Homework 5

Marcelo Santos Carielo

ago/2023

Questão - Modelagem das ações da VALE: uma abordagem ARMA-GARCH

Seguindo o roteiro visto em aula, ajuste modelos GARCH(1, 1), EGARCH(1, 1) e GJR(1, 1) para as ações da VALE, usando as distribuições t-Student e Normal. Verifique qual é o melhor modelo dentre esses, verificando também os resíduos. Apresente previsões para a volatilidade condicional.

Resposta

Para este exercício, usaremos a série de retornos do VALE de `01/01/2019` até o dia de hoje (`29-07-29`). O código abaixo coleta esses dados do Yahoo Finance.

```
library(rugarch)
library(BatchGetSymbols)

# define datas de início e fim
date_init <- "2019-01-01"
date_end <- "2023-07-29"
#date_end <- Sys.Date()

# coleta dados da VALE
tickers <- c("VALE3.SA")
assets <- BatchGetSymbols(tickers=tickers,
                          first.date=date_init,
                          last.date=date_end,
                          type.return="log", # log retorno
                          freq.data="daily")
assets <- assets[[2]]

vale <- assets %>%
  filter(ticker=="VALE3.SA")
```

Agora vemos um resumo estatístico e transformamos os dados para o formato de série temporal:

```
library(fBasics)

daily_returns_vale <- vale %>%
  select(ref.date, ret.closing.prices)

basicStats(daily_returns_vale$ret.closing.prices)
```

```
##             X..daily_returns_vale.ret.closing.prices
## nobs                                     1137.000000
## NAs                                         1.000000
## Minimum                                    -0.281822
## Maximum                                     0.193574
## 1. Quartile                                -0.013279
## 3. Quartile                                 0.012907
## Mean                                        0.000247
## Median                                      0.000000
## Sum                                         0.280463
## SE Mean                                     0.000784
## LCL Mean                                   -0.001292
## UCL Mean                                    0.001786
## Variance                                    0.000699
## Stdev                                       0.026431
## Skewness                                   -0.775017
## Kurtosis                                   17.355477
```

```
date <- daily_returns_vale %>%
  select(ref.date) %>%
  rename(date=ref.date) %>%
  slice(-1)

daily_returns_vale <- daily_returns_vale %>%
  select(ret.closing.prices) %>%
  slice(-1)

daily_returns_vale <- as.ts(daily_returns_vale)
```

O resumo estatístico acima mostra que a curtose ficou maior do que 3, indicando que a série analisada possui cauda pesada. Além disso, também notamos que a média ficou por volta de zero. Estes resultados estão dentro do esperado (média zero e cauda pesada).

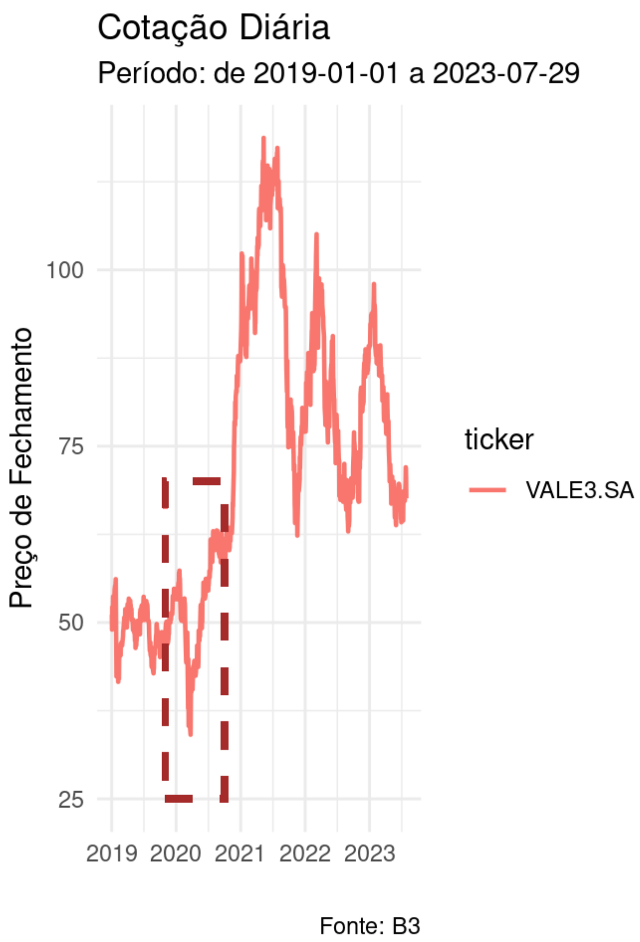
Vejamos os gráficos do preço diário e do log-retorno da série temporal da VALE.

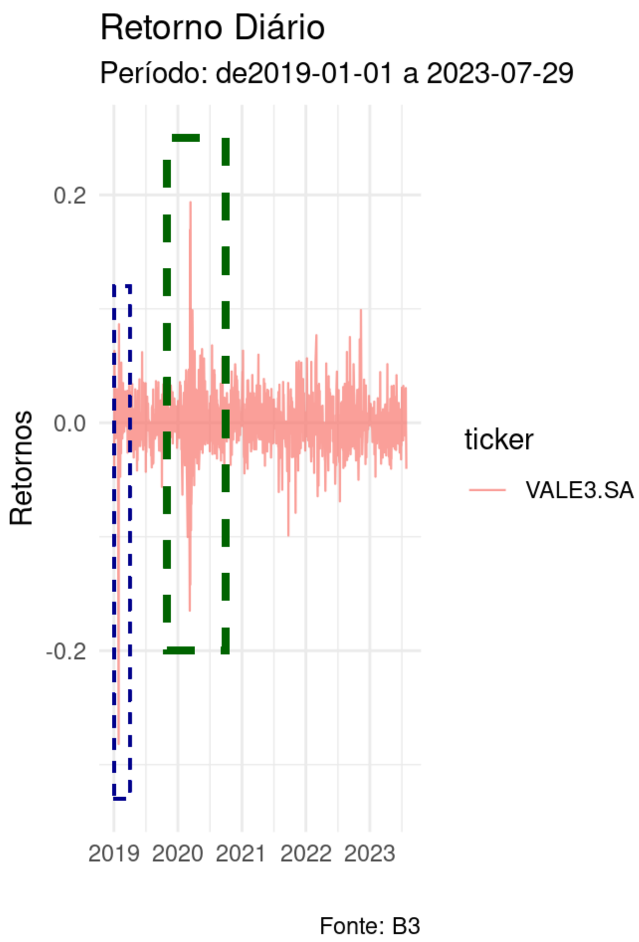
```
library(ggplot2, quietly=TRUE)
library(gridExtra, quietly=TRUE)

g <- ggplot(data=assets) +
  geom_line(mapping=aes(x=ref.date, y=price.close, color=ticker),
            linewidth=0.8, na.rm=TRUE) +
  geom_rect(aes(xmin=as.Date("2019-11-01"), xmax=as.Date("2020-10-01"),
                ymin=25, ymax=70),
            fill="transparent", linetype=2, color="brown", size=1.2) +
  labs(x="", y="Preço de Fechamento",
       title="Cotação Diária",
       subtitle=paste("Período: de ", date_init, " a ", date_end, sep=""),
       caption="Fonte: B3") +
  theme_minimal()

g.returns <- ggplot(data=assets) +
  geom_line(aes(x=ref.date, y=ret.closing.prices, color=ticker),
            alpha=0.7, linewidth=0.4, na.rm=TRUE) +
  geom_rect(aes(xmin=as.Date("2019-11-01"), xmax=as.Date("2020-10-01"),
                ymin=-0.2, ymax=0.25),
            fill="transparent", linetype=2, color="darkgreen", size=1.2) +
  geom_rect(aes(xmin=as.Date("2019-01-01"), xmax=as.Date("2019-04-01"),
                ymin=-0.33, ymax=0.12),
            fill="transparent", linetype=2, color="darkblue", size=0.5) +
  labs(x="" , y="Retornos",
       title="Retorno Diário",
        subtitle=paste("Período: de", date_init, " a ", date_end, sep=""),
        caption="Fonte: B3") +
    theme_minimal()

grid.arrange(g, g.returns, nrow=1, ncol=2)
```





O gráfico dos retornos mostra o aumento da volatilidade no período de início da pandemia no Brasil. Notamos ainda o início de 2019 também foi um período de alta volatilidade das ações da VALE.

A seguir vamos estimar modelos GARCH(1, 1), EGARCH(1, 1) e GJR(1, 1) para a série de retornos da VALE, usando as distribuições t-Student e Normal.

TESTE LM

A hipótese nula do Teste LM é que não há heterocedasticidade condicional (efeito ARCH). O código a seguir realiza o Teste LM para lags 1, 2, 3, 5, 10 e 15.

```
library(FinTS)
ArchTest(daily_returns_vale, lags=1,demean=TRUE)
```

```
##
##  ARCH LM-test; Null hypothesis: no ARCH effects
##
## data:  daily_returns_vale
## Chi-squared = 52.697, df = 1, p-value = 3.891e-13
```

```
ArchTest(daily_returns_vale, lags=2,demean=TRUE)
```

```
##
##  ARCH LM-test; Null hypothesis: no ARCH effects
##
## data:  daily_returns_vale
## Chi-squared = 75.953, df = 2, p-value < 2.2e-16
```

```
ArchTest(daily_returns_vale, lags=3,demean=TRUE)
```

```
##
##  ARCH LM-test; Null hypothesis: no ARCH effects
##
## data:  daily_returns_vale
## Chi-squared = 94.783, df = 3, p-value < 2.2e-16
```

```
ArchTest(daily_returns_vale, lags=5,demean=TRUE)
```

```
##
##  ARCH LM-test; Null hypothesis: no ARCH effects
##
## data:  daily_returns_vale
## Chi-squared = 99.699, df = 5, p-value < 2.2e-16
```

```
ArchTest(daily_returns_vale, lags=10,demean=TRUE)
```

```
##
##  ARCH LM-test; Null hypothesis: no ARCH effects
##
## data:  daily_returns_vale
## Chi-squared = 103.6, df = 10, p-value < 2.2e-16
```

```
ArchTest(daily_returns_vale, lags=20,demean=TRUE)
```

```
##
##  ARCH LM-test; Null hypothesis: no ARCH effects
##
## data:  daily_returns_vale
## Chi-squared = 464.59, df = 20, p-value < 2.2e-16
```

Em todos os casos acima não rejeitamos a hipótese nula, pois $p < 0.05$. Logo, a série das variâncias não é autocorrelacionada e uma boa opção para modelarmos os retornos da VALE é usarmos modelos da família ARCH.

GARCH(1, 1)

Estimamos um modelo GARCH(1, 1) para a série da VALE com o seguinte código:

```
garch.spec.student <- ugarchspec(variance.model=list(model="sGARCH",
                                                     garchOrder=c(1, 1)),
                                 mean.model=list(armaOrder=c(1, 1),
                                                 include.mean=TRUE),
                                 distribution.model="std")
garch.spec.normal <- ugarchspec(variance.model=list(model="sGARCH",
                                                    garchOrder=c(1, 1)),
                                mean.model=list(armaOrder=c(1, 1),
                                                include.mean=TRUE),
                                distribution.model="norm")

garch.fit.vale.student <- ugarchfit(spec=garch.spec.student,
                                    data=daily_returns_vale)

garch.fit.vale.normal <- ugarchfit(spec=garch.spec.normal,
                                   data=daily_returns_vale)

garch.fit.vale.student
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(1,0,1)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000496    0.000539  0.92035 0.357391
## ar1     0.596578    0.220662  2.70358 0.006860
## ma1    -0.640059    0.210183 -3.04525 0.002325
## omega   0.000071    0.000022  3.16715 0.001539
## alpha1  0.105720    0.030508  3.46529 0.000530
## beta1   0.769408    0.057060 13.48420 0.000000
## shape   5.242010    0.770816  6.80060 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000496    0.000576  0.86033 0.389605
## ar1     0.596578    0.139434  4.27856 0.000019
## ma1    -0.640059    0.131380 -4.87183 0.000001
## omega   0.000071    0.000019  3.64798 0.000264
## alpha1  0.105720    0.042408  2.49292 0.012670
## beta1   0.769408    0.054136 14.21252 0.000000
## shape   5.242010    1.006414  5.20860 0.000000
##
## LogLikelihood : 2711.734
##
## Information Criteria
## ------------------------------------
##
## Akaike       -4.7619
## Bayes        -4.7308
## Shibata      -4.7619
## Hannan-Quinn -4.7501
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.1247  0.7240
## Lag[2*(p+q)+(p+q)-1][5]    1.0140  1.0000
## Lag[4*(p+q)+(p+q)-1][9]    3.1502  0.8666
## d.o.f=2
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
```

```
## Lag[1]                     0.02059  0.8859
## Lag[2*(p+q)+(p+q)-1][5]   0.11148  0.9977
## Lag[4*(p+q)+(p+q)-1][9]   0.16299  1.0000
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]   0.03452 0.500 2.000  0.8526
## ARCH Lag[5]   0.09693 1.440 1.667  0.9874
## ARCH Lag[7]   0.12091 2.315 1.543  0.9990
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  2.3093
## Individual Statistics:
## mu     0.3600
## ar1    0.4824
## ma1    0.4180
## omega  0.1927
## alpha1 0.2895
## beta1  0.2317
## shape  0.5620
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.69 1.9 2.35
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           1.0079 0.3137
## Negative Sign Bias  0.7736 0.4393
## Positive Sign Bias  0.5772 0.5639
## Joint Effect        1.1813 0.7575
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     18.72       0.4750
## 2    30     31.25       0.3536
## 3    40     32.10       0.7752
## 4    50     45.78       0.6045
##
##
## Elapsed time : 0.2370598
```

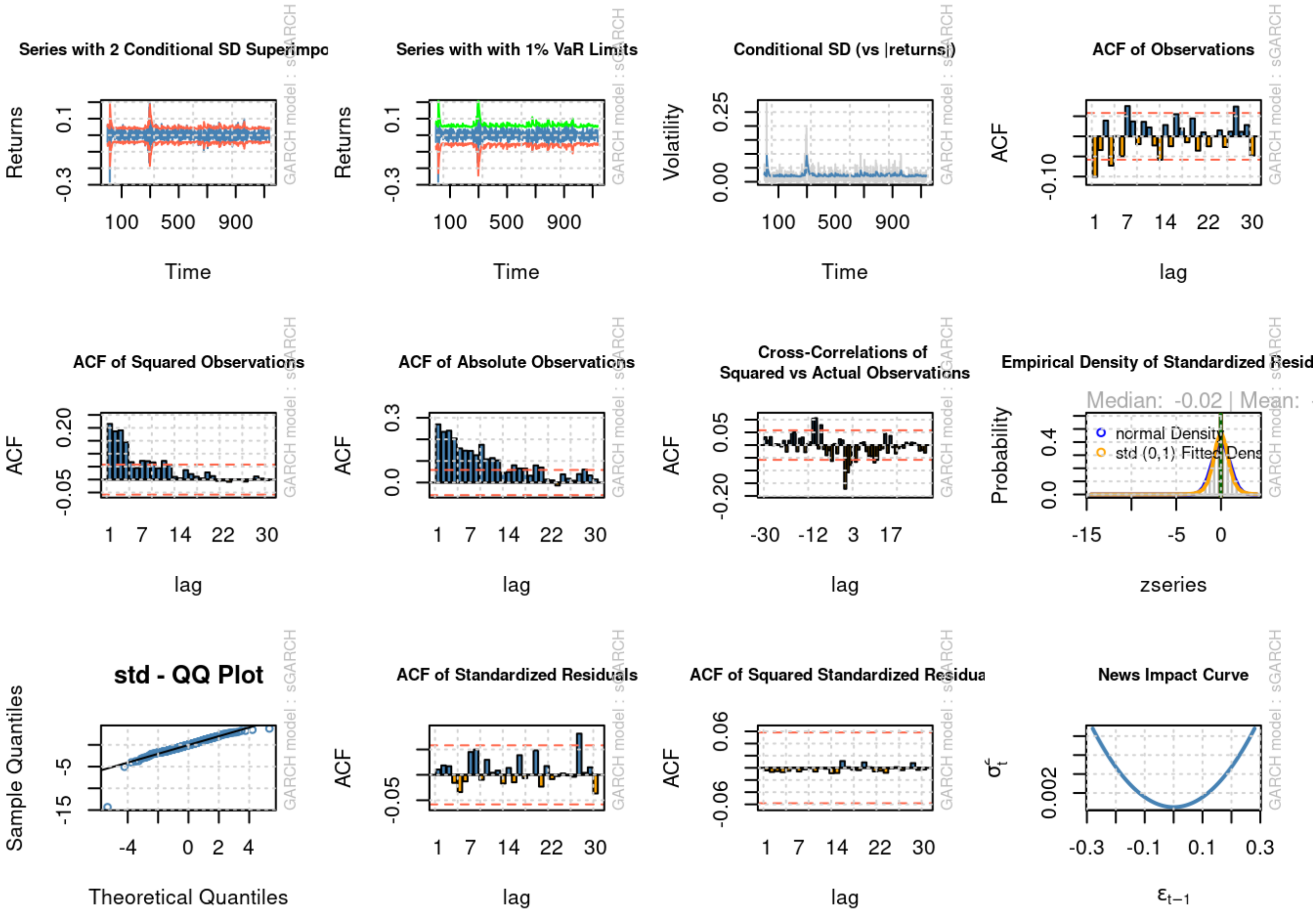
```
garch.fit.vale.normal
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : sGARCH(1,1)
## Mean Model   : ARFIMA(1,0,1)
## Distribution : norm
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000170    0.000637  0.26735 0.789201
## ar1     0.610005    0.343612  1.77527 0.075853
## ma1    -0.645701    0.330404 -1.95427 0.050669
## omega   0.000036    0.000016  2.26934 0.023248
## alpha1  0.049497    0.011640  4.25237 0.000021
## beta1   0.890526    0.032739 27.20103 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000170    0.000840  0.20276 0.839323
## ar1     0.610005    0.289281  2.10869 0.034971
## ma1    -0.645701    0.277037 -2.33074 0.019767
## omega   0.000036    0.000048  0.76215 0.445968
## alpha1  0.049497    0.048221  1.02646 0.304676
## beta1   0.890526    0.119050  7.48029 0.000000
##
## LogLikelihood : 2605.142
##
## Information Criteria
## ------------------------------------
##
## Akaike       -4.5760
## Bayes        -4.5494
## Shibata      -4.5760
## Hannan-Quinn -4.5659
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                   0.004179  0.9485
## Lag[2*(p+q)+(p+q)-1][5]  0.799785  1.0000
## Lag[4*(p+q)+(p+q)-1][9]  2.880865  0.9079
## d.o.f=2
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                    0.04009  0.8413
## Lag[2*(p+q)+(p+q)-1][5]   0.04955  0.9995
```

```
## Lag[4*(p+q)+(p+q)-1][9]   0.07334  1.0000
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]   0.01063 0.500 2.000  0.9179
## ARCH Lag[5]   0.01547 1.440 1.667  0.9991
## ARCH Lag[7]   0.03805 2.315 1.543  0.9999
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  1.8809
## Individual Statistics:
## mu     0.1492
## ar1    0.5450
## ma1    0.5115
## omega  0.2021
## alpha1 0.1488
## beta1  0.1658
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.49 1.68 2.12
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           1.0315 0.3025
## Negative Sign Bias  1.6740 0.0944   *
## Positive Sign Bias  0.1906 0.8489
## Joint Effect        2.8967 0.4078
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     63.01    1.279e-06
## 2    30     62.52    2.969e-04
## 3    40     80.27    1.112e-04
## 4    50     86.54    7.540e-04
##
##
## Elapsed time : 0.1135535
```

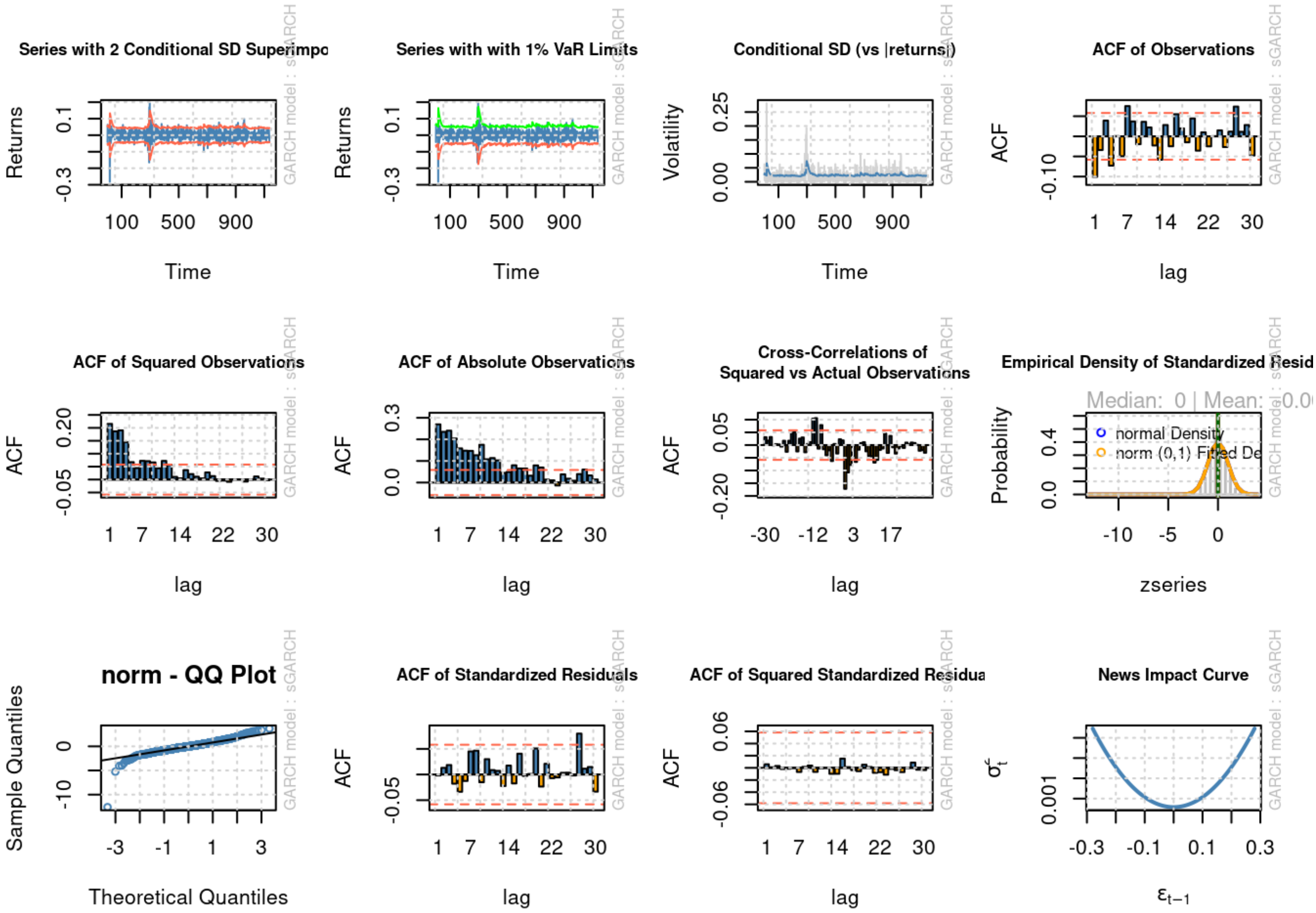
```
#infocriteria(garch.fit.vale.normal)
#infocriteria(garch.fit.vale.student)
options(repr.plot.width=15, repr.plot.height=15)
plot(garch.fit.vale.student, which="all")
```

```
##
## please wait...calculating quantiles...
```




```
plot(garch.fit.vale.normal, which="all")
```

```
##
## please wait...calculating quantiles...
```



EGARCH (Exponential GARCH)

Agora vamos estimar um modelo EGARCH(1, 1) para a mesma série de retornos:

```
#?ugarchspec
egarch.spec.student <- ugarchspec(variance.model=list(model="eGARCH",
                                                      garchOrder=c(1, 1)),
                                  mean.model=list(armaOrder=c(1, 1),
                                                  include.mean=TRUE),
                                  distribution.model="std")
egarch.spec.normal <- ugarchspec(variance.model=list(model="eGARCH",
                                                     garchOrder=c(1, 1)),
                                 mean.model=list(armaOrder=c(1, 1),
                                                 include.mean=TRUE),
                                 distribution.model="norm")

egarch.fit.vale.student <- ugarchfit(spec=egarch.spec.student,
                                     data=daily_returns_vale)

egarch.fit.vale.normal <- ugarchfit(spec=egarch.spec.normal,
                                    data=daily_returns_vale)

egarch.fit.vale.student
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : eGARCH(1,1)
## Mean Model   : ARFIMA(1,0,1)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error   t value Pr(>|t|)
## mu      0.000305    0.000543   0.56107 0.574753
## ar1     0.585531    0.043632  13.41988 0.000000
## ma1    -0.627989    0.042096 -14.91799 0.000000
## omega  -0.556195    0.189119  -2.94098 0.003272
## alpha1 -0.074640    0.026644  -2.80136 0.005089
## beta1   0.926352    0.024950  37.12790 0.000000
## gamma1  0.154126    0.031232   4.93487 0.000001
## shape   5.229301    0.780191   6.70259 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error   t value Pr(>|t|)
## mu      0.000305    0.000580   0.52468 0.599807
## ar1     0.585531    0.013036  44.91564 0.000000
## ma1    -0.627989    0.013346 -47.05568 0.000000
## omega  -0.556195    0.278914  -1.99414 0.046137
## alpha1 -0.074640    0.032250  -2.31439 0.020646
## beta1   0.926352    0.036339  25.49216 0.000000
## gamma1  0.154126    0.050332   3.06220 0.002197
## shape   5.229301    1.081064   4.83718 0.000001
##
## LogLikelihood : 2715.315
##
## Information Criteria
## ------------------------------------
##
## Akaike       -4.7664
## Bayes        -4.7309
## Shibata      -4.7665
## Hannan-Quinn -4.7530
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                      0.126  0.7226
## Lag[2*(p+q)+(p+q)-1][5]     1.241  0.9998
## Lag[4*(p+q)+(p+q)-1][9]     3.243  0.8504
## d.o.f=2
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
```

```
## ------------------------------------
##                          statistic p-value
## Lag[1]                    0.004729  0.9452
## Lag[2*(p+q)+(p+q)-1][5]  0.046540  0.9996
## Lag[4*(p+q)+(p+q)-1][9]  0.083633  1.0000
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]   0.01103 0.500 2.000  0.9164
## ARCH Lag[5]   0.04657 1.440 1.667  0.9955
## ARCH Lag[7]   0.06438 2.315 1.543  0.9998
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  2.2821
## Individual Statistics:
## mu     0.2706
## ar1    0.3954
## ma1    0.3488
## omega  0.1363
## alpha1 0.1485
## beta1  0.1308
## gamma1 0.3031
## shape  0.4737
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.89 2.11 2.59
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           1.0903 0.2758
## Negative Sign Bias  0.8171 0.4141
## Positive Sign Bias  0.2752 0.7832
## Joint Effect        1.3735 0.7118
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     15.48      0.69171
## 2    30     42.61      0.04945
## 3    40     37.03      0.56011
## 4    50     45.87      0.60095
##
##
## Elapsed time : 0.2451379
```

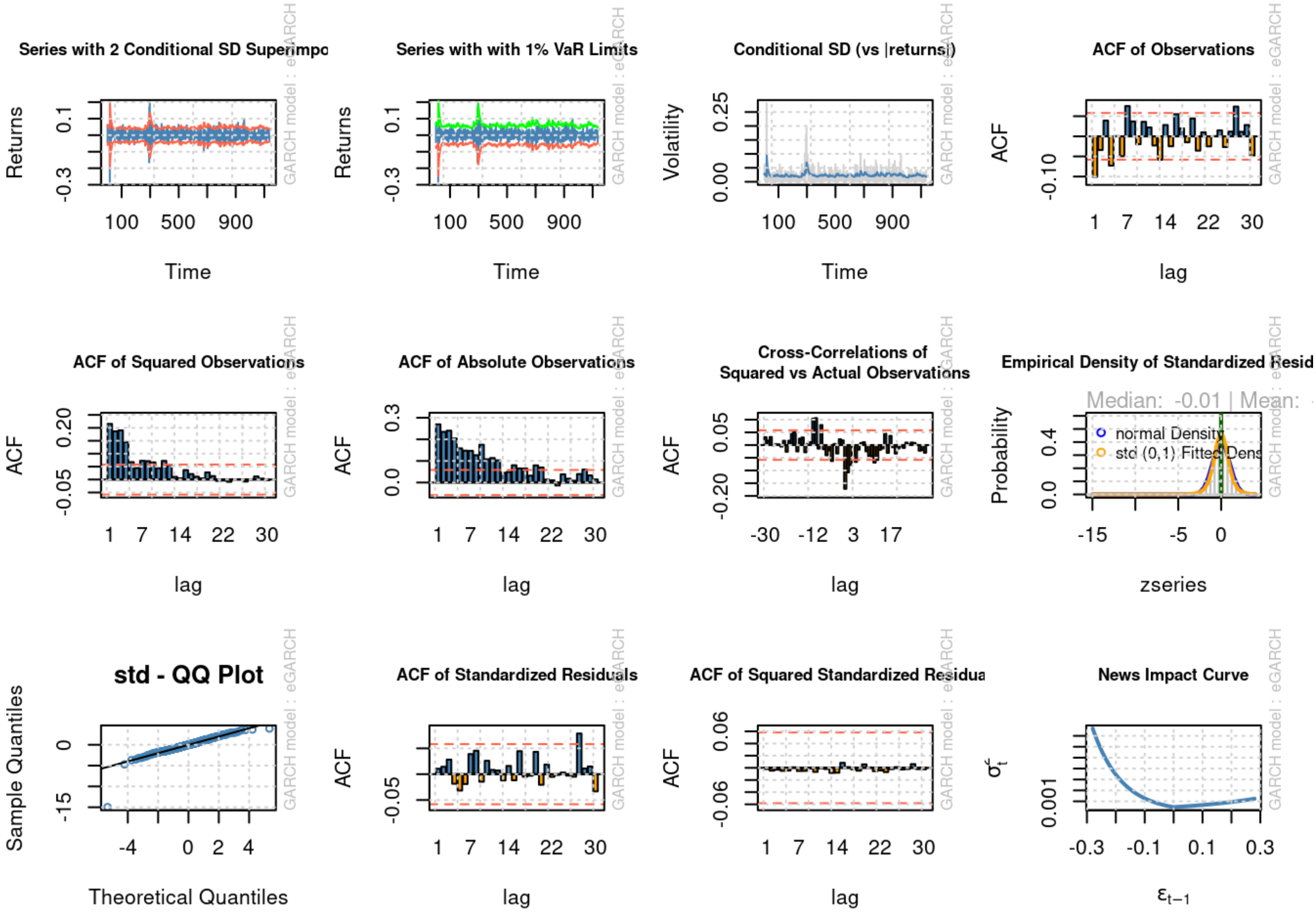
```
egarch.fit.vale.normal
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : eGARCH(1,1)
## Mean Model   : ARFIMA(1,0,1)
## Distribution : norm
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error   t value Pr(>|t|)
## mu      -0.00012    0.000671   -0.1786 0.858252
## ar1      0.67105    0.033818   19.8428 0.000000
## ma1     -0.70271    0.032188  -21.8312 0.000000
## omega   -0.24657    0.001904 -129.5082 0.000000
## alpha1  -0.02485    0.009842   -2.5247 0.011578
## beta1    0.96597    0.000877 1101.2948 0.000000
## gamma1   0.10437    0.008632   12.0906 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error   t value Pr(>|t|)
## mu      -0.00012    0.000873  -0.13718  0.89089
## ar1      0.67105    0.013876  48.36205  0.00000
## ma1     -0.70271    0.016219 -43.32565  0.00000
## omega   -0.24657    0.020645 -11.94335  0.00000
## alpha1  -0.02485    0.039845  -0.62366  0.53285
## beta1    0.96597    0.002272 425.17142  0.00000
## gamma1   0.10437    0.015429   6.76430  0.00000
##
## LogLikelihood : 2606.869
##
## Information Criteria
## ------------------------------------
##
## Akaike       -4.5772
## Bayes        -4.5462
## Shibata      -4.5773
## Hannan-Quinn -4.5655
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                    0.05975  0.8069
## Lag[2*(p+q)+(p+q)-1][5]   0.90935  1.0000
## Lag[4*(p+q)+(p+q)-1][9]   2.97274  0.8947
## d.o.f=2
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
```

```
## Lag[1]                    0.08376  0.7723
## Lag[2*(p+q)+(p+q)-1][5]   0.23415  0.9899
## Lag[4*(p+q)+(p+q)-1][9]   0.26442  0.9998
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]   0.04135 0.500 2.000  0.8389
## ARCH Lag[5]   0.04185 1.440 1.667  0.9962
## ARCH Lag[7]   0.04899 2.315 1.543  0.9999
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  1.7098
## Individual Statistics:
## mu     0.1806
## ar1    0.4022
## ma1    0.3859
## omega  0.2511
## alpha1 0.2119
## beta1  0.2535
## gamma1 0.2199
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.69 1.9 2.35
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value    prob sig
## Sign Bias          1.09801 0.27243
## Negative Sign Bias 1.84678 0.06504   *
## Positive Sign Bias 0.02883 0.97700
## Joint Effect       3.52288 0.31781
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     60.30    3.465e-06
## 2    30     67.38    6.861e-05
## 3    40     98.01    5.514e-07
## 4    50    100.71    1.948e-05
##
##
## Elapsed time : 0.1434305
```

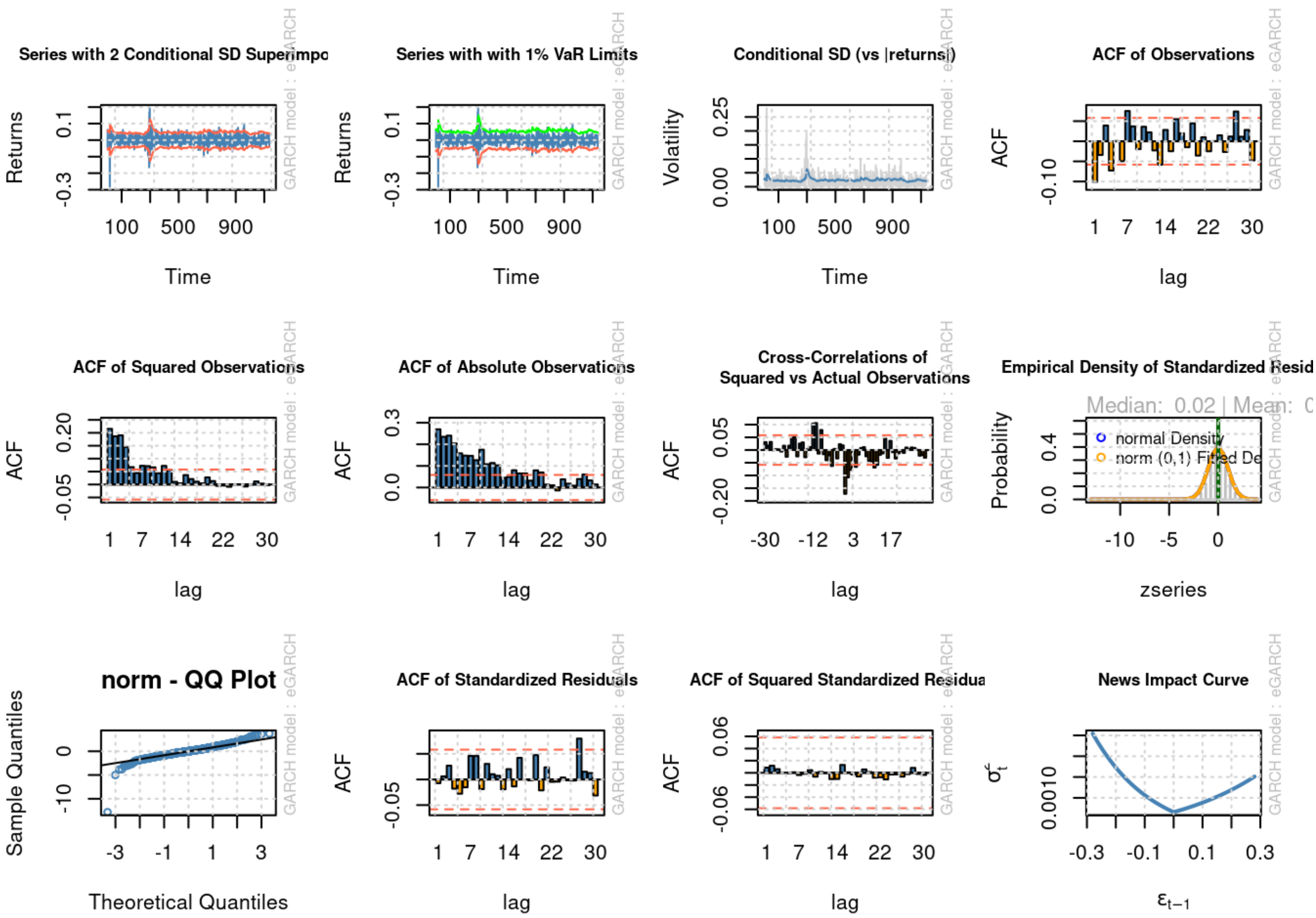
```
#infocriteria(egarch.fit.vale.normal)
#infocriteria(egarch.fit.vale.student)
options(repr.plot.width=15, repr.plot.height=15)
plot(egarch.fit.vale.student, which="all")
```

```
##
## please wait...calculating quantiles...
```




```
plot(egarch.fit.vale.normal, which="all")
```

```
##
## please wait...calculating quantiles...
```



GRJ - GARCH

Agora vamos estimar um modelo GJR(1, 1) para a mesma série de retornos:

```
#https://search.r-project.org/CRAN/refmans/rugarch/html/ugarchspec-methods.html
gjr_garch.spec.student <- ugarchspec(variance.model=list(model="gjrGARCH",
                                                          garchOrder=c(1, 1)),
                                     mean.model=list(armaOrder=c(1, 1),
                                                     include.mean=TRUE),
                                  distribution.model="std")
gjr_garch.spec.normal <- ugarchspec(variance.model=list(model="gjrGARCH",
                                                         garchOrder=c(1, 1)),
                                    mean.model=list(armaOrder=c(1, 1),
                                                    include.mean=TRUE),
                                    distribution.model="norm")

gjr_garch.fit.vale.student <- ugarchfit(spec=gjr_garch.spec.student,
                                        data=daily_returns_vale)

gjr_garch.fit.vale.normal <- ugarchfit(spec=gjr_garch.spec.normal,
                                       data=daily_returns_vale)

gjr_garch.fit.vale.student
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : gjrGARCH(1,1)
## Mean Model   : ARFIMA(1,0,1)
## Distribution : std
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000284    0.000545  0.52207 0.601619
## ar1     0.608930    0.218656  2.78487 0.005355
## ma1    -0.650897    0.208302 -3.12478 0.001779
## omega   0.000080    0.000024  3.38724 0.000706
## alpha1  0.037005    0.028160  1.31413 0.188804
## beta1   0.745671    0.060236 12.37906 0.000000
## gamma1  0.160741    0.062026  2.59151 0.009556
## shape   5.408759    0.808587  6.68915 0.000000
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000284    0.000579  0.49159 0.623007
## ar1     0.608930    0.140526  4.33321 0.000015
## ma1    -0.650897    0.133325 -4.88204 0.000001
## omega   0.000080    0.000025  3.14669 0.001651
## alpha1  0.037005    0.028117  1.31610 0.188140
## beta1   0.745671    0.067246 11.08868 0.000000
## gamma1  0.160741    0.078331  2.05206 0.040163
## shape   5.408759    1.091120  4.95707 0.000001
##
## LogLikelihood : 2716.723
##
## Information Criteria
## ------------------------------------
##
## Akaike       -4.7689
## Bayes        -4.7334
## Shibata      -4.7690
## Hannan-Quinn -4.7555
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                     0.3006  0.5835
## Lag[2*(p+q)+(p+q)-1][5]    1.3976  0.9992
## Lag[4*(p+q)+(p+q)-1][9]    3.4091  0.8194
## d.o.f=2
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
```

```
## ------------------------------------
##                          statistic p-value
## Lag[1]                      0.03556  0.8504
## Lag[2*(p+q)+(p+q)-1][5]     0.15472  0.9955
## Lag[4*(p+q)+(p+q)-1][9]     0.22065  0.9999
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]    0.0506 0.500 2.000  0.8220
## ARCH Lag[5]    0.1137 1.440 1.667  0.9842
## ARCH Lag[7]    0.1404 2.315 1.543  0.9987
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  2.3625
## Individual Statistics:
## mu     0.2865
## ar1    0.3715
## ma1    0.3228
## omega  0.1798
## alpha1 0.3211
## beta1  0.2676
## gamma1 0.2977
## shape  0.5389
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.89 2.11 2.59
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias           0.9648 0.3349
## Negative Sign Bias  0.3262 0.7443
## Positive Sign Bias  0.4071 0.6840
## Joint Effect        0.9644 0.8099
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     18.72       0.4750
## 2    30     31.68       0.3343
## 3    40     37.45       0.5406
## 4    50     48.60       0.4895
##
##
## Elapsed time : 0.4241905
```

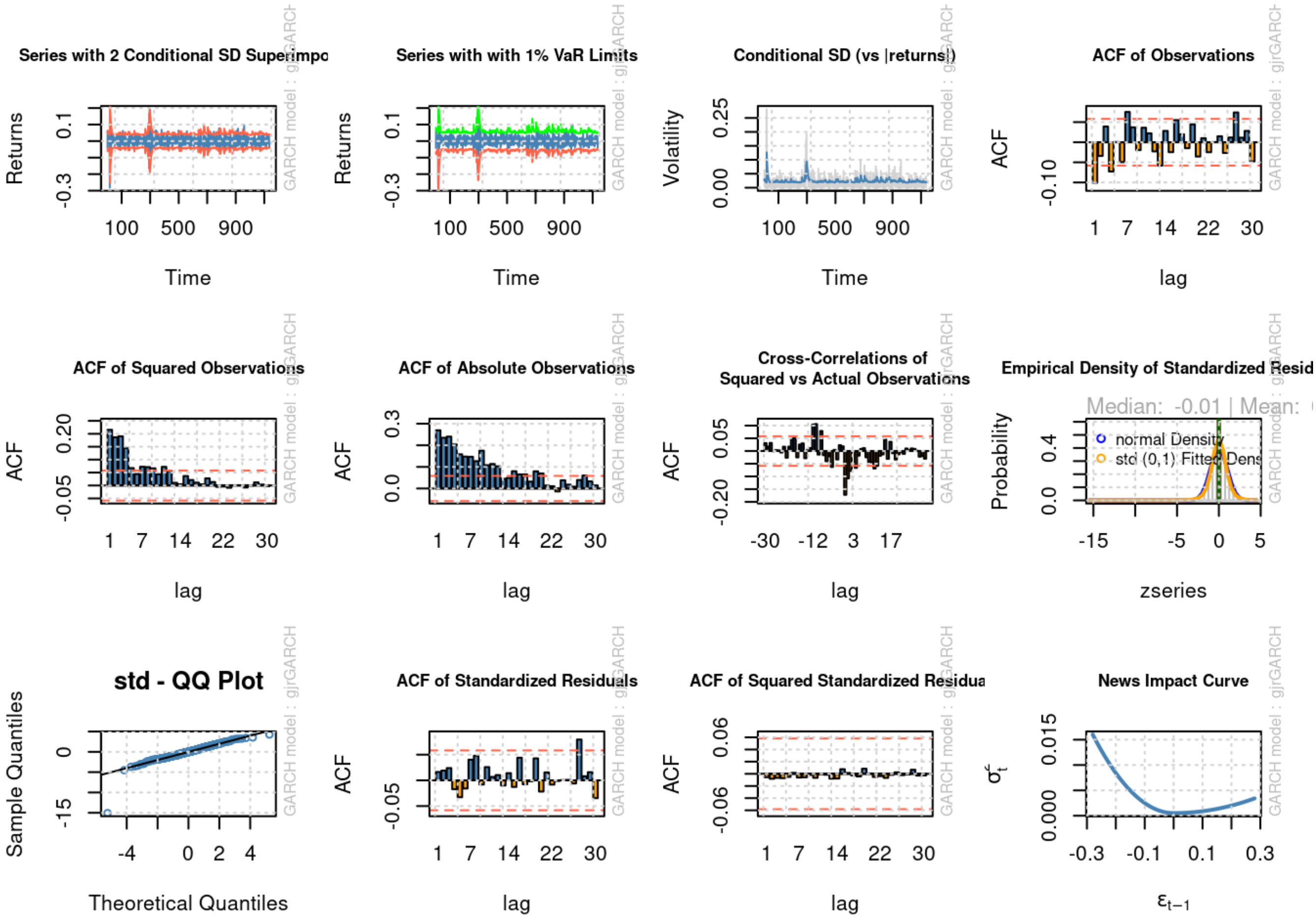
```
gjr_garch.fit.vale.normal
```

```
##
## *---------------------------------*
## *          GARCH Model Fit        *
## *---------------------------------*
##
## Conditional Variance Dynamics
## -----------------------------------
## GARCH Model  : gjrGARCH(1,1)
## Mean Model   : ARFIMA(1,0,1)
## Distribution : norm
##
## Optimal Parameters
## ------------------------------------
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000075    0.000640  0.11696 0.906894
## ar1     0.734034    0.225233  3.25900 0.001118
## ma1    -0.763016    0.213002 -3.58220 0.000341
## omega   0.000161    0.000074  2.18110 0.029176
## alpha1  0.011273    0.029757  0.37882 0.704821
## beta1   0.651912    0.151948  4.29036 0.000018
## gamma1  0.167394    0.077494  2.16009 0.030766
##
## Robust Standard Errors:
##         Estimate  Std. Error  t value Pr(>|t|)
## mu      0.000075    0.000696  0.10756 0.914341
## ar1     0.734034    0.204238  3.59401 0.000326
## ma1    -0.763016    0.204093 -3.73858 0.000185
## omega   0.000161    0.000213  0.75387 0.450925
## alpha1  0.011273    0.038657  0.29161 0.770588
## beta1   0.651912    0.399272  1.63275 0.102521
## gamma1  0.167394    0.207350  0.80730 0.419494
##
## LogLikelihood : 2608.582
##
## Information Criteria
## ------------------------------------
##
## Akaike       -4.5802
## Bayes        -4.5492
## Shibata      -4.5803
## Hannan-Quinn -4.5685
##
## Weighted Ljung-Box Test on Standardized Residuals
## ------------------------------------
##                         statistic p-value
## Lag[1]                    0.03582  0.8499
## Lag[2*(p+q)+(p+q)-1][5]   0.83674  1.0000
## Lag[4*(p+q)+(p+q)-1][9]   3.17762  0.8619
## d.o.f=2
## H0 : No serial correlation
##
## Weighted Ljung-Box Test on Standardized Squared Residuals
## ------------------------------------
##                         statistic p-value
```

```
## Lag[1]                    0.003614  0.9521
## Lag[2*(p+q)+(p+q)-1][5]  0.052858  0.9995
## Lag[4*(p+q)+(p+q)-1][9]  0.071041  1.0000
## d.o.f=2
##
## Weighted ARCH LM Tests
## ------------------------------------
##             Statistic Shape Scale P-Value
## ARCH Lag[3]  0.007498 0.500 2.000  0.9310
## ARCH Lag[5]  0.025949 1.440 1.667  0.9981
## ARCH Lag[7]  0.033925 2.315 1.543  0.9999
##
## Nyblom stability test
## ------------------------------------
## Joint Statistic:  1.7393
## Individual Statistics:
## mu     0.2307
## ar1    0.3445
## ma1    0.3312
## omega  0.2209
## alpha1 0.2057
## beta1  0.2095
## gamma1 0.3954
##
## Asymptotic Critical Values (10% 5% 1%)
## Joint Statistic:          1.69 1.9 2.35
## Individual Statistic:     0.35 0.47 0.75
##
## Sign Bias Test
## ------------------------------------
##                    t-value   prob sig
## Sign Bias          0.82333 0.4105
## Negative Sign Bias 0.79740 0.4254
## Positive Sign Bias 0.07324 0.9416
## Joint Effect       1.09455 0.7784
##
##
## Adjusted Pearson Goodness-of-Fit Test:
## ------------------------------------
##   group statistic p-value(g-1)
## 1    20     71.85    4.515e-08
## 2    30     77.26    2.898e-06
## 3    40     86.96    1.630e-05
## 4    50     97.27    4.934e-05
##
##
## Elapsed time : 0.4410481
```

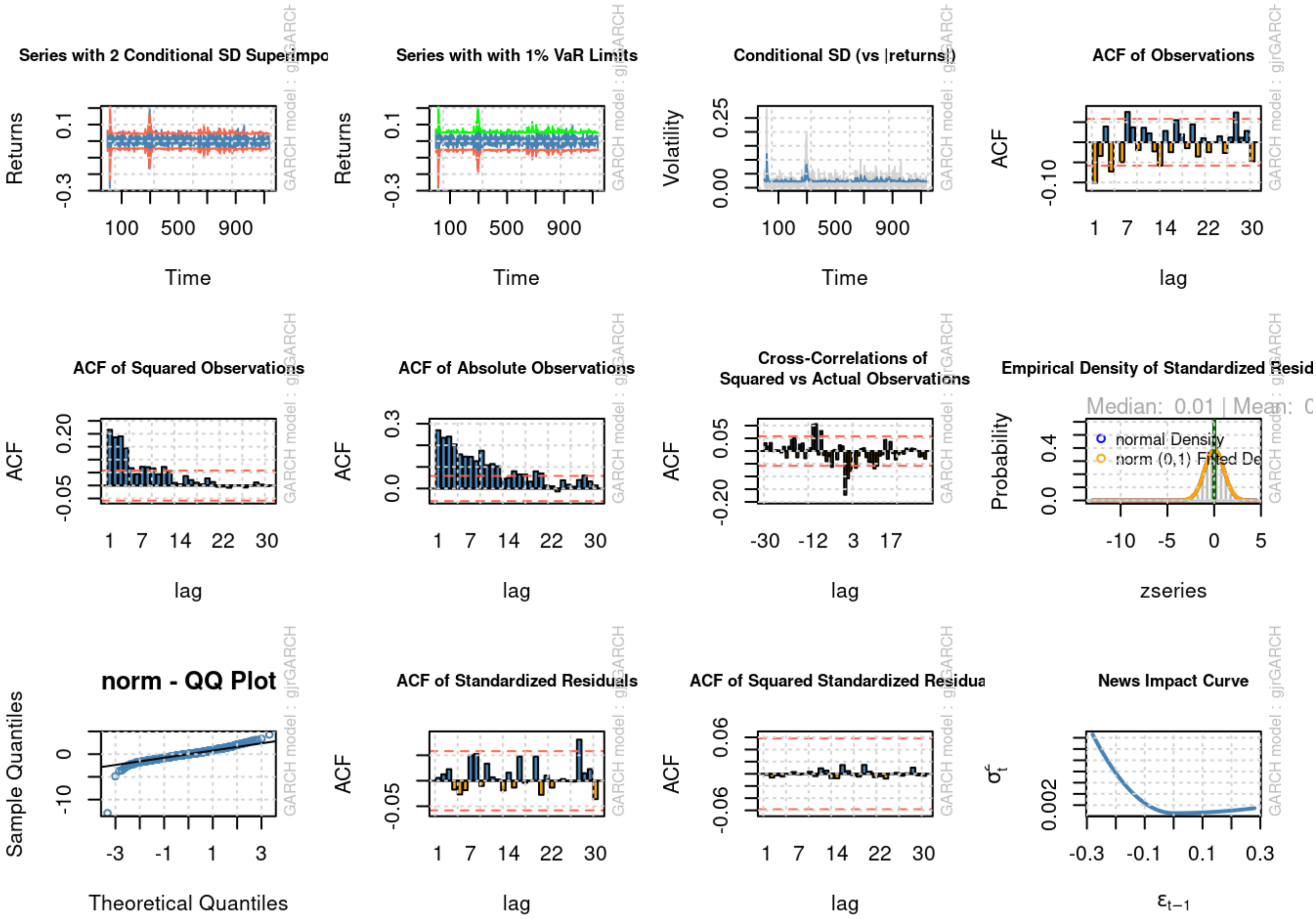
```
#infocriteria(gjr_garch.fit.vale.normal)
#infocriteria(gjr_garch.fit.vale.student)
options(repr.plot.width=15, repr.plot.height=15)
plot(gjr_garch.fit.vale.student, which="all")
```

```
##
## please wait...calculating quantiles...
```




```
plot(gjr_garch.fit.vale.normal, which="all")
```

```
##
## please wait...calculating quantiles...
```



Coeficientes de persistência e *half-life*

Agora vamos calcular os coeficientes de persistência e half-life para cada um dos modelos ajustados acima.

Primeiramente, calculamos os coeficientes de persistência:

```
paste("garch.normal:", persistence(garch.fit.vale.normal))
```

```
## [1] "garch.normal: 0.940023085058988"
```

```
paste("garch.student:", persistence(garch.fit.vale.student))
```

```
## [1] "garch.student: 0.875127179811051"
```

```
paste("egarch.normal:", persistence(egarch.fit.vale.normal))
```

```
## [1] "egarch.normal: 0.965965583402948"
```

```
paste("egarch.student:", persistence(egarch.fit.vale.student))
```

```
## [1] "egarch.student: 0.926351723034298"
```

```
paste("gjr_garch.normal:", persistence(gjr_garch.fit.vale.normal))
```

```
## [1] "gjr_garch.normal: 0.746881897221491"
```

```
paste("gjr_garch.student:", persistence(gjr_garch.fit.vale.student))
```

```
## [1] "gjr_garch.student: 0.863046388509446"
```

Os valores acima indicam que haverá maior pesistência dos choques no caso de usarmos o modelo EGARCH(1, 1) com distribuição Normal (`egarch.normal`). Ou seja, escolhendo este modelo haverá uma maior persistência da volatilidade.

Por outro lado, escolhendo o GJR(1, 1) com distribuição Normal (`gjr_garch.normal`) haverá uma menor persistência da volatilidade.

Calculamos os coeficientes de *half-life* com os códigos abaixo:

```
paste("garch.normal:", halflife(garch.fit.vale.normal))
```

```
## [1] "garch.normal: 11.2067535303504"
```

```
paste("garch.student:", halflife(garch.fit.vale.student))
```

```
## [1] "garch.student: 5.19654908031971"
```

```
paste("egarch.normal:", halflife(egarch.fit.vale.normal))
```

```
## [1] "egarch.normal: 20.0174924840579"
```

```
paste("egarch.student:", halflife(egarch.fit.vale.student))
```

```
## [1] "egarch.student: 9.06059514996847"
```

```
paste("gjr_garch.normal:", halflife(gjr_garch.fit.vale.normal))
```

```
## [1] "gjr_garch.normal: 2.37502632831976"
```

```
paste("gjr_garch.student:", halflife(gjr_garch.fit.vale.student))
```

```
## [1] "gjr_garch.student: 4.70610406178515"
```

Pelos valores acima, notamos que com a escolha do modelo GRJ(1, 1) com distribuição Normal (`gjr_garch.normal`), modelo correspondente ao menor valor de *half-time*, teremos uma menor quantidade de dias (cerca de 2 dias) para o choque ser dissipado pela metade.

Por outro lado, escolhendo o EGARCH(1, 1) com distribuição Normal (`egarch.normal`), levará mais dias para que um choque se dissipe pela metade. De fato, os cálculos indicam que demorará cerca de 11 dias para que isso ocorra.

Critério de Informação

Agora iremos calcular os critérios de informação de Akaike, Bayesiano (Schwarz), Shibata e Hannan-Quinn para dos nossos modelos:

```
print("garch.fit.vale.normal:")
```

```
## [1] "garch.fit.vale.normal:"
```

```
infocriteria(garch.fit.vale.normal)
```

```
##
## Akaike       -4.575955
## Bayes        -4.549360
## Shibata      -4.576010
## Hannan-Quinn -4.565910
```

```
print("garch.fit.vale.student:")
```

```
## [1] "garch.fit.vale.student:"
```

```
infocriteria(garch.fit.vale.student)
```

```
##
## Akaike       -4.761855
## Bayes        -4.730828
## Shibata      -4.761930
## Hannan-Quinn -4.750136
```

```
print("egarch.fit.vale.normal:")
```

```
## [1] "egarch.fit.vale.normal:"
```

```
infocriteria(egarch.fit.vale.normal)
```

```
##
## Akaike       -4.577233
## Bayes        -4.546206
## Shibata      -4.577309
## Hannan-Quinn -4.565514
```

```
print("egarch.fit.vale.student:")
```

```
## [1] "egarch.fit.vale.student:"
```

```
infocriteria(egarch.fit.vale.student)
```

```
##
## Akaike       -4.766400
## Bayes        -4.730940
## Shibata      -4.766498
## Hannan-Quinn -4.753006
```

```
print("gjr_garch.fit.vale.normal:")
```

```
## [1] "gjr_garch.fit.vale.normal:"
```

```
infocriteria(gjr_garch.fit.vale.normal)
```

```
##
## Akaike       -4.580250
## Bayes        -4.549223
## Shibata      -4.580325
## Hannan-Quinn -4.568531
```

```
print("gjr_garch.fit.vale.student:")
```

```
## [1] "gjr_garch.fit.vale.student:"
```

```
infocriteria(gjr_garch.fit.vale.student)
```

```
##
## Akaike       -4.768878
## Bayes        -4.733419
## Shibata      -4.768977
## Hannan-Quinn -4.755485
```

Critério de Informação - Resumo

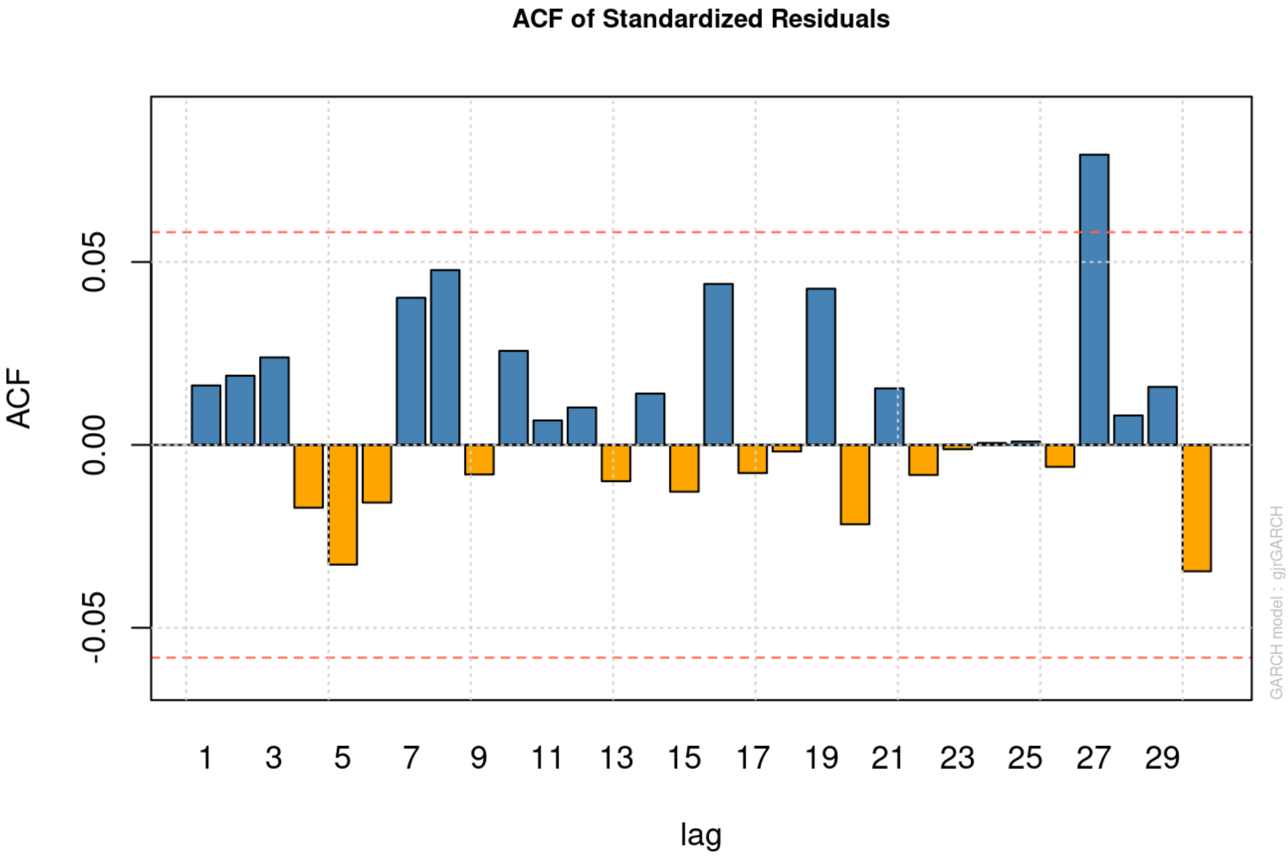
Modelo	Akaike	Bayes	Shibata	Hannan-Quinn
garch.fit.vale.normal	-4.575955	-4.549360	-4.576010	-4.565910
garch.fit.vale.student	-4.761855	-4.730828	-4.761930	-4.750136
egarch.fit.vale.normal	-4.577233	-4.546206	-4.577309	-4.565514
egarch.fit.vale.student	-4.766400	-4.730940	-4.766498	-4.753006
gjr_garch.fit.vale.normal	-4.580250	-4.549223	-4.580325	-4.568531
gjr_garch.fit.vale.student	**-4.768878**	**-4.733419**	**-4.768977**	**-4.755485**

Como na tabela acima o modelo GJR(1,1) com t-Student obteve os menores valores para cada um dos critérios de informação apresentados, escolhemos este modelo (`gjr_garch.fit.vale.student`) para prosseguirmos as análises.

Resíduos

Vejamos a ACF dos resíduos:

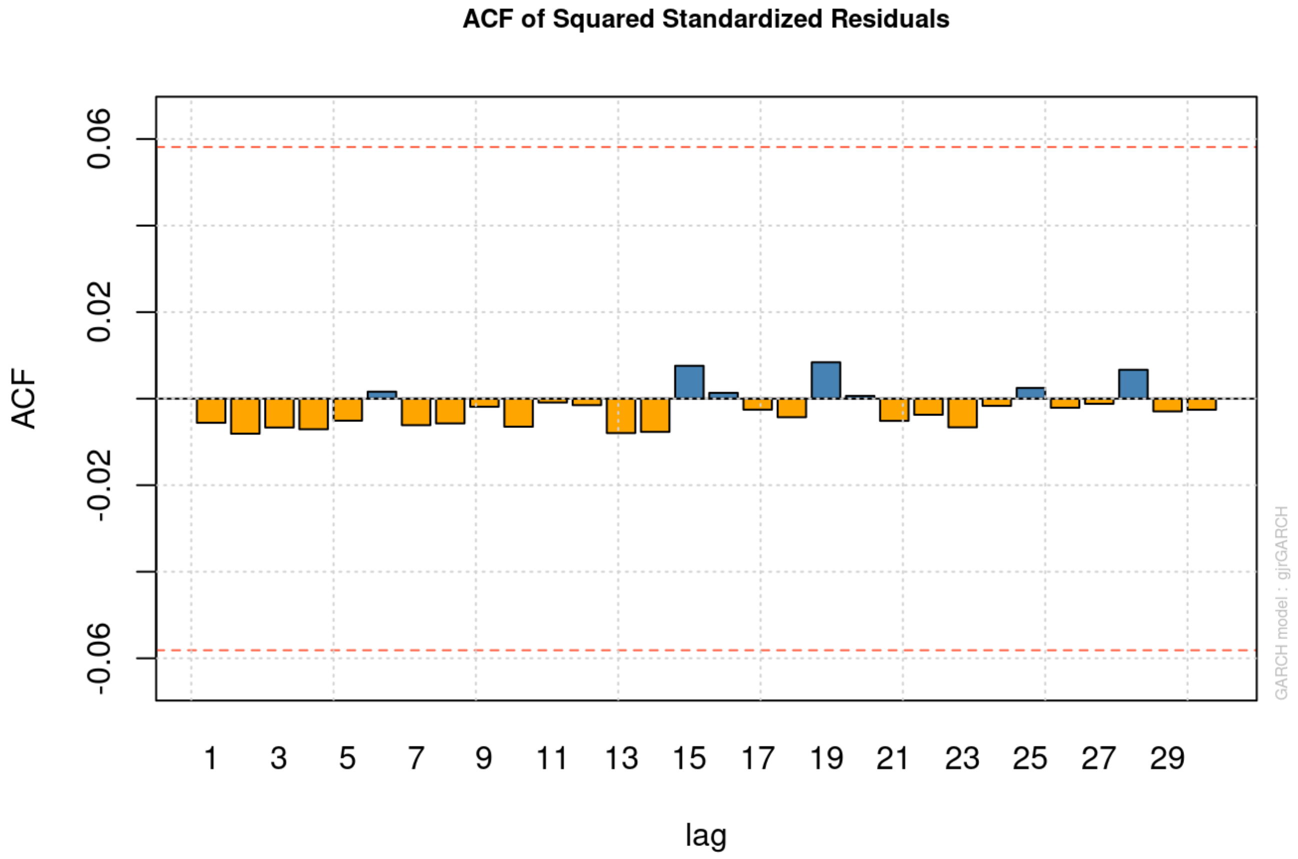
```
options(repr.plot.width=25, repr.plot.height=15)
#plot(gjr_garch.fit.vale.student, which="all")
plot(gjr_garch.fit.vale.student, which=10)
```



Notemos que para `lag = 29` temos uma autocorrelação significativa para a série analisada. Para os demais lags os valores exibidos no gráfico não são significativos.

Vejamos os resíduos ao quadrado:

```
options(repr.plot.width=25, repr.plot.height=15)
#plot(gjr_garch.fit.vale.student, which="all")
plot(gjr_garch.fit.vale.student, which=11)
```

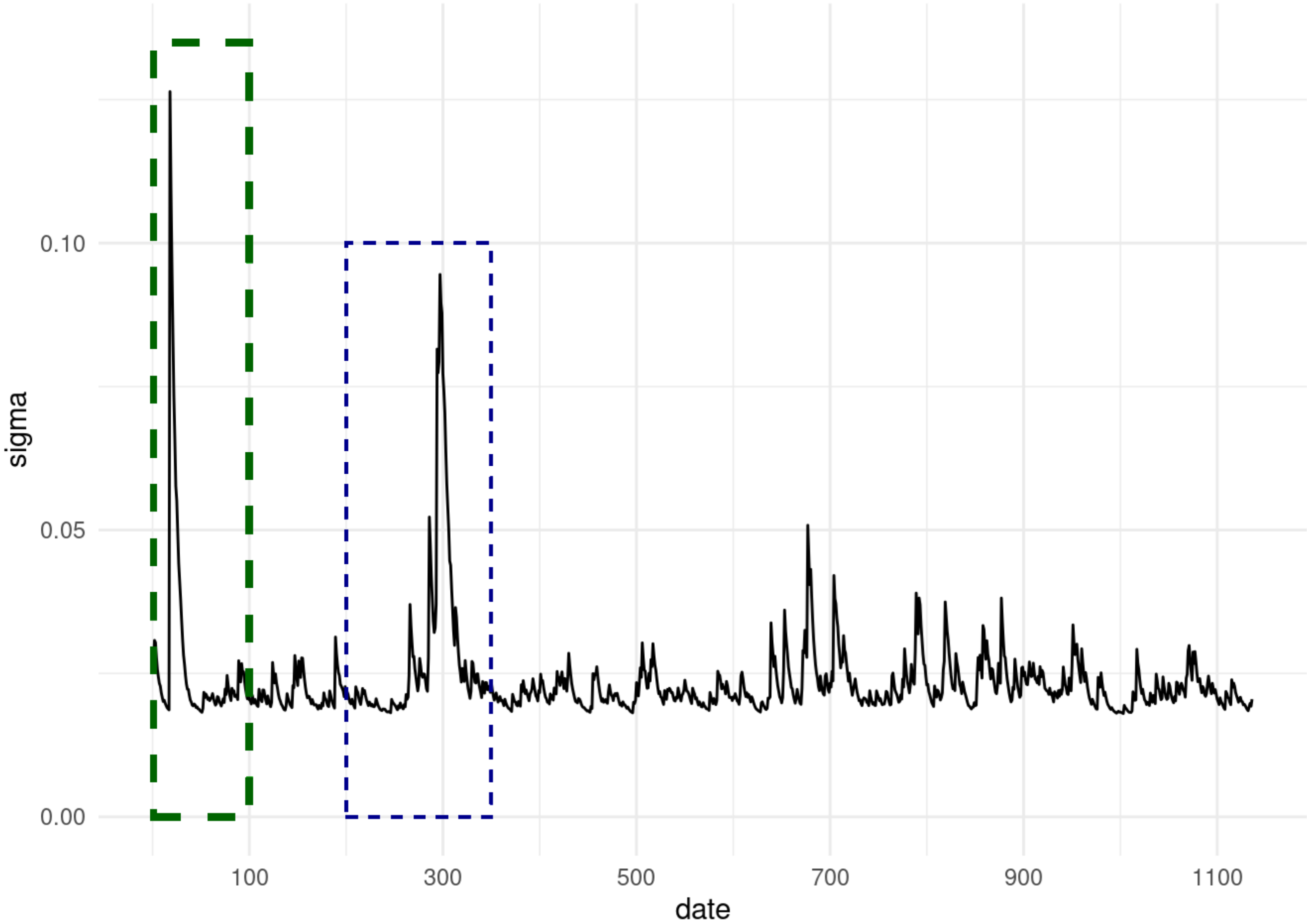


Para os resíduos ao quadrado não notamos autocorrelação significativa para nenhum dos lags exibidos no gráfico.

Volatilidade condicional

A seguir temos o gráfico da volatilidade condicional para o modelo selecionado acima:

```r
library(tbl2xts)
sigma <- sigma(gjr_garch.fit.vale.student) %>% xts_tbl()
colnames(sigma) <- c("date", "sigma")
sigma <- sigma %>% mutate(date=as.Date(date))
#sigma$date
ggplot(sigma) + geom_line(aes(x=date , y=sigma)) +
  geom_rect(aes(xmin=as.Date("1-01-01"), xmax=as.Date("100-01-01"),
              ymin=0, ymax=0.135),
            fill="transparent", linetype=2, color="darkgreen", size=1.2) +
  geom_rect(aes(xmin=as.Date("200-01-01"), xmax=as.Date("350-01-01"),
              ymin=0, ymax=0.100),
            fill="transparent", linetype=2, color="darkblue", size=0.5) +
  theme_minimal()
```



Nos trechos destacados no gráfico acima temos um aumento na volatilidade. Os períodos correspondem ao início de 2019 e ao início da pandemia.

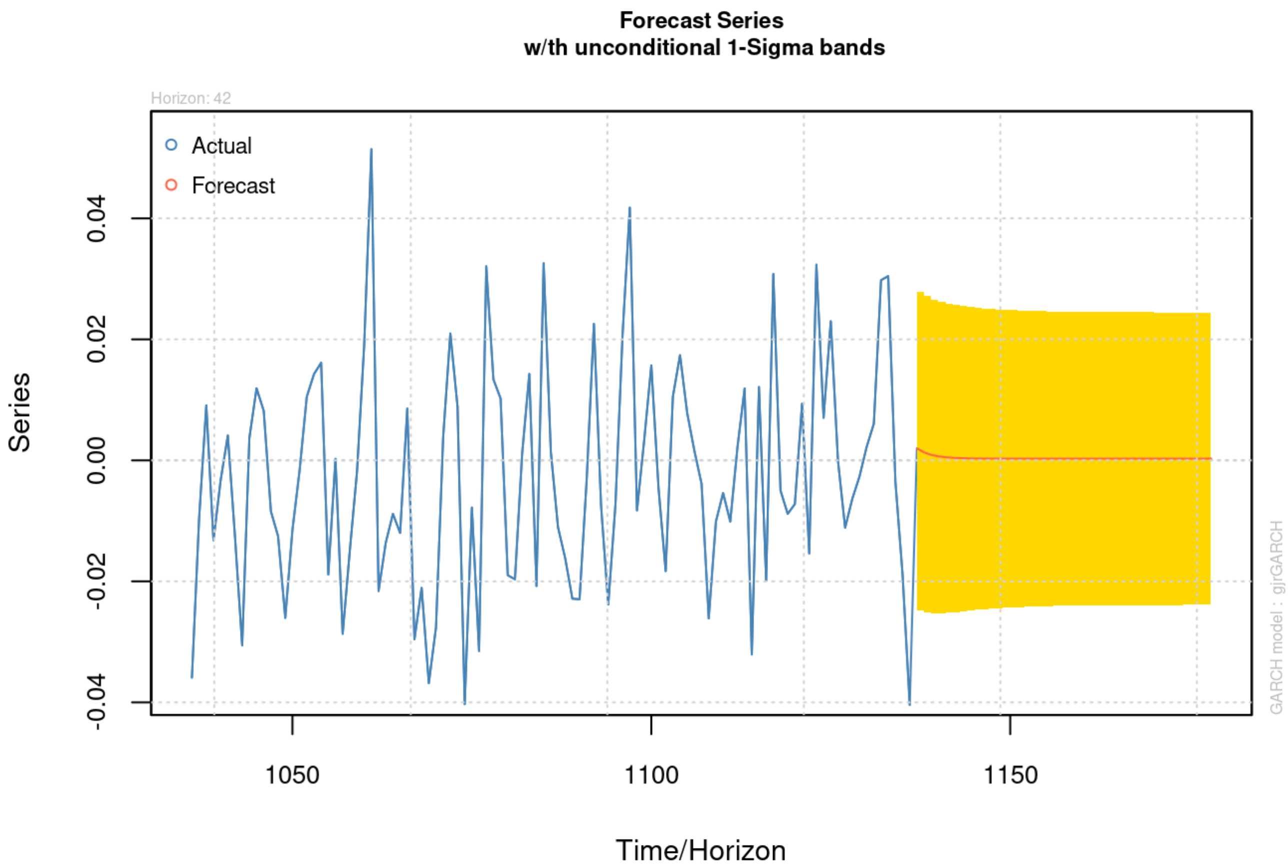
Previsões para volatilidade condicional

Usando o modelo GJR(1, 1) com t-Student `gjr_garch.fit.vale.student` estimado acima, realizamos a predição para volatilidade condicional 42 passos a frente:

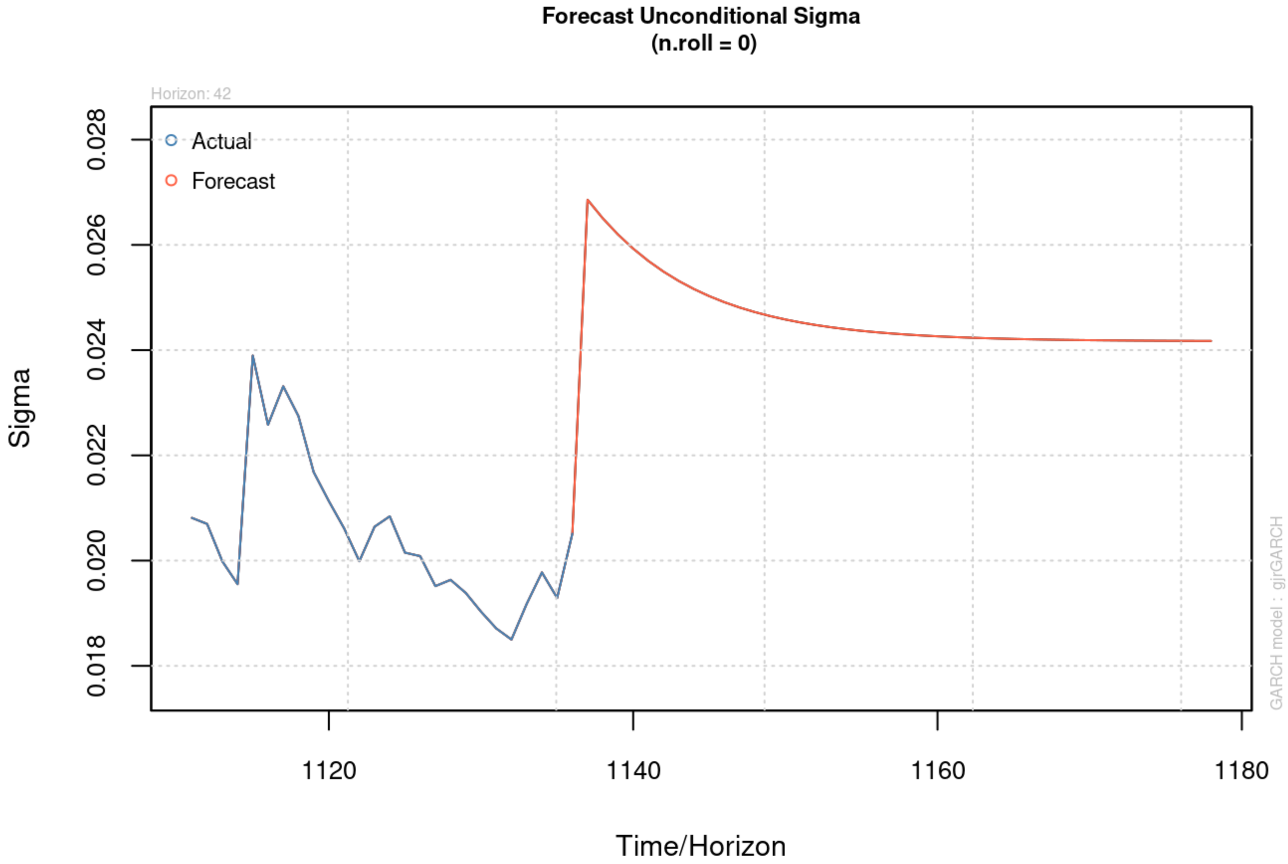
```
garchf.1 <- ugarchforecast(gjr_garch.fit.vale.student, n.ahead=42)
garchf.1
```

```
##
## *---------------------------------*
## *       GARCH Model Forecast      *
## *---------------------------------*
## Model: gjrGARCH
## Horizon: 42
## Roll Steps: 0
## Out of Sample: 0
##
## 0-roll forecast [T0=1136-01-01]:
##         Series   Sigma
## T+1  0.0019883 0.02686
## T+2  0.0013220 0.02651
## T+3  0.0009163 0.02620
## T+4  0.0006692 0.02593
## T+5  0.0005187 0.02569
## T+6  0.0004271 0.02549
## T+7  0.0003714 0.02531
## T+8  0.0003374 0.02516
## T+9  0.0003167 0.02503
## T+10 0.0003041 0.02491
## T+11 0.0002964 0.02481
## T+12 0.0002918 0.02472
## T+13 0.0002889 0.02465
## T+14 0.0002872 0.02458
## T+15 0.0002861 0.02453
## T+16 0.0002855 0.02448
## T+17 0.0002851 0.02443
## T+18 0.0002849 0.02440
## T+19 0.0002847 0.02437
## T+20 0.0002846 0.02434
## T+21 0.0002846 0.02432
## T+22 0.0002845 0.02429
## T+23 0.0002845 0.02428
## T+24 0.0002845 0.02426
## T+25 0.0002845 0.02425
## T+26 0.0002845 0.02424
## T+27 0.0002845 0.02423
## T+28 0.0002845 0.02422
## T+29 0.0002845 0.02421
## T+30 0.0002845 0.02421
## T+31 0.0002845 0.02420
## T+32 0.0002845 0.02420
## T+33 0.0002845 0.02419
## T+34 0.0002845 0.02419
## T+35 0.0002845 0.02419
## T+36 0.0002845 0.02418
## T+37 0.0002845 0.02418
## T+38 0.0002845 0.02418
## T+39 0.0002845 0.02418
## T+40 0.0002845 0.02418
## T+41 0.0002845 0.02417
## T+42 0.0002845 0.02417
```

```
plot(garchf.1, which=1)
```




```
plot(garchf.1, which=3)
```



Pelos gráficos notamos que para os próximos 42 dias haverá um aumento na volatilidade, que tenderá a se estabilizar em um valor mais alto que o dos últimos dias. Por sua vez, a média continuará por volta de zero.

Referências

- Materiais das aulas (profa. Andreza Palma)
- CAP. 4 do livro "TSAY, Ruey S. *An introduction to analysis of financial data with R.* John Wiley & Sons, 2014."