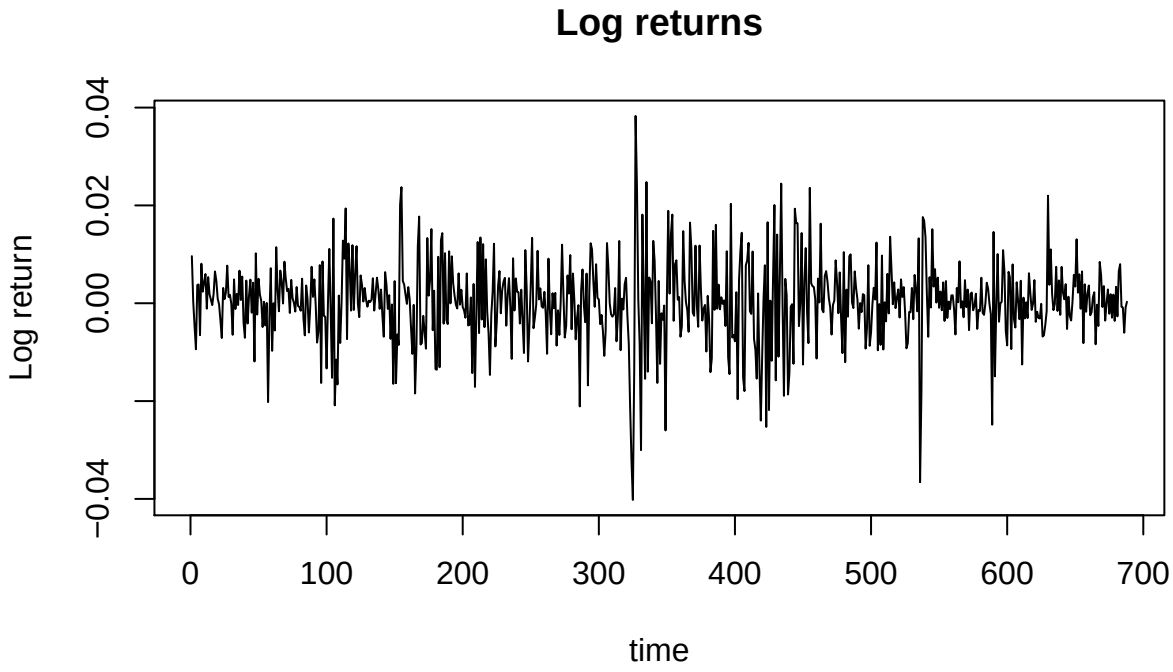
GARCH models

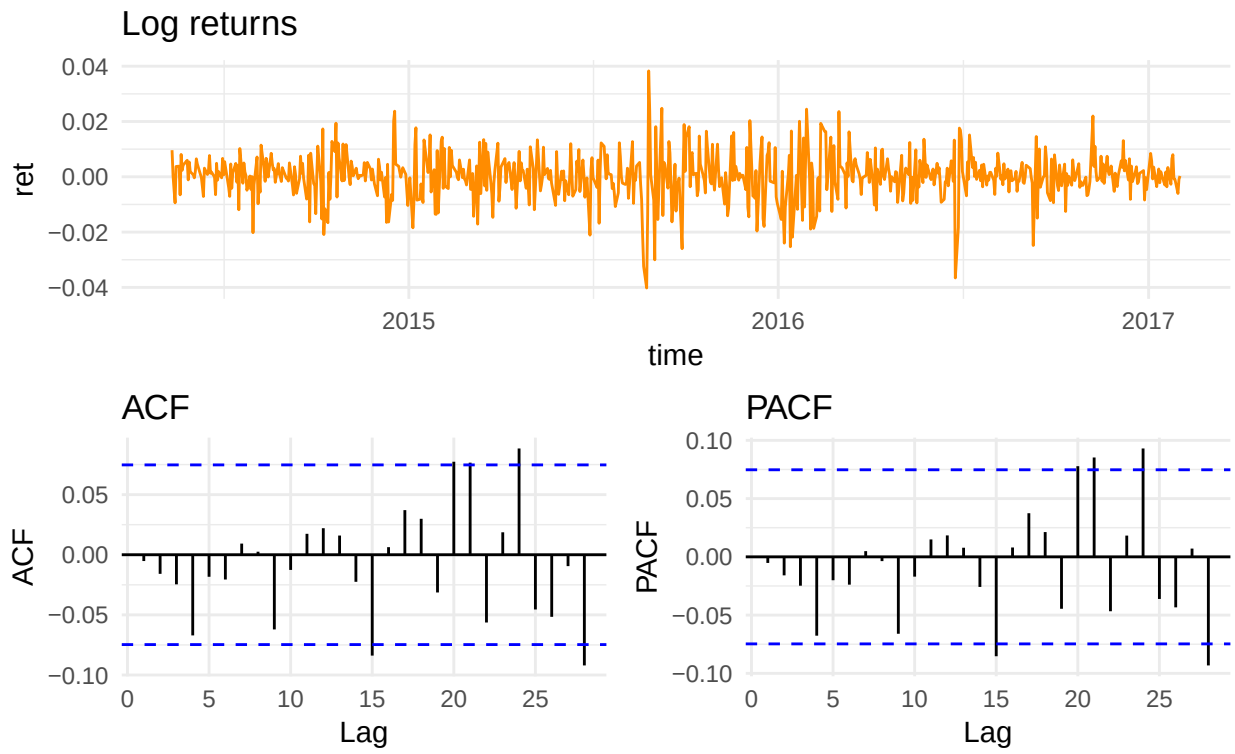
Josep Puig

Uno de los temas más actuales de las series temporales es la predicción de las volatilidades de los retornos de un determinado activo (normalmente acciones de empresas). Estos modelos se conoce como **conditional heteroscedastic models**. Utilizado por ejemplo cuando se intenta estimar el precio de una opción 'call' utilizando el método de Black-Scholes, tambien para modelos de Value at risk para una determinada posición de un portfolio, etc.

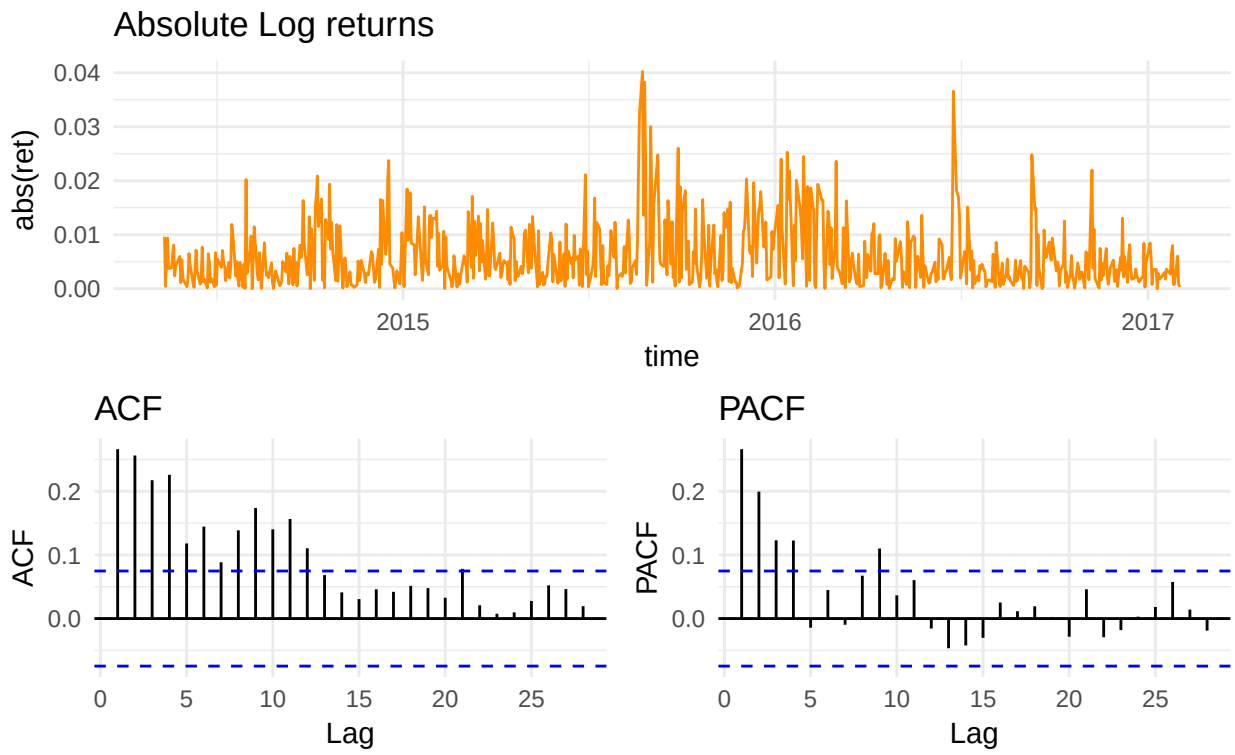
En la siguiente figura observamos como los logs return de **S&P 500** siguen claramente un ruido blanco; por lo tante hacen que un modelo ARIMA o ETS no sea útil para predecir la serie. Sin embargo, se puede observar que la varianza no es constante a lo largo del tiempo:

```
## time series ends   2017-02-01
```





Se observa que la serie sigue claramente un ruido blanco. Sin embargo, si representamos los valores absolutos si que podemos observar que no sigue un ruido blanco:



Se observa en este caso que si que hay una relación.

El modelo general se basa en que los **log returns** (r_t) es función de:

$$r_t = \mu + a_t \quad ; \quad a_t = \sigma_t \epsilon_t$$

Donde μ es la media de la serie (0 en la gran mayoría de los **logs returns**), y $\epsilon \sim N(0,1)$.

σ_t^2 es lo que los modelos GARCH intentarán predecir.

ARCH(p)

Modelo desarrollado por Engle (1982). Es el modelo más sencillo de los modelos GARCH. ARCH(p) sigue la siguiente estructura:

$$a_t = \sigma_t \epsilon_t, \qquad \sigma_t^2 = \omega + \alpha_1 a_{t-1}^2 + \alpha_2 a_{t-2}^2 + \cdots + \alpha_p a_{t-p}^2$$

ARCH(1) por ejemplo es:

$$\sigma_t^2 = \alpha_0 + \alpha_1 a_{t-1}^2$$

GARCH(p,q)

Ampliación del modelo ARCH(p) propuesto por Bollerslev (1986)

$$\sigma_t^2 = \omega + \sum_{j=1}^{p} \alpha_j r_{t-j} + \sum_{j=1}^{q} \beta_j \sigma_{t-j}$$

Modelo GARCH exponencial

$$ln(\sigma_t^2) = \omega + \sum_{j=1}^{q} \left(\alpha_j z_{t-j} + \gamma(|z_{t-j}| - E|z_{t-j}|)\right) + \sum_{j=1}^{p} \beta_j ln(\sigma_{t-1}^2)$$

z_t es el ϵ_t estandarizado (el error estadarizado).

Caso particular de las acciones de telefonica:

```
library(moments)
```

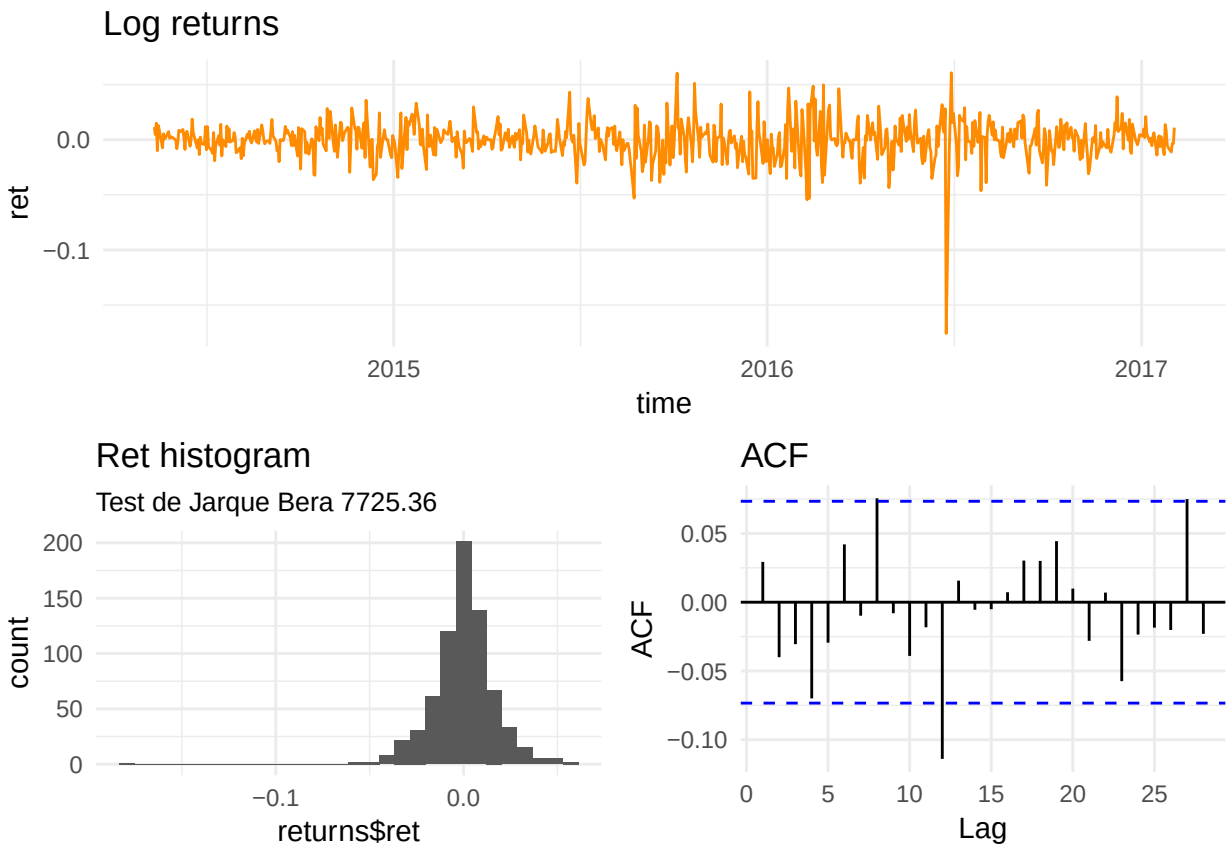
```
##
## Attaching package: 'moments'
```

```
## The following objects are masked from 'package:timeDate':
##
##     kurtosis, skewness
```

```
cotizaciones = get.hist.quote("TEF.MC", start= Sys.Date() - 1000,
               end= Sys.Date(), quote="AdjClose",
               provider="yahoo",
               compression="d", retclass="zoo")
returns = data.frame(time = time(cotizaciones)[-1], ret = diff(log(cotizaciones$AdjClose)) )
g1 <- ggplot(returns) + geom_line(aes(x=time,y=ret), col='darkorange') +
  ggtitle('Log returns')+theme_minimal()
```

```
g2 <- ggplot(NULL) + geom_histogram(aes(x = returns$ret)) +
  labs(title='Ret histogram',
       subtitle = paste('Test de Jarque Bera', round(jarque.test(returns$ret)$statistic,2))) +
  theme_minimal()
g3 <- ggAcf(returns$ret)+theme_minimal() + ggtitle('ACF')
grid.arrange(g1, arrangeGrob(g2,g3, ncol=2))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Estimamos modelo con el paquete **rugarch** CON INTERVENCIÓN EN LA MEDIA: Dia 23 de junio pasó el brexit. Creo una variable dicotómica 0,1 que cuando sea el dia 24 de junio de 2016 (el dia después) sea 1 y el resto 0.

```
library(rugarch)
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'rugarch'
```

```
## The following object is masked from 'package:stats':
##
##     sigma
```

```
spec1=ugarchspec(variance.model = list(model="eGARCH", submodel = NULL,garchOrder=c(1,2),
                                       external.regressors  =
                                         as.matrix(1*(returns$time == '2016-06-24'))),
                 mean.model = list(armaOrder=c(0,0), include.mean=F),
```

```
                  distribution.model = 'norm')
mod = ugarchfit(data = returns$ret,spec=spec1)
```

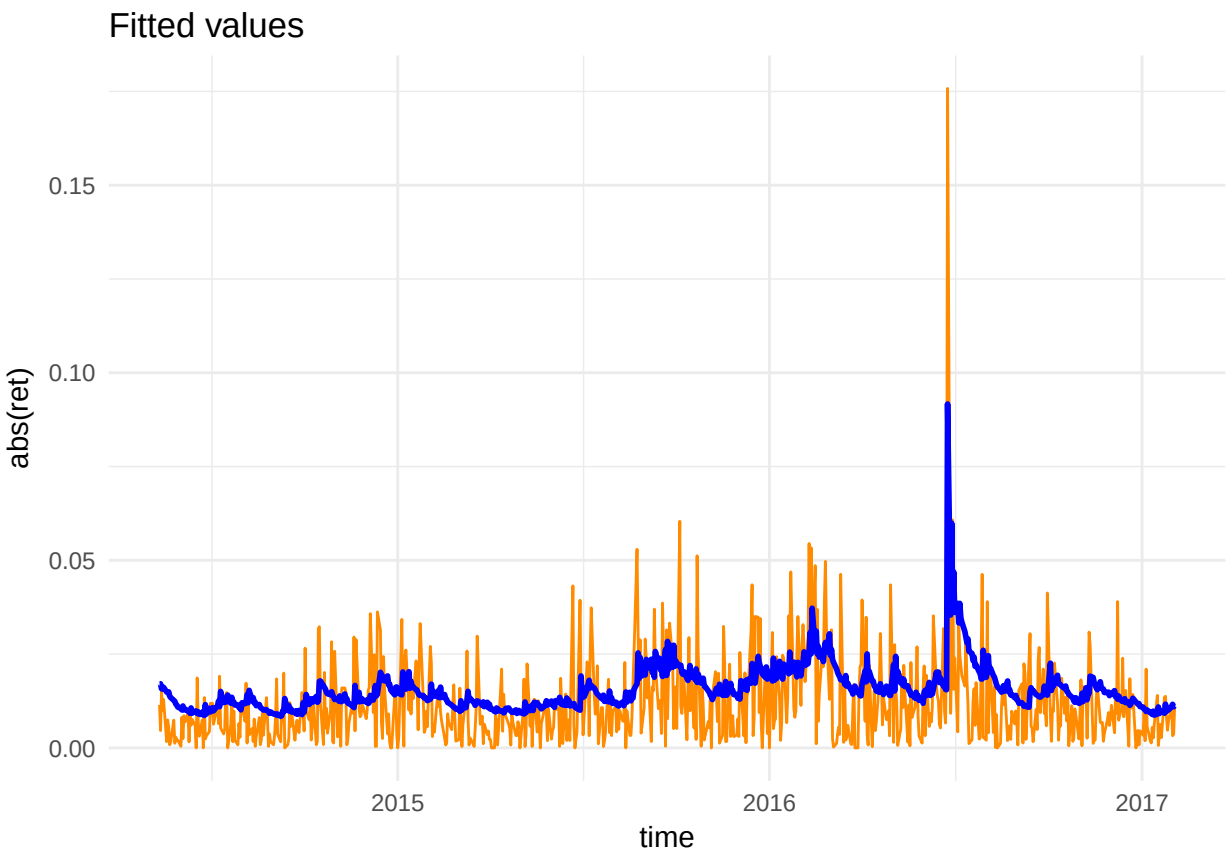
```
results = data.frame( mod@fit$matcoef)
stargazer(results, summary = FALSE, header=FALSE)
```

Table 1:

	X.Estimate	X.Std..Error	X.t.value	Pr...t..
omega	-0.372	0.128	-2.913	0.004
alpha1	-0.144	0.033	-4.344	0.00001
beta1	0.324	0.029	11.065	0
beta2	0.632	0.027	22.986	0
gamma1	0.206	0.052	3.939	0.0001
vxreg1	3.604	0.678	5.313	0.00000

Fitted Values y residuos

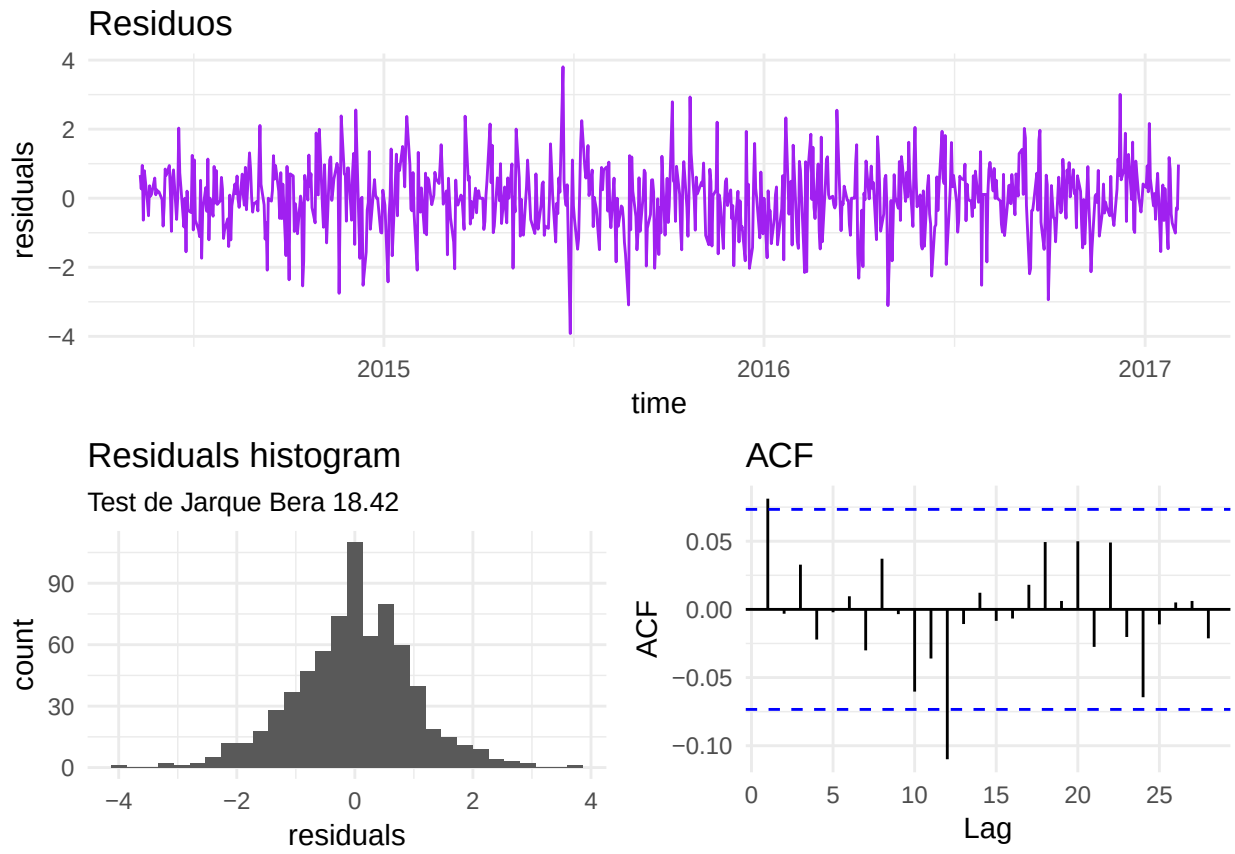
```
fit = mod@fit$sigma

ggplot(returns) + geom_line(aes(x=time,y=abs(ret)), col='darkorange') +
  geom_line(aes(x = returns$time, y=fit), col='blue', size=1.0) +
  theme_minimal() +
  ggtitle('Fitted values')
```



Residuales

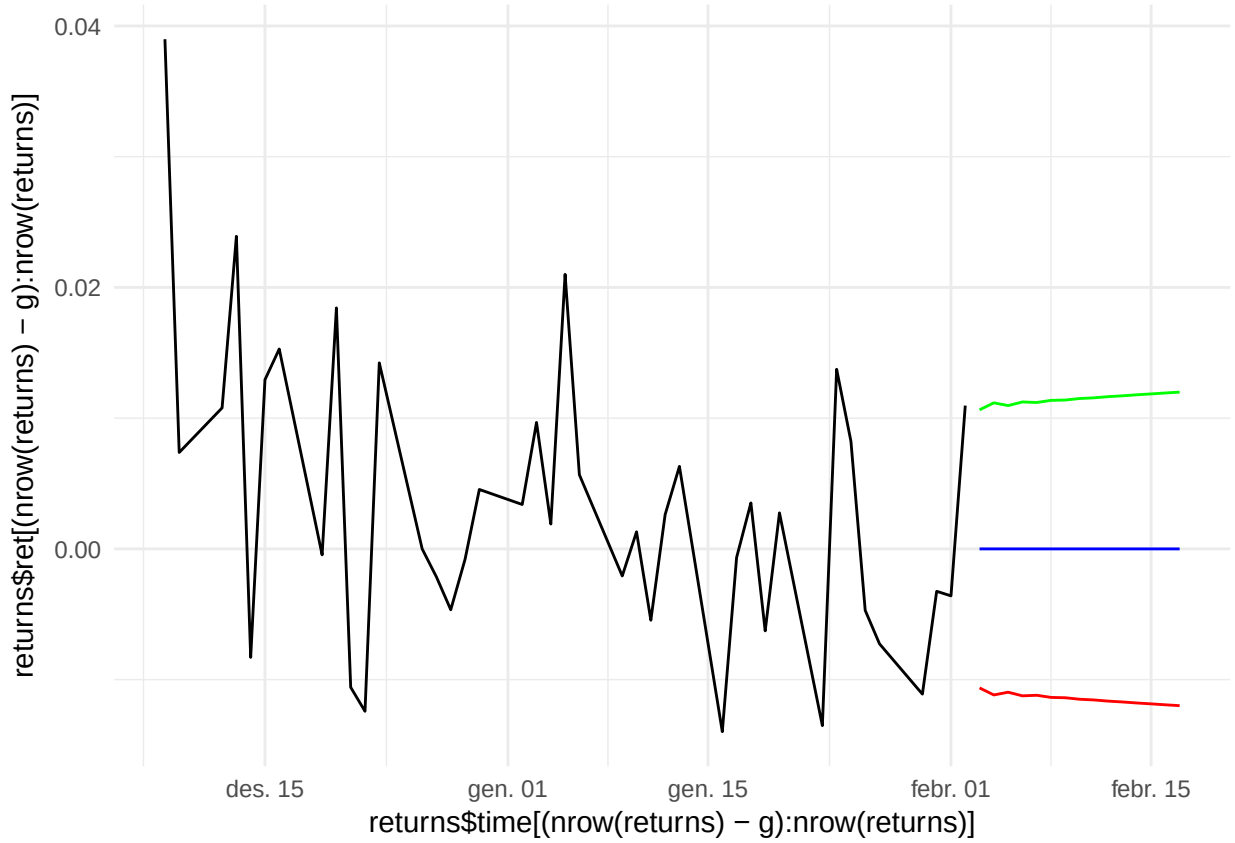
```
residuals = mod@fit$z
g1 <- ggplot(returns) + geom_line(aes(x=time,y=residuals), col='purple') + theme_minimal()+
  ggtitle('Residuos')
g2 <- ggplot(NULL) + geom_histogram(aes(x = residuals)) +
  labs(title='Residuals histogram',
       subtitle = paste('Test de Jarque Bera', round(jarque.test(residuals)$statistic,2))) +
  theme_minimal()
g3 <- ggAcf(residuals)+theme_minimal() + ggtitle('ACF')
grid.arrange(g1, arrangeGrob(g2,g3, ncol=2))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

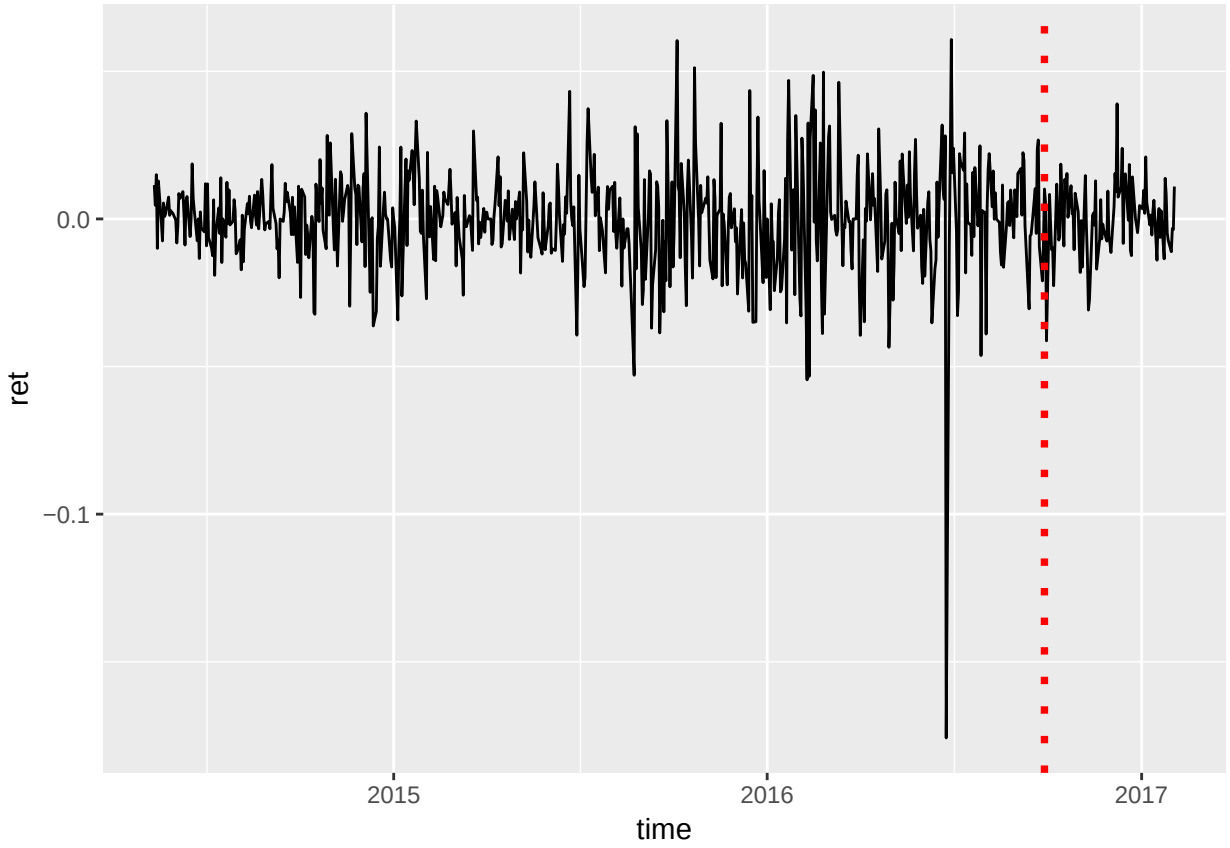


Forecast

```
n = 15 #numero de predicciones
g = 40 #Dias anteriores a la prediccion para graficar
forc =  ugarchforecast(mod, n.ahead = n)
forecast = data.frame(ret =
                        c(returns$ret[(nrow(returns)-g):nrow(returns)],forc@forecast$seriesFor),
                      date = c(returns$time[(nrow(returns)-g):nrow(returns)],
                               seq(from=as.Date(returns$time[length(returns$time)]+1),
                                   length.out = n, by='day')))
posdate = seq(from=as.Date(returns$time[length(returns$time)]+1),
              length.out = n, by='day')
ggplot(NULL) + geom_line(aes(x =returns$time[(nrow(returns)-g):nrow(returns)],
                             y = returns$ret[(nrow(returns)-g):nrow(returns)]))+
  geom_line(aes(x = posdate, y = forc@forecast$seriesFor), col='blue') +
  geom_line(aes(x = posdate, y = forc@forecast$sigmaFor),col='green') +
  geom_line(aes(x = posdate, y = -forc@forecast$sigmaFor),col='red') +
  theme_minimal()
```




```
ggplot(returns, aes(x=time,y=ret)) + geom_line() +
  geom_vline(aes(xintercept = as.numeric(returns$time[returns$time=="2016-09-28"])),linetype=3, col='re
```




```
      brex = which(returns$time=='2016-06-23')
      n = 2
      n1 = 4
      leng = nrow(returns)
      xreg = c()
      for (i in n:n1){
        reg = rep(0,leng)
        reg[i+brex] = 1
        xreg = cbind(xreg,reg)
      }
      colnames(xreg) = paste(paste('Dia',n:n1, sep=''))
      xreg = as.matrix(xreg)

spec1=ugarchspec(variance.model = list(model="eGARCH", submodel = NULL,garchOrder=c(1,1)),
                    mean.model = list(armaOrder=c(0,0), include.mean=T, external.regressors = xreg))
mod = ugarchfit(data=returns$ret,spec=spec1)
forc =  ugarchforecast(mod, n.ahead = 10)
```