la información en R. Es recomendable transformar esta información en un objeto R, esto contribuye a un mejor rendimiento del software en los cálculos que se van a realizar y evita errores al utilizar funciones relacionadas con las series de tiempo.

Para realizar la transformación utilizamos la función ts(). A continuación el detalle de esta función:

```
ts(data = NA, start = 1, end = numeric(), frequency = 1,
     deltat = 1, ts.eps = getOption("ts.eps"), class = ,
     names = )
```

Argumentos:

- data.- Un vector o matriz de los valores observados de las series temporales.
- start.- El tiempo de la primera observación.
- end.- El tiempo de la última observación.
- frequency.- El número de observaciones por unidad de tiempo.
- deltat.- La fracción del período de muestreo entre observaciones sucesivas.
- ts.eps.- Tolerancia de comparación de series temporales.
- class.- Clase a ser dada al resultado.
- names.- Un vector de caracteres de nombres para la serie en una serie múltiple.

Código de Ejemplo:

```
ejemplo_data_ts <- ts(ejemplo_data$Close)

View(ejemplo_data_ts)
```

2.3. Estadística descriptiva de la Serie de Tiempo

Como primer acercamiento a la estructura de la serie de tiempo, se grafican y calculan sus principales estadísticos. Esto permitirá tener una idea del comportamiento de la serie. Para lo cual se debe hacer uso de las siguientes funciones: plot.ts() y summary(). A continuación el detalle de estas funciones:

```
plot.ts(x, y = NULL, plot.type = c("multiple", "single"),
        xy.labels, xy.lines, panel = lines axes = TRUE, …)
```

Argumentos:

- x,y.- Objetos de series temporales, por lo general heredando de la clase "ts".
- plot,type.- Para las series temporales multivariadas.
- xy.labels.- Lógico, indicando si las etiquetas de texto () deberían usarse para un gráfico x-y.
- xy.lines.- Lógico, indicando si se deben dibujar líneas para un gráfico x-y.
- panel.- Una función (x, col, bg, pch, tipo, etc) que da la acción a realizar en cada panel de la pantalla para plot.type = "multiple".
- axes.- Lógico, indicando si deben dibujarse los ejes xy yy.

```
summary(object, ...)
```

Argumentos:

- object.- Un objeto para el que se desea un resumen.

Código de Ejemplo:

```
plot.ts(ejemplo_data_ts,main = 'Serie de Tiempo Amazon Ago2016Ago2017', xlab = 'Tiempo', ylab = 'Precio de Cierre')

summary(ejemplo_data_ts)
```

2.4. Calculo de Retornos, Retornos al cuadrado y Retornos absolutos

Para calcular los retornos de una Serie de Tiempo de acciones, utilizaremos la siguiente expresión:

- Retorno[4].-

```
ret = diff(log(ts⁵))*100
```

- Retorno al Cuadrado.-

```
ret_cua = ret^2
```

- Retorno Absoluto.-

```
ret_abs = abs(ret)
```

Código de Ejemplo:

```
ret = diff(log(ejemplo_data_ts))*100

ret_cua = ret^2

ret_abs = abs(ret)
```

2.5. Pruebas para detectar efectos hetorecedásticos

Para detectar la presencia de efectos hetorocedásticos en la serie de residuos asociada a un Modelo de Media vamos a utilizar las pruebas: Ljung-Box y Multiplicadores de Lagrange.

2.5.1. Prueba Ljung-Box, para independencia

Consiste en probar si existe dependencia lineal entre los rezagos de la serie de tiempo. Para esto utilizamos la prueba de hipótesis Ljung-Box.

$$H_0: \rho_1 = \rho_2 = \cdots = \rho_h = 0$$

---

[4] Este cálculo se basa en la definición de rendimiento: $r_t = \log p_t - \log p_{t-1}$.

[5] Hace referencia al objeto serie de tiempo que guarda los precios de la acción en análisis.

$H_1: \exists \rho_i \neq 0$

La prueba de hipótesis Ljung-Box está implementada en R con la función: box.test(). A continuación los detalles de esta función:

```
Box.test(x, lag = 1, type = c("Box-Pierce", "Ljung-Box"),
fitdf = 0)
```

Argumentos:

- x.- Un vector numérico o series temporales univariadas.
- lag.- La estadística se basará en coeficientes de autocorrelación de retardo.
- type.- Prueba a realizar: se utiliza la coincidencia parcial.
- fitdf.- Número de grados de libertad que se restará si x es una serie de residuos.

Código de Ejemplo:

```
install.packages('xts')

library('xts')

Box.test(coredata(ret),type = 'Ljung-Box', lag =1)
```

2.5.2. Prueba Multiplicador de Lagrange, para efectos ARCH, GARCH

Consiste en probar si es apropiado modelar la varianza de un modelo de media, utilizando un modelo ARCH o GARCH. Para esto utilizamos la prueba de hipótesis LM (Multiplicador de Lagrange).

$H_0: \alpha_1 = \alpha_2 = \cdots = \alpha_h = 0$

$H_1: \exists \alpha_i \neq 0$

La prueba de hipótesis LM está implementada en R con la función: Arch.Test(). A continuación los detalles de esta función:

```
ArchTest (x, lags=12, demean = FALSE)
```

Argumentos:

- x.- Vector numérico
- lags.- Número entero positivo, que representa los retrasos.
- demean.- Lógico, elimine la media antes de calcular la estadística de prueba.

Código de Ejemplo:

```
install.packages('FinTS')

library('FinTS')

ArchTest(ret, lag=1)
```

2.6. Función de Auto-correlaciones

Para calcular las auto-correlaciones de una serie de tiempo, utilizaremos la función: acf(),

A continuación los detalles de esta función:

```
acf(x, lag.max = NULL,

    type = c("correlation", "covariance", "partial"),

    plot = TRUE, na.action = na.fail, demean = TRUE, ...)
```

Argumentos:

- x.- Un objeto numérico de serie temporal univariante o multivariado.
- lag.max.- Retardo máximo en la que calcular el acf.
- type.- Cadena de caracteres que da el tipo de acf a calcular.
- plot.- Lógico F/T, dibuja el acf.
- na.action.- Función a ser llamada para manejar valores perdidos.
- demean.- Lógico. ¿Deben las covarianzas ser sobre los medios de la muestra?

Código de Ejemplo:

```
acf(ret)
```

Se utiliza el mismo comando para el cálculo de auto-correlaciones en modelos de media y varianza. Es el usuario quien debe elegir la serie según corresponda correctamente.

2.7. Función de Auto-correlaciones Parciales

Para calcular las auto-correlaciones parciales de una serie de tiempo, utilizaremos la función: pacf(), A continuación los detalles de esta función:

```
pacf(x, lag.max, plot, na.action, ...)
```

Argumentos:

- x.- Un objeto numérico de serie temporal univariante o multivariado.
- lag.max.- Retardo máximo en la que calcular el acf.
- type.- Cadena de caracteres que da el tipo de acf a calcular.
- plot.- Lógico F/T, dibuja el acf.
- na.action.- Función a ser llamada para manejar valores perdidos.

Código de Ejemplo:

```
pacf(ret)
```

Se utiliza el mismo comando para el cálculo de auto-correlaciones en modelos de media y varianza. Es el usuario quien debe elegir la serie según corresponda correctamente.

2.8. Modelos ARIMA(p,d,q)

Para estimar modelos de media ARIMA(p,d,q), utilizaremos la función: arima(). A continuación los detalles de esta función:

```
arima(x, order = c(0L, 0L, 0L),
      seasonal = list(order = c(0L, 0L, 0L), period = NA),
      xreg = NULL, include.mean = TRUE,
      transform.pars = TRUE,
```

```
fixed = NULL, init = NULL,
method = c("CSS-ML", "ML", "CSS"),
SSinit = c("Gardner1980", "Rossignol2011"),
optim.method = "BFGS",
optim.control = list(), kappa = 1e6)
```

Argumentos:

- x.- Una serie temporal univariada.
- order.- Una especificación de la parte no estacional del modelo ARIMA: los tres componentes enteros (p, d, q) son el orden AR, el grado de diferenciación y el orden MA
- seasonal.- Una especificación de la parte estacional del modelo ARIMA, más el período (que por defecto a la frecuencia (x)). Esto debe ser una lista con el orden de los componentes y el período.
- xreq.- Opcionalmente, un vector o matriz de regresores externos, que debe tener el mismo número de filas que x.
- include.mean.- ¿Debería el modelo ARMA incluir un término medio / intercepto? T/F.
- transform.pars.- Los parámetros AR se transforman para asegurar que permanezcan en la región de estacionariedad, T/F
- fixed.- Vector numérico opcional de la misma longitud que el número total de parámetros.
- init.- Vector numérico opcional de los valores de los parámetros iniciales.
- method.- Método de ajuste: máxima verosimilitud o minimizar la suma de cuadrados condicionales.

- SSinit.- Una cadena que especifica el algoritmo para calcular la inicialización del estado-espacio de la verosimilitud.
- optim.method.- El valor pasado como el argumento del método a optim.
- optim.control.- Lista de parámetros de control para optim.
- kappa.- La varianza previa (como un múltiplo de la varianza de las innovaciones) para las observaciones pasadas en un modelo diferenciado. No reduzca esto.

Código de Ejemplo:

```
install.packages('tseries')
library('tseries')
```

arima001 <- arima(ret,c(0,0,1)) # -->> MA(1)

arima100 <- arima(ret,c(1,0,0)) # -->> AR(1)

arima101 <- arima(ret,c(1,0,1)) # -->> ARMA(1,1)

arima111 <- arima(ret,c(1,1,1)) # -->> ARIMA(1,1,1)

De la parte teórica sabemos que cuando una de las componentes del modelo ARIMA(p,d,q) es cero, esta parte es inexistente. Por ejemplo: un modelo ARIMA(1,0,1) es igual a un modelo ARMA(1,1). Lo mismo ocurre en las demás combinaciones de estos parámetros en la función de R, como se muestra en el código.

2.9. Modelos GARCH(r,s)

Para estimar modelos de varianza GARCH(r,s), utilizaremos la función: GARCH(). A continuación los detalles de esta función:

```
garch(x, order = c(1, 1), series = NULL, control = garch.control(...), ...)
```

Argumentos:

- x.- Un vector numérico o una serie de tiempo.
- order.- Un vector entero de dos dimensiones que da los órdenes del modelo para que se ajuste
- seies.- Nombre de la serie.
- control.- Una lista de parámetros de control configurada por garch.control.
- garch.control.- Otros configuraciones de la estimación del modelo.
- trace.- Trace optimizador de salida? T/F.

Código de Ejemplo:

```
install.packages('fGarch')

library('fGarch')

garch01 <- garch(arima100$residuals,order=c(0,1),trace=F)
# -->> ARCH(1)

garch11 <- garch(arima100$residuals,order=c(1,1),trace=F)
# -->> GARCH(1,1)
```

2.10. Criterio de elección entre Modelos

Sabemos que: una serie de tiempo puede estimarse con más de un modelo, lo que genera la necesidad de tener un criterio de selección. Para elegir el mejor de ellos, bajo este criterio.

Vamos a utilizar el AIC[6] como criterio de elección, entonces el modelo con AIC asociado de menor valor, será elegido como Mejor Modelo o Modelo Final. Esto se aplicará entre los Modelos que estiman la media y en los Modelos que estima la varianza de forma independiente.

---

[6] El criterio de información de Akaike (AIC) es una medida de la calidad relativa de un modelo estadístico. $AIC = 2k - 2\ln(L)$;

$Donde: k = num. parámetros; L = max. valor\ de\ la\ función\ de\ verosimilitud.$

Código de Ejemplo:

```
library('tseries')

library('fGarch')

# Para Modelos ARMA(p,d,q)

ModeloMedia <-
c('arima001','arima100','arima101','arima111')

ModeloMedia_ValorAIC <-
c(arima001$aic,arima100$aic,arima101$aic,arima111$aic)

ModeloMedia_TablaComparacion <-
data.frame(ModeloMedia,ModeloMedia_ValorAIC)

subset(ModeloMedia_TablaComparacion,ModeloMedia_ValorAIC ==
min(ModeloMedia_TablaComparacion$ModeloMedia_ValorAIC))

# Para Modelo  GARCH(r,s)

ModeloVarianza <- c('garch01','garch11')

ModeloVarianza_ValorAIC <- c(AIC(garch01),AIC(garch11))

ModeloVarianza_TablaComparacion <-
data.frame(ModeloVarianza,ModeloVarianza_ValorAIC)

subset(ModeloVarianza_TablaComparacion,ModeloVarianza_Valor
AIC ==
min(ModeloVarianza_TablaComparacion$ModeloVarianza_ValorAIC
))
```

3. Estimación de un modelo ARIMA-GARCH, para una serie de rendimientos financieros.

Para realizar la estimación de un modelo ARIMA-GARCH, hemos seleccionado la serie de precios ajustados semanales de Google, comprendida entre el 23/08/2004 hasta el 28/08/2017. Esta información fue consultada en Yahoo! Finance y guarda en GitHub para que el lector pueda replicar los cálculos, a continuación la dirección:

https://raw.githubusercontent.com/jorgepaguay86/EstadisticaAplicada_TFM_UGR/master/SerieDeTiempo_Google_Ago2004Ago2017Semanal.csv

Como inicio de esta estimación vamos a familiarizarnos con la serie de datos, mediante un análisis gráfico-descriptivo de la misma.

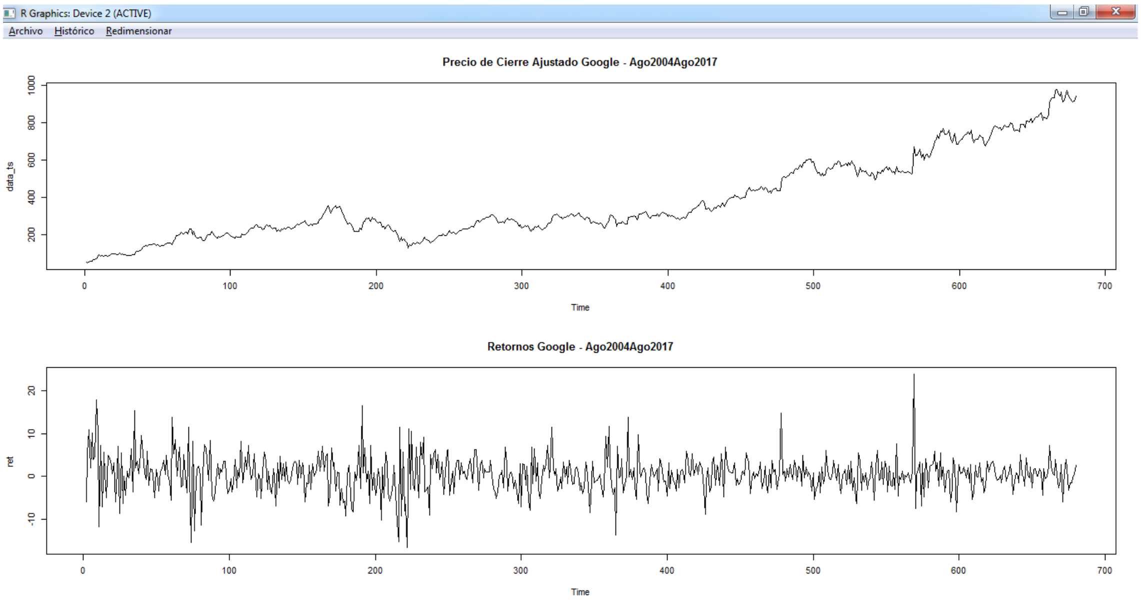


Figura 3.1: Gráfico, Precios de Cierre y Retornos Google Ago2004Jun2016

Estadísticos descriptivos:

```
> summary(data_ts)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  49.82  228.50  294.20  376.30  534.20  975.60
> summary(ret)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
-16.6700  -1.9320   0.5336   0.4237   2.6500  23.8500
```

Código en R:

```
install.packages('readr')

library('readr')

data <-
read.csv('https://raw.githubusercontent.com/jorgepaguay86/E
stadisticaAplicada_TFM_UGR/master/SerieDeTiempo_Google_Ago2
004Ago2017Semanal.csv')

data_ts <- ts(data$AdjClose) #ojo

ret = diff(log(data_ts))*100

win.graph(width=4.875, height=2.5,pointsize=8)

par(mfrow=c(2,1))

plot(data_ts,main = 'Precio de Cierre Ajustado Google -
Ago2004Ago2017')

plot(ret,main = 'Retornos Google - Ago2004Ago2017')

summary(data_ts)

summary(ret)
```

En la serie de precios ajustados podemos ver una tendencia creciente con picos moderados, esta serie es el insumo que nos permite calcular los rendimientos de la acción de Google, como hemos visto en el capítulo 2.

En la serie de rendimientos podemos ver que no existe tendencia, solo dispersión. Antes de la observación 400 se ve mayor dispersión, que después de esta observación. Esto sugiere heterocedasticidad en la serie de rendimientos.

En el presente trabajo, nos enfocaremos en la serie de rendimientos, ya que el objetivo planteado es estimar un modelo ARIMA-GARCH para una serie de rendimientos. Para realizar la estimación del ARIMA-GARCH, se seguirá los siguientes pasos:

1. Verificar que la serie es estacionaria mediante la Prueba de Raíces Unitarias.

2. Estimar un modelo de media ARIMA para la serie de retorno, con el fin de eliminar cualquier dependencia lineal en la serie.
3. Utilizar los residuos estandarizados y los residuos estandarizados al cuadrado del modelo de media para probar los efectos de GARCH.
4. Estimar un modelo de volatilidad si los efectos GARCH son estadísticamente significativos.
5. Comprobar el modelo ajustado cuidadosamente y refinarlo si es necesario.

3.1. Análisis de la serie de rendimientos, prueba de raíces unitarias.

Hemos visto que la serie de rendimientos no tiene tendencia lo cual nos sugiere que es estacionaria. En esta sección vamos a confirmar que la serie es estacionaria, utilizando la prueba de raíces unitarias.

```
> adf.test(ret) # p-value < 0.05 => no unit-root

	Augmented Dickey-Fuller Test

data:  ret
Dickey-Fuller = -9.1746, Lag order = 8, p-value = 0.01
alternative hypothesis: stationary
```

Código en R:

```
install.packages('tseries')

library('tseries')

adf.test(ret) # p-value < 0.05 => no unit-root
```

Con un p- value de 0.01, rechazamos Ho en favor de H1. Por tanto, la serie de retornos es estacionaria.

3.2. Estimación del modelo ARIMA

Sabiendo que la serie de retornos es estacionaria podemos continuar con la estimación de un modelo ARIMA, para esta estimación seguiremos los siguientes pasos:

1. Calcular las auto-correlaciones simple y parciales, para identificar los máximos valores posibles de p y q.

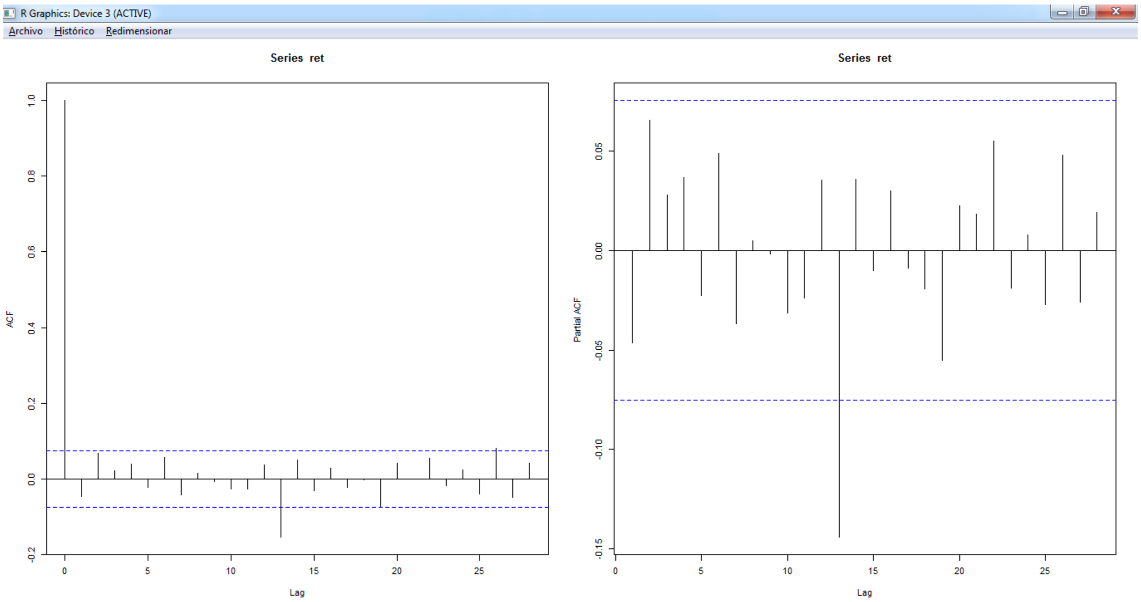


Figura 3.2: Gráfico, Auto-Correlaciones Simples y Parciales de la Serie de Retornos

```
win.graph(width=4.875, height=2.5,pointsize=8)
par(mfrow=c(1,2))
acf(ret)
pacf(ret)
```

En las ACF podemos ver que el retardo 2 está muy cercano a la banda de significancia por lo cual podemos asumirlo como válido, también vemos que el retardo 13 es significativo. Esto nos permite ver que el valor de q esta entre 2 y 13.

En las PACF podemos ver algo similar, el retardo 2 está muy cercano a la banda de significancia por lo cual podemos asumirlo como válido, también vemos que el retardo 13 es significativo. Esto nos permite ver que el valor de p esta entre 2 y 13.

2. Generar varios modelos plausibles para la estimación de la serie.

Para la generación de modelos plausibles utilizaremos los valores (5,5) correspondientes a (p,q). Dado que al introducir valores muy altos como 13, se generan modelos con muchos coeficientes que no son significativos.

Código en R:

```
{
  ModeloArima001 <- arima(ret, order = c(0,0,1))
  ModeloArima002 <- arima(ret, order = c(0,0,2))
  ModeloArima003 <- arima(ret, order = c(0,0,3))
  ModeloArima004 <- arima(ret, order = c(0,0,4))
  ModeloArima005 <- arima(ret, order = c(0,0,5))
  ModeloArima100 <- arima(ret, order = c(1,0,0))
  ModeloArima200 <- arima(ret, order = c(2,0,0))
  ModeloArima300 <- arima(ret, order = c(3,0,0))
  ModeloArima400 <- arima(ret, order = c(4,0,0))
  ModeloArima500 <- arima(ret, order = c(5,0,0))
  ModeloArima101 <- arima(ret, order = c(1,0,1))
  ModeloArima102 <- arima(ret, order = c(1,0,2))
  ModeloArima103 <- arima(ret, order = c(1,0,3))
  ModeloArima104 <- arima(ret, order = c(1,0,4))
  ModeloArima105 <- arima(ret, order = c(1,0,5))
  ModeloArima201 <- arima(ret, order = c(2,0,1))
  ModeloArima202 <- arima(ret, order = c(2,0,2))
  ModeloArima203 <- arima(ret, order = c(2,0,3))
  ModeloArima204 <- arima(ret, order = c(2,0,4))
  ModeloArima205 <- arima(ret, order = c(2,0,5))
  ModeloArima301 <- arima(ret, order = c(3,0,1))
```

```
  ModeloArima302 <- arima(ret, order = c(3,0,2))

  ModeloArima303 <- arima(ret, order = c(3,0,3))

  ModeloArima304 <- arima(ret, order = c(3,0,4))

  ModeloArima305 <- arima(ret, order = c(3,0,5))

  ModeloArima401 <- arima(ret, order = c(4,0,1))

  ModeloArima402 <- arima(ret, order = c(4,0,2))

  ModeloArima403 <- arima(ret, order = c(4,0,3))

  ModeloArima404 <- arima(ret, order = c(4,0,4))

  ModeloArima405 <- arima(ret, order = c(4,0,5))

  ModeloArima501 <- arima(ret, order = c(5,0,1))

  ModeloArima502 <- arima(ret, order = c(5,0,2))

  ModeloArima503 <- arima(ret, order = c(5,0,3))

  ModeloArima504 <- arima(ret, order = c(5,0,4))

  ModeloArima505 <- arima(ret, order = c(5,0,5))

}
```

3. Elegir el mejor de ellos utilizando el criterio de información AIC.

Para elegir el mejor modelo usaremos el criterio de información AIC, el modelo de menor AIC será elegido. Después de realizar estos cálculos, hemos encontrado que el mejor modelo de media es el ARIMA(4,0,3).

```
> subset(TablaValoresAICArima, TablaValoresAICArima$AIC == min(TablaVa
loresAICArima$AIC))
               df      AIC
ModeloArima403  9 3897.105
> coeftest(ModeloArima403)

z test of coefficients:

           Estimate Std. Error  z value Pr(>|z|)    
ar1        0.931324   0.050309  18.5123  < 2e-16 ***
ar2        0.909488   0.064262  14.1528  < 2e-16 ***
ar3       -0.938032   0.039763 -23.5906  < 2e-16 ***
ar4       -0.028039   0.040276  -0.6962  0.48632    
ma1       -0.989775   0.033617 -29.4429  < 2e-16 ***
ma2       -0.824374   0.062503 -13.1895  < 2e-16 ***
ma3        0.943500   0.032493  29.0371  < 2e-16 ***
```

```
intercept  0.418667   0.166276   2.5179  0.01181 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Código en R:

```
{
  TablaValoresAICArima <- AIC(
    ModeloArima001,
    ModeloArima002,
    ModeloArima003,
    ModeloArima004,
    ModeloArima005,
    ModeloArima100,
    ModeloArima200,
    ModeloArima300,
    ModeloArima400,
    ModeloArima500,
    ModeloArima101,
    ModeloArima102,
    ModeloArima103,
    ModeloArima104,
    ModeloArima105,
    ModeloArima201,
    ModeloArima202,
    ModeloArima203,
    ModeloArima204,
    ModeloArima205,
    ModeloArima301,
    ModeloArima302,
```

```
    ModeloArima303,
    ModeloArima304,
    ModeloArima305,
    ModeloArima401,
    ModeloArima402,
    ModeloArima403,
    ModeloArima404,
    ModeloArima405,
    ModeloArima501,
    ModeloArima502,
    ModeloArima503,
    ModeloArima504,
    ModeloArima505
      )
}
subset(TablaValoresAICArima, TablaValoresAICArima$AIC ==
min(TablaValoresAICArima$AIC))
library('lmtest')
coeftest(ModeloArima403)
```

4. Comprobar el modelo ajustado cuidadosamente y refinarlo si es necesario.

Para comprobar el ajuste utilizaremos la serie de residuos estandarizados el qqnorm asociado y sus correlogramas.

En la serie de residuos vemos un comportamiento aleatorio, lo cual habla bien de ellos, en el qqnorm vemos que los valores se aproximan a la línea de normalidad y en los correlogramas podemos ver que las autocorrelaciones simples y parciales son nos significativas, entonces podemos concluir que el ModeloArima403 proporciona un buen ajuste de la Serie de Retornos de Google, en el periodo analizado.

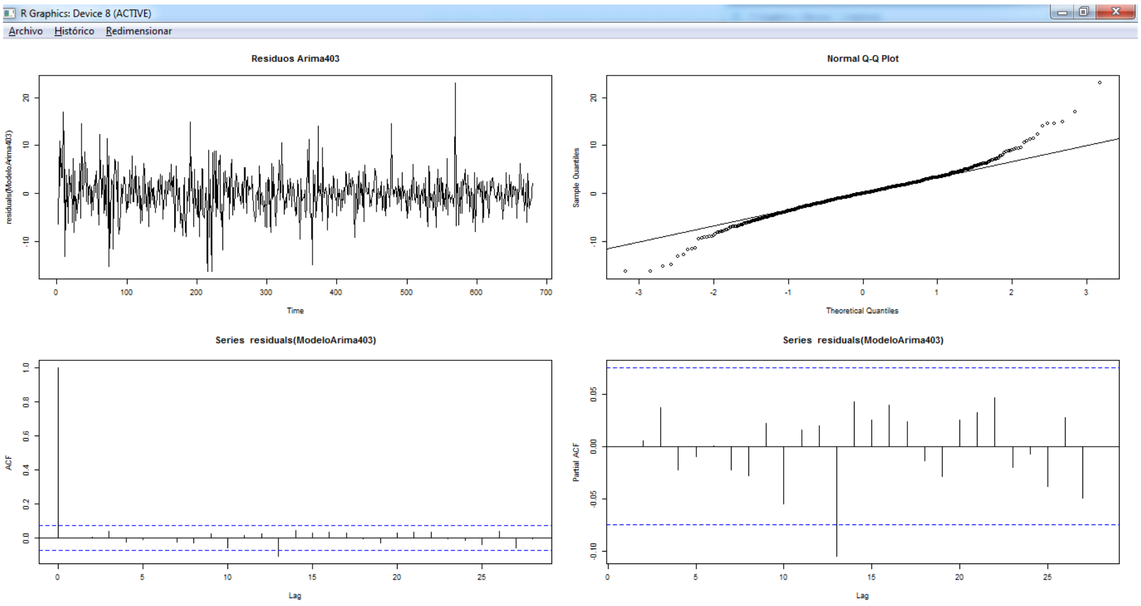


Figura 3.3: Gráfico, Evaluación de los Residuos Estandarizados del ModeloArima403

Código en R:

```
{
win.graph(width=4.875, height=2.5,pointsize=8)
par(mfrow=c(2,2))
plot.ts(residuals(ModeloArima403),main = 'Residuos
Arima403')
qqnorm(residuals(ModeloArima403))
qqline(residuals(ModeloArima403))
acf(residuals(ModeloArima403),na.action = na.omit)
pacf(residuals(ModeloArima403),na.action = na.omit)
}
```

3.3. Análisis de la Serie de Residuos Estandarizados del Modelo ARIMA, efectos GARCH

Para determinar si los efectos GARCH son significativos utilizaremos las pruebas de hipótesis: Ljung-Box y Multiplicador de Lagrange para efectos GARCH.

```
> Box.test(residuals(ModeloArima403)^2,type = 'Ljung-Box',lag=1)

	Box-Ljung test

data:  residuals(ModeloArima403)^2
X-squared = 13.862, df = 1, p-value = 0.0001968

> ArchTest(residuals(ModeloArima403))

	ARCH LM-test; Null hypothesis: no ARCH effects

data:  residuals(ModeloArima403)
Chi-squared = 36.005, df = 12, p-value = 0.0003234
```

Código en R:

```
Box.test(residuals(ModeloArima403)^2,type = 'Ljung-Box',lag=1)

library('FinTS')

ArchTest(residuals(ModeloArima403))
```

Los p-value de las pruebas son 0.0001968 y 0.0003234 respectivamente, en ambos casos rechazamos la Hipotesis nula en favor de la alternativa. Por tanto, los errores estandarizados al cuadro del Modelo Arima403 están correlacionados entre si y pueden ser Modelados mediante un proceso GARCH.

3.4. Estimación del modelo GARCH

Para la estimación del modelo GARCH, seguiremos los siguientes pasos:

1. Calcular las auto-correlaciones simple y parciales, para identificar los máximos valores posibles de r y s.

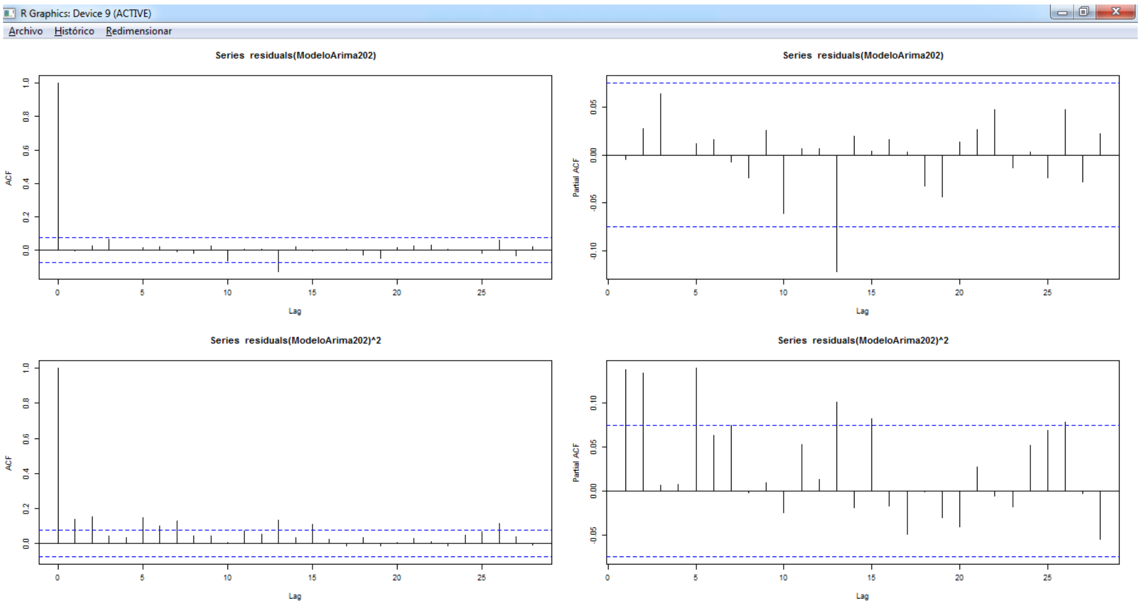


Figura 3.4: Gráfico, Auto-Correlaciones Simples y Parciales de la Serie de Residuos Estandarizados del Modelo Arima403.

Código en R:

```
{
win.graph(width=4.875, height=2.5,pointsize=8)
par(mfrow=c(2,2))
acf(residuals(ModeloArima403),na.action = na.omit)
pacf(residuals(ModeloArima403),na.action = na.omit)
acf(residuals(ModeloArima403)^2,na.action = na.omit)
pacf(residuals(ModeloArima403)^2,na.action = na.omit)
}
```

En los correlogramas de los residuos estandarizados vemos que las auto-correlaciones simples y parciales no pasan la banda de significancia por tanto no son significativas, lo que se esperaba del ajuste del modelo ARIMA de media.

Por otro lado, en los auto-correlogramas de los errores estandarizados al cuadrado vemos que existe significancia de las auto-correlaciones simples y parciales, hasta el retardo 15.

2. Generar varios modelos plausibles para la estimación en variabilidad de la serie.

Para la generación de modelos plausibles vamos a establecer los valores de (r,s) en (5,5), con el fin de evitar que se generen modelos con coeficientes no significativos.

Código en R:

```
{
  ModeloGarch01 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(0,1),na.action=na.omit)
  ModeloGarch02 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(0,2),na.action=na.omit)
  ModeloGarch03 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(0,3),na.action=na.omit)
  ModeloGarch04 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(0,4),na.action=na.omit)
  ModeloGarch05 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(0,5),na.action=na.omit)
  ModeloGarch10 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(1,0),na.action=na.omit)
  ModeloGarch20 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(2,0),na.action=na.omit)
```

```
  ModeloGarch30 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(3,0),na.action=na.omit)
  ModeloGarch40 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(4,0),na.action=na.omit)
  ModeloGarch50 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(5,0),na.action=na.omit)
  ModeloGarch11 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(1,1),na.action=na.omit)
  ModeloGarch12 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(1,2),na.action=na.omit)
  ModeloGarch13 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(1,3),na.action=na.omit)
  ModeloGarch14 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(1,4),na.action=na.omit)
  ModeloGarch15 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(1,5),na.action=na.omit)
  ModeloGarch21 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(2,1),na.action=na.omit)
  ModeloGarch22 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(2,2),na.action=na.omit)
```

```
  ModeloGarch23 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(2,3),na.action=na.omit)

  ModeloGarch24 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(2,4),na.action=na.omit)

  ModeloGarch25 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(2,5),na.action=na.omit)

  ModeloGarch31 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(3,1),na.action=na.omit)

  ModeloGarch32 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(3,2),na.action=na.omit)

  ModeloGarch33 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(3,3),na.action=na.omit)

  ModeloGarch34 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(3,4),na.action=na.omit)

  ModeloGarch35 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(3,5),na.action=na.omit)

  ModeloGarch41 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(4,1),na.action=na.omit)

  ModeloGarch42 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(4,2),na.action=na.omit)
```

```
  ModeloGarch43 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(4,3),na.action=na.omit)

  ModeloGarch44 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(4,4),na.action=na.omit)

  ModeloGarch45 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(4,5),na.action=na.omit)

  ModeloGarch51 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(5,1),na.action=na.omit)

  ModeloGarch52 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(5,2),na.action=na.omit)

  ModeloGarch53 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(5,3),na.action=na.omit)

  ModeloGarch54 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(5,4),na.action=na.omit)

  ModeloGarch55 <-
garch(na.omit(residuals(ModeloArima403)),order =
c(5,5),na.action=na.omit)

}
```

3. Elegir el mejor de ellos utilizando el criterio de información AIC.

Para elegir el mejor de estos modelos utilizaremos el criterio de información AIC, de modo que el modelo con menor AIC será nuestro modelo elegido.

```
> subset(TablaValoresAICGarch, TablaValoresAICGarch$AIC == min(TablaVa
loresAICGarch$AIC))
                  df       AIC
```

```
ModeloGarch15   7 3794.909
```

Después de realizar los cálculos antes descritos, obtenemos que el mejor modelo de varianza GARCH para ajustar los errores estandarizados del Modelo Arima403 es el Modelo Garch15.

```
> summary(ModeloGarch15)

Call:
garch(x = na.omit(residuals(ModeloArima403)), order = c(1, 5),     na.
action = na.omit)

Model:
GARCH(1,5)

Residuals:
     Min       1Q   Median       3Q      Max
-3.13180 -0.59711  0.00495  0.54506  6.24085

Coefficient(s):
    Estimate  Std. Error  t value Pr(>|t|)
a0 1.234e+01   3.762e+00    3.280  0.00104 **
a1 9.722e-02   4.112e-02    2.365  0.01805 *
a2 9.412e-02   5.145e-02    1.829  0.06737 .
a3 7.145e-03   4.315e-02    0.166  0.86847
a4 6.980e-14   3.866e-02    0.000  1.00000
a5 8.963e-02   3.504e-02    2.558  0.01053 *
b1 3.285e-02   2.875e-01    0.114  0.90904
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Diagnostic Tests:
        Jarque Bera Test

data:  Residuals
X-squared = 410.49, df = 2, p-value < 2.2e-16


        Box-Ljung test

data:  Squared.Residuals
X-squared = 0.037071, df = 1, p-value = 0.8473
```

Código en R:

```
{
  TablaValoresAICGarch <-
    AIC(
      ModeloGarch01,
      ModeloGarch02,
      ModeloGarch03,
```

```
ModeloGarch04,
ModeloGarch05,
ModeloGarch10,
ModeloGarch20,
ModeloGarch30,
ModeloGarch40,
ModeloGarch50,
ModeloGarch11,
ModeloGarch12,
ModeloGarch13,
ModeloGarch14,
ModeloGarch15,
ModeloGarch21,
ModeloGarch22,
ModeloGarch23,
ModeloGarch24,
ModeloGarch25,
ModeloGarch31,
ModeloGarch32,
ModeloGarch33,
ModeloGarch34,
ModeloGarch35,
ModeloGarch41,
ModeloGarch42,
ModeloGarch43,
ModeloGarch44,
ModeloGarch45,
```

```
       ModeloGarch51,
       ModeloGarch52,
       ModeloGarch53,
       ModeloGarch54,
       ModeloGarch55
          )
}
subset(TablaValoresAICGarch, TablaValoresAICGarch$AIC == min(TablaValoresAICGarch$AIC))
coeftest(ModeloGarch15)
summary(ModeloGarch15)
```

4. Comprobar el modelo ajustado cuidadosamente y refinarlo si es necesario.

Para comprobar el ajuste utilizaremos la serie de residuos estandarizados el qqnorm asociado y sus correlogramas.

En la serie de residuos vemos un comportamiento aleatorio, lo cual habla bien de ellos, en el qqnorm vemos que los valores se aproximan a la línea de normalidad y en los correlogramas de los errores estandarizados al cuadrado, podemos ver que las autocorrelaciones simples y parciales son nos significativas, entonces podemos concluir que el ModeloGarch15 proporciona un buen ajuste de la Serie de Residuos Estandarizados del Modelo Arima403, estimado anteriormente.

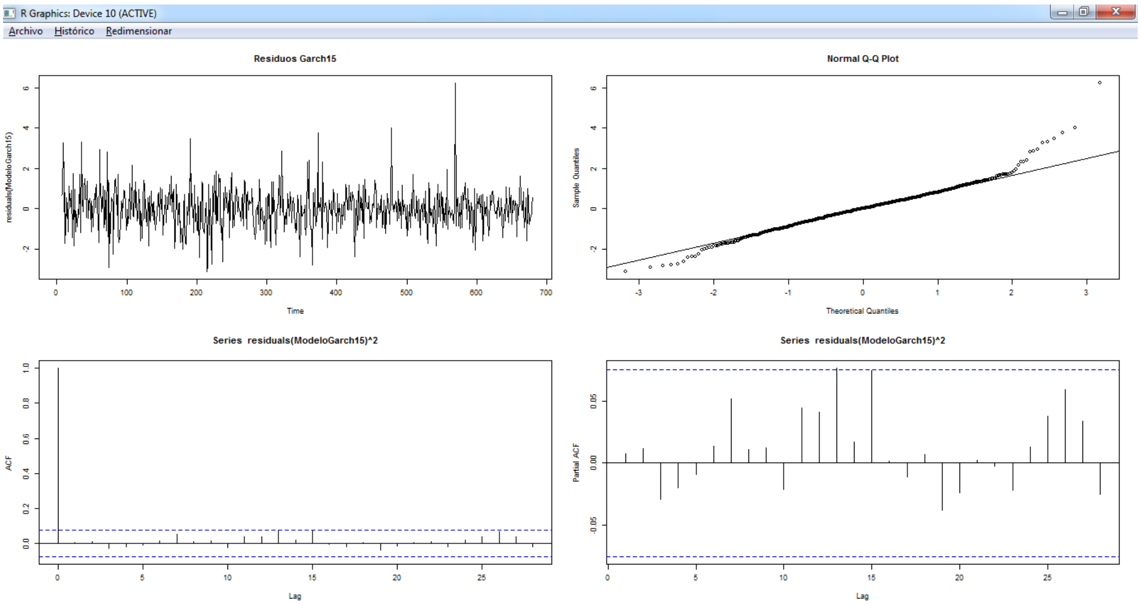


Figura 3.5: Gráfico, Evaluación de los Residuos Estandarizados del ModeloGarch15

Código en R:

```
{
win.graph(width=4.875, height=2.5,pointsize=8)
par(mfrow=c(2,2))
plot.ts(residuals(ModeloGarch15),main = 'Residuos
Garch15')
qqnorm(residuals(ModeloGarch15))
qqline(residuals(ModeloGarch15))
acf(residuals(ModeloGarch15)^2,na.action = na.omit)
pacf(residuals(ModeloGarch15)^2,na.action = na.omit)
}
```

3.5. Presentación Modelo ARIMA-GARCH estimado

Como resultado de la estimación plantea al inicio de este capítulo, obtenemos que el Modelo ARIMA(4,03) - GRACH(1,5) es el mejor estimador de la Serie de Retornos de Google en el periodo analizado.

Modelo: ARIMA(4,03) - GRACH(1,5)

$$ret_t = \ 0.418667 + 0.931324\, ret_{t-1} + 0.909488\, ret_{t-2} - 0.938032\, ret_{t-3}$$
$$- 0.028039\, ret_{t-4} + \ u_t \ - 0.989775 + u_{t-1} - 0.824374\, u_{t-2}$$
$$+ 0.943500\, u_{t-3}$$

Donde:

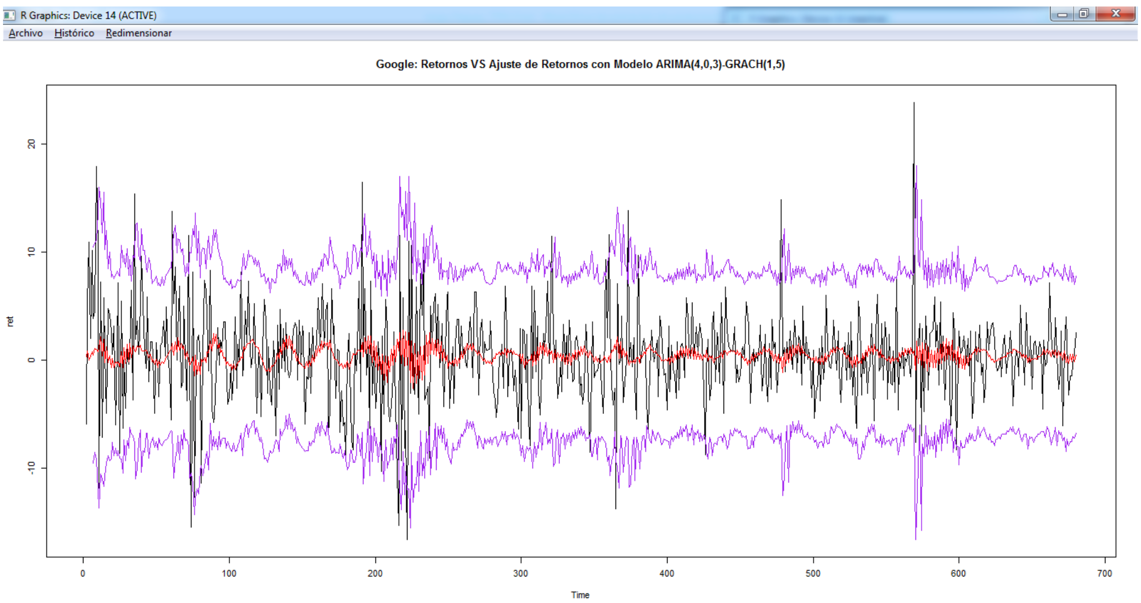
ret_t: Es el rendimiento de una acción de Google en el tiempo t.

$$u_t = \sqrt{h_t \varepsilon_t}$$

$$h_t = 12.341 + 0.097225\, ret^2{}_{t-1} + \ 0.094121\, ret^2{}_{t-2} + 0.0071455\, ret^2{}_{t-3}$$
$$+ 0.08963\, ret^2{}_{t-5} + 0.032851\, h^2{}_{t-1}$$

ε_t: Es un Ruido Blanco de variables independientes e idénticamente

Gráficamente tenemos:



Código en R:

```
{
fit <- fitted.values(ModeloArima403)
```

```
fitgarch <- fitted.values(ModeloGarch15)[,1]
low <- fit - (1.96 * fitgarch)
high <- fit + (1.96 * fitgarch)
win.graph(width=4.875, height=2.5,pointsize=8)
plot(ret,
      main = 'Google: Retornos VS Ajuste de Retornos con
Modelo ARIMA(4,0,3)-GRACH(1,5)',
      type = 'l')
lines(low,col = 'purple')
lines(high,col = 'purple')
lines(fit,col = 'red')
}
```

Bibliografía.

1. Cryer. J., Chan. K. (2012). Time series analysis with applications in R, Springer
2. Gujarati, D., Porter, D.. (2010). Econometría, Mc Graw Hill.
3. Peña, D. (2005). Análisis de series temporales. Alianza Editorial.
4. Tsay, R. S. (2002). Analisys of Financial Time Series. New York: John Wiley &Sons Inc.
5. Eric Zivot (2004). Working with Financial Time Series Data in R, Department of Economics, University of Washington