Homework 3 (Lista ARMA prática)

Marcelo Santos Carielo

ago/2023

Questão 1

1. Usando todos os passos vistos no módulo sobre ARMA, encontre o melhor modelo para os retornos diários do índice Ibovespa. Utilize o período de 2021 - presente. Você pode usar a função auto.arima, mas deve fazer a identificação do modelo usando as FAC e FACP, diagnóstico, etc. Para recordar, os passos são os seguintes: (a) Fazer uma análise visual da série, verificando os fatos estilizados. (b) Fazer a análise da FAC e da FACP. Objetivo é entender as autocorrelações da série de dados e nos ajudar a determinar qual o modelo e a ordem de defasagem escolher (identificação) (c) Estimar o modelo baseado na defasagem escolhida pelos critérios FAC e FACP. Qual a estatística-t de cada parâmetro? Qual o valor dos critérios de informação (BIC e AIC)? (estimação) (d) Diagnóstico dos resíduos. Verificar se os resíduos se comportam como ruído branco. (diagnóstico)`

Resposta 1

Para este exercício, usaremos a série de retornos do IBOVESPA de `01/01/2019` até o dia de hoje (`2023-07-06`). O código abaixo coleta esses dados do Yahoo Finance.

```
library(BatchGetSymbols)

# define datas de início e fim
date_init <- "2019-01-01"
date_end <- "2023-07-06"
#date_end <- Sys.Date()

# coleta dados do IBOVESPA
tickers <- c("^BVSP")
assets <- BatchGetSymbols(tickers=tickers,
                          first.date=date_init,
                          last.date=date_end,
                          type.return="log", # log retorno
                          freq.data="daily")
ibovespa <- assets[[2]]
```

Após coletarmos os dados, com frequência diária, realizamos os ajustes necessários para termos a série temporal de interesse:

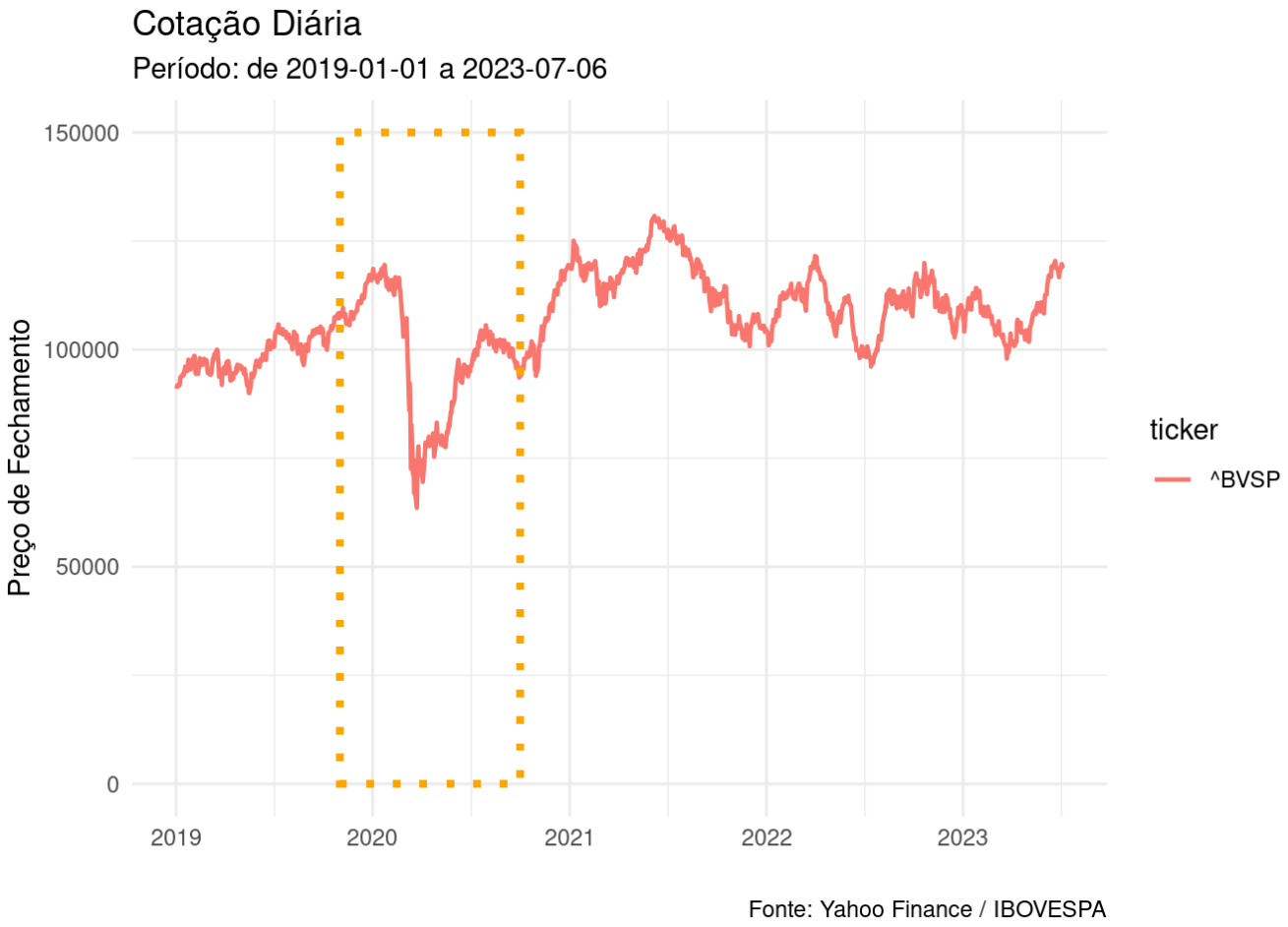
```
daily_returns <- ibovespa %>%
  select(ref.date, ret.closing.prices)

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

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

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

(a) Os códigos abaixo geram gráficos do preço diário e do log-retorno para o IBOVESPA.

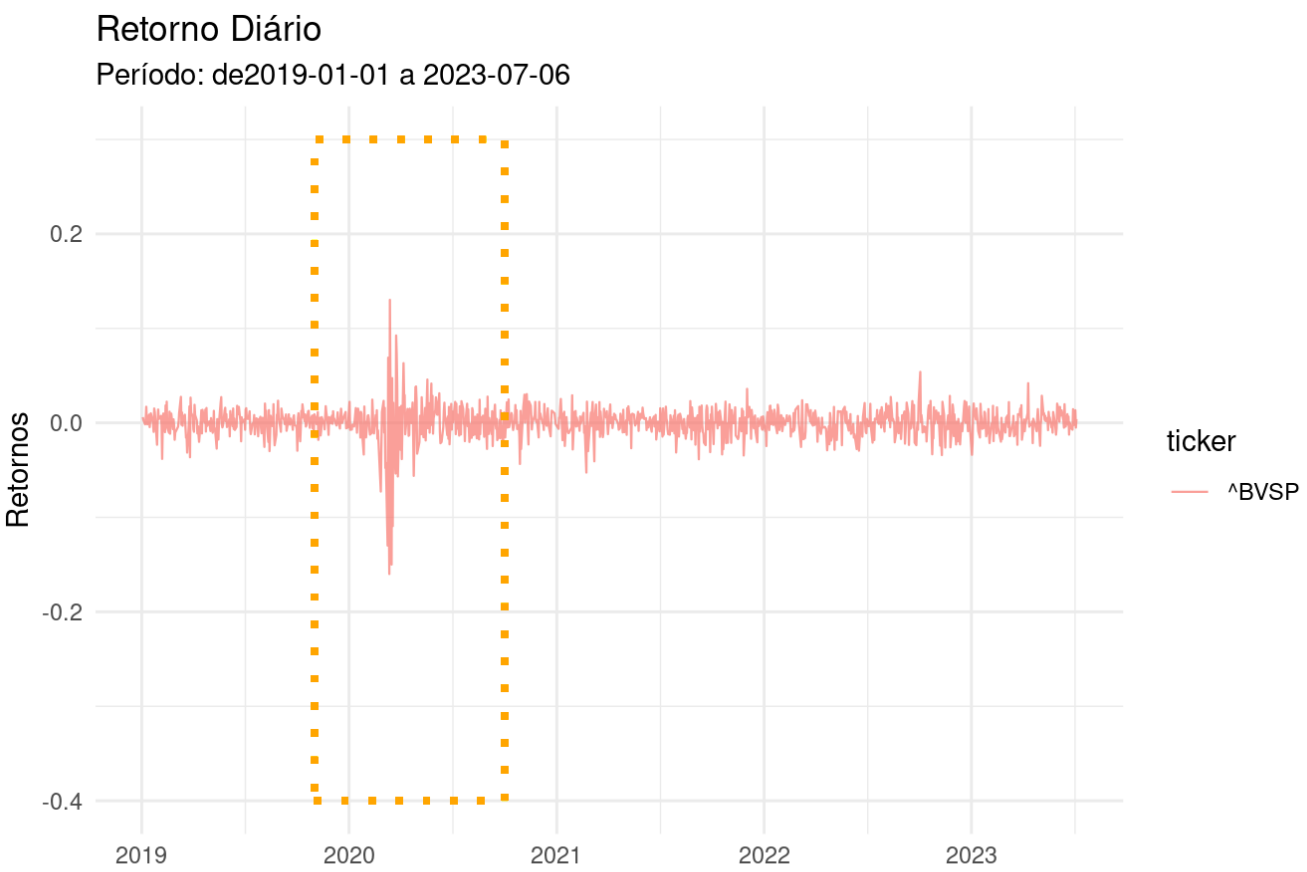
```
library(ggplot2, quietly=TRUE)
library(gridExtra, quietly=TRUE)
min(date$date)
```

```
## [1] "2019-01-03"
```

```
# preço diário
g <- ggplot(data=ibovespa) +
  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=0, ymax=1.5e5),
            fill="transparent", linetype=3, color="orange", size=1.2) +
  labs(x="", y="Preço de Fechamento",
       title="Cotação Diária",
       subtitle=paste("Período: de ", date_init, " a ", date_end, sep=""),
       caption="Fonte: Yahoo Finance / IBOVESPA") +
  theme_minimal()
g
```

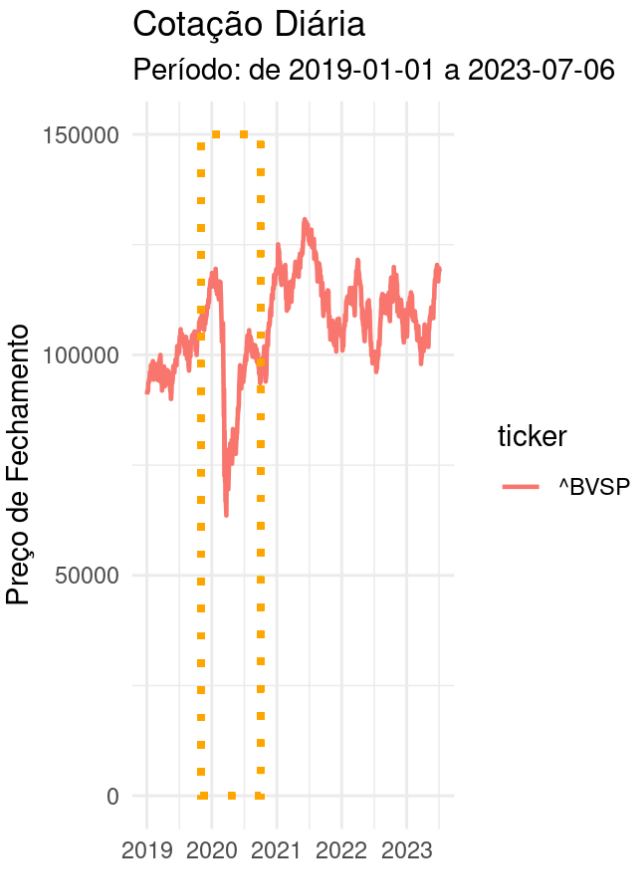



```
# retorno diário
g.returns <- ggplot(data=ibovespa) +
  geom_line(aes(x=ref.date, y=ret.closing.prices, color=ticker),
            alpha=0.7, linewidth=0.4, na.rm=TRUE) +
  geom_rect(aes(xmin=as.Date("2019-11-01"), xmax=as.Date("2020-10-01"),
                ymin=-0.4, ymax=0.3),
            fill="transparent", linetype=3, color="orange", size=1.2) +
  labs(x="" , y="Retornos",
       title="Retorno Diário",
        subtitle=paste("Período: de", date_init, " a ", date_end, sep=""),
        caption="Fonte: Yahoo Finance / IBOVESPA") +
    theme_minimal()
g.returns
```

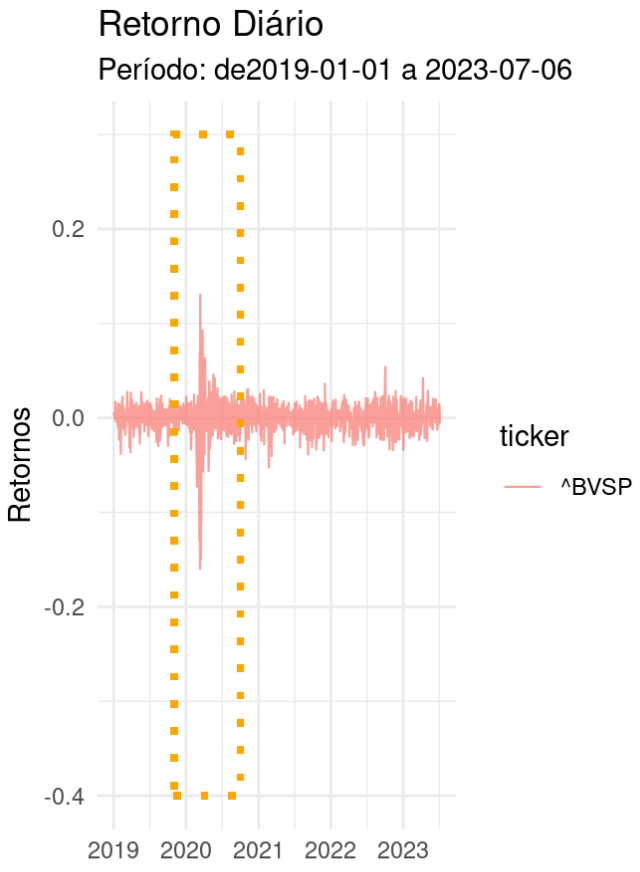


Fonte: Yahoo Finance / IBOVESPA

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



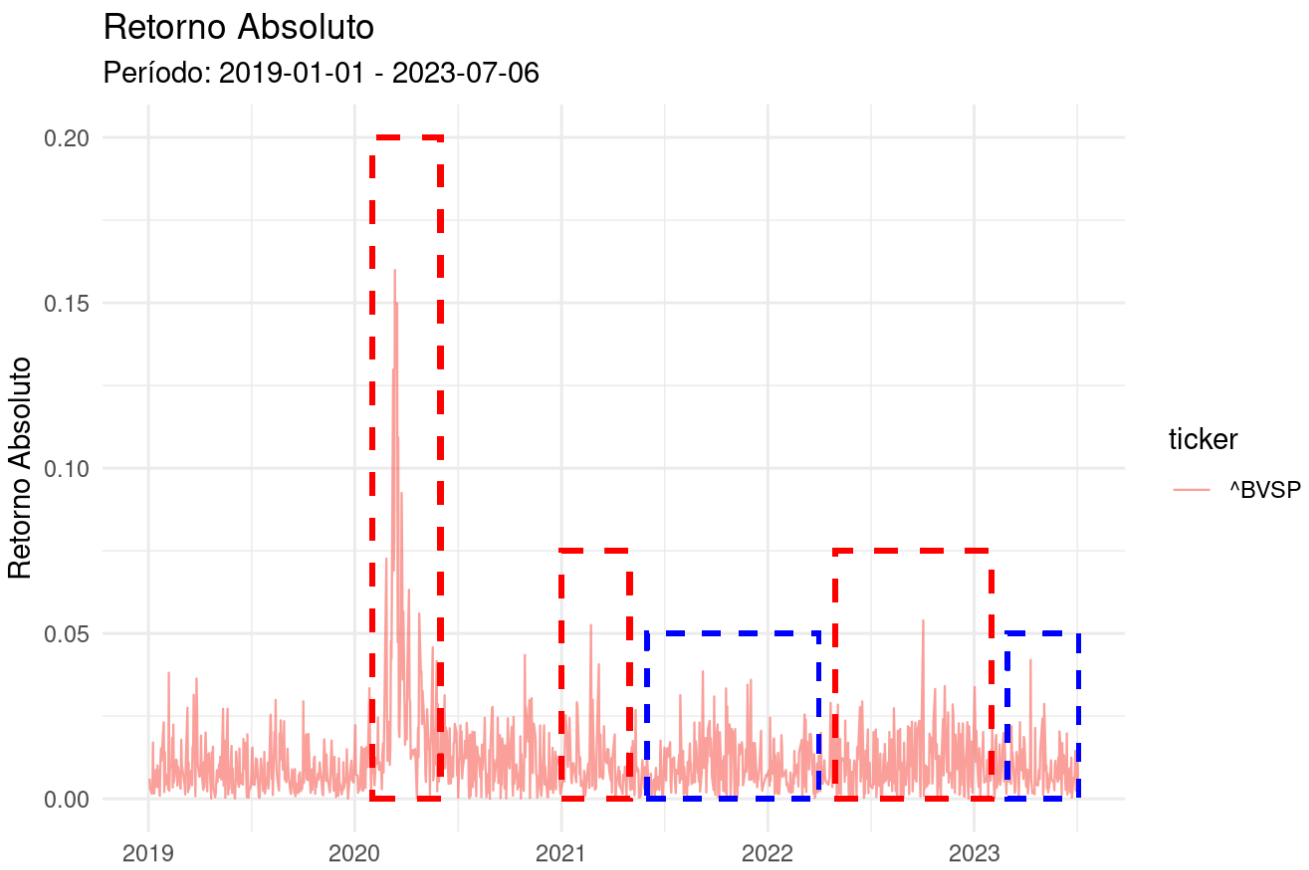
Fonte: Yahoo Finance / IBOVESPA



Fonte: Yahoo Finance / IBOVESPA

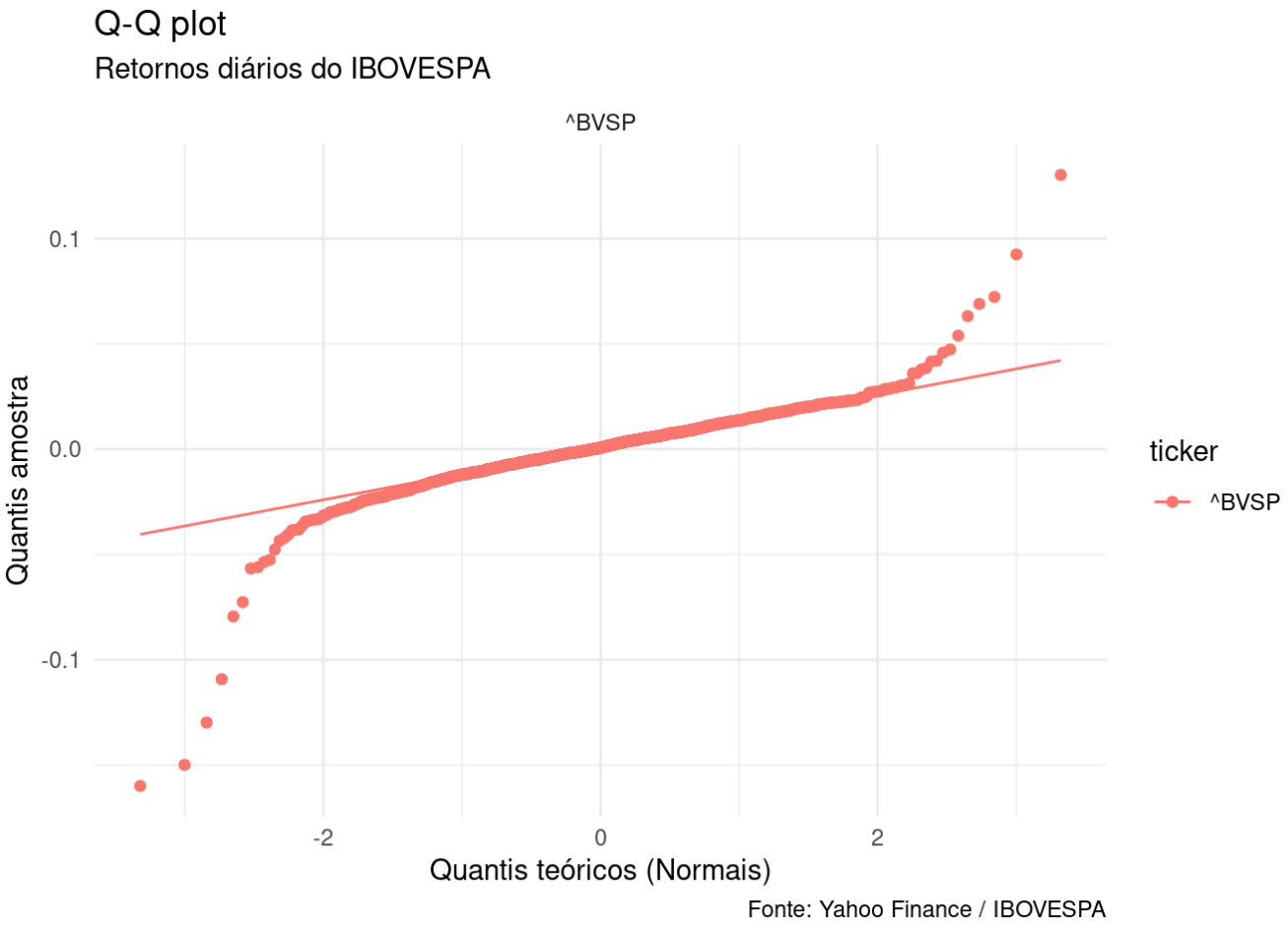
Analisando os gráficos, notamos que a média dos retornos é 0, além disso a cotação diária e os retornos possuem o efeito da alavancagem, pois por volta do primeiro semestre de 2020. O gráfico de retornos mostra que no mesmo período a volatilidade aumentou e há uma correlação negativa entre os retornos e os preços. De fato, foi neste período que a pandemia de COVID-19 (uma *bad news*) começou a crescer no Brasil, aumentando os níveis de incerteza (alta volatilidade).

```
# retornos absolutos
g.volatility <- ggplot(data=ibovespa) +
  geom_line(aes(x=ref.date, y=abs(ret.closing.prices), color=ticker),
            alpha=0.7, linewidth=0.4, na.rm=TRUE) +
  geom_rect(aes(xmin=as.Date("2020-02-01"), xmax=as.Date("2020-06-01"),
                ymin=0, ymax=0.2),
            fill="transparent", linetype=2, color="red", size=1) +
  geom_rect(aes(xmin=as.Date("2021-01-01"), xmax=as.Date("2021-05-01"),
                ymin=0, ymax=0.075),
            fill="transparent", linetype=2, color="red", size=1) +
  geom_rect(aes(xmin=as.Date("2021-06-01"), xmax=as.Date("2022-04-01"),
                ymin=0, ymax=0.05),
            fill="transparent", linetype=2, color="blue", size=0.8) +
  geom_rect(aes(xmin=as.Date("2022-04-30"), xmax=as.Date("2023-02-01"),
                ymin=0, ymax=0.075),
            fill="transparent", linetype=2, color="red", size=1) +
  geom_rect(aes(xmin=as.Date("2023-03-01"), xmax=as.Date("2023-07-05"),
                ymin=0, ymax=0.05),
            fill="transparent", linetype=2, color="blue", size=0.8) +
  labs( x="", y="Retorno Absoluto",
        title="Retorno Absoluto",
        subtitle=paste("Período: ", date_init, " - ", date_end, sep=""),
        caption="Fonte: Yahoo Finance / IBOVESPA")+
  theme_minimal()
g.volatility
```

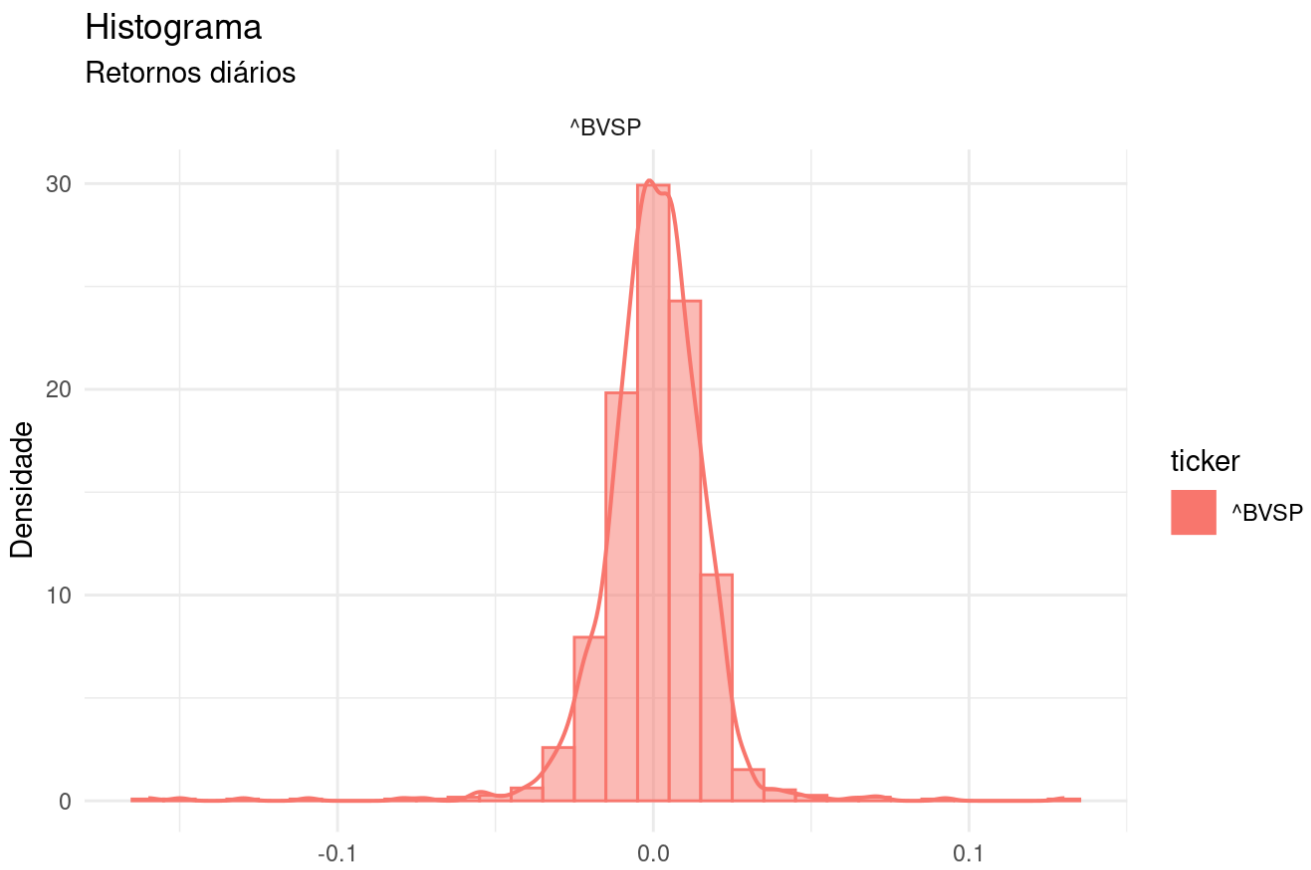


O gráfico acima mostra os valores absolutos dos retornos. Nos trechos destacados em vermelho, temos períodos de maior volatilidade. Já nos períodos destacads em azul, há uma diminuição da volatilidade, se comparado com o os que estão em vermelho.

```
qqplot <- ggplot(data=ibovespa,
                 aes(sample=ret.closing.prices, color=ticker)) +
  stat_qq(na.rm=TRUE) +
  stat_qq_line(na.rm=TRUE) +
    labs(x="Quantis teóricos (Normais)", y="Quantis amostra",
         title="Q-Q plot",
         subtitle="Retornos diários do IBOVESPA",
         caption="Fonte: Yahoo Finance / IBOVESPA") +
  theme_minimal() +
  facet_wrap(~ticker, nrow=2)
qqplot
```




```
# histograma
histogram <- ggplot(data=ibovespa) +
  geom_histogram(aes(x=ret.closing.prices, y=after_stat(density),
                     fill=ticker, color=ticker),
                 linetype=1, alpha=0.5, bins=30, na.rm=TRUE) +
  geom_density(aes(x=ret.closing.prices, y=after_stat(density), color=ticker),
               na.rm=TRUE, linewidth=0.7) +
  labs(x="", y="Densidade", title="Histograma",
       subtitle="Retornos diários",
       caption="Fonte: B3") +
  theme_minimal() +
  facet_wrap(~ticker, nrow=2)
histogram
```



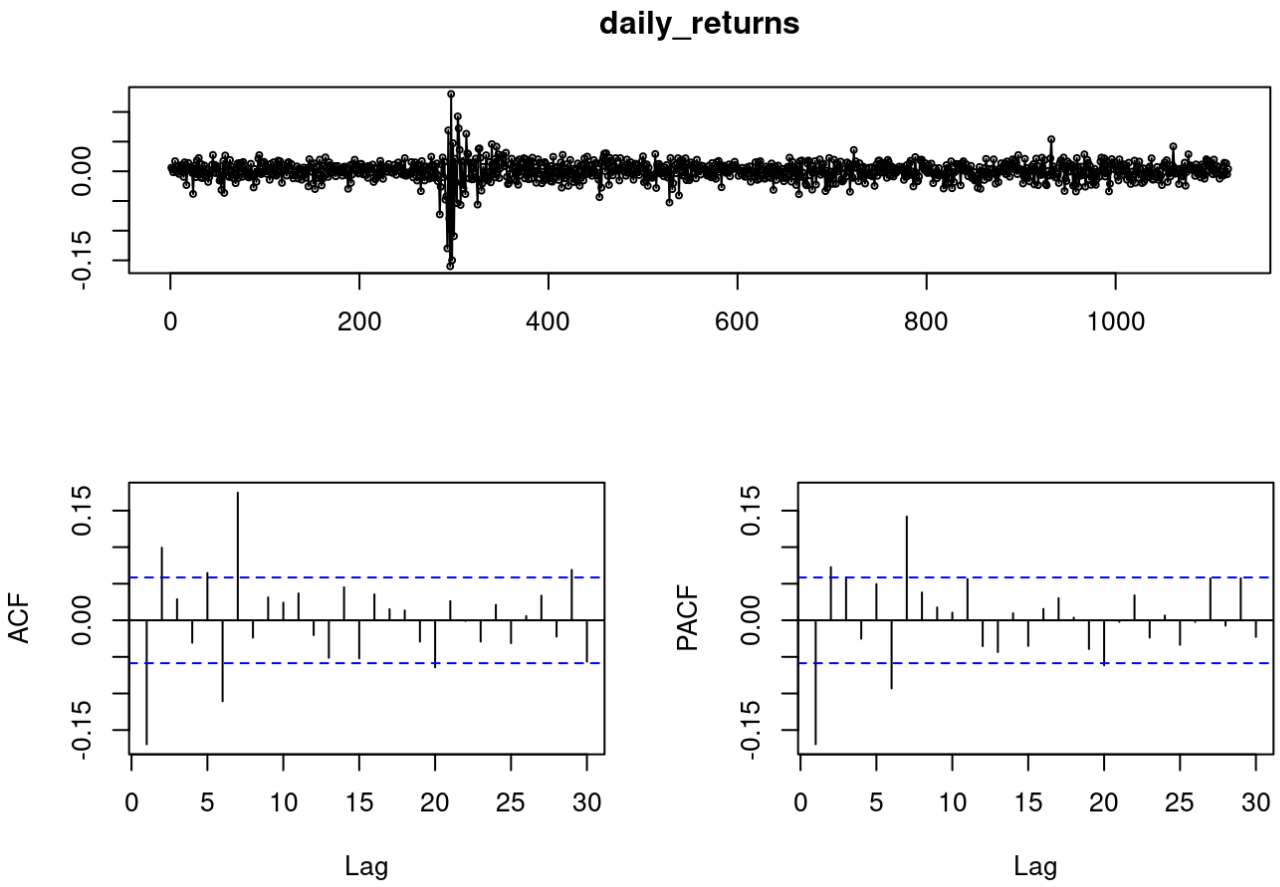
Fonte: B3

Analisando as caudas nos gráficos Q-Q plot acima, notamos que seus percentis essão afastados da Normal. Dessa maneira, a distribuição dos retornos do IBOVESPA possui cauda pesada. Ou seja, eventos raros são comuns de ocorrer. Os histogramas sugerem que a média da distribuição dos retornos é zero.

(b) Vejamos agora uma análise da ACF e da PACF da série de retornos do IBOVESPA.

Para ter uma ideia do modelo a ser estimado, usamos as FAC e FACP. Para gerar os gráficos usaremos o `tsdisplay` :

```
library(forecast)
tsdisplay(daily_returns)
```



O gráfico da PACF acima, sugere um modelo AR(1). Já pelo gráfico da ACF, escolhemos um modelo MA(2) ou MA(1). Logo, uma proposta de modelo para esse processo é o ARMA(1, 2) ou ARMA(1, 1).

(c) Para realizarmos a estimação do modelo ARMA, após as análises do item (b), iremos supor que se trata de uma série estacionária. Além dissom, usaremos o comando `arima` para ajustar o modelo (d=0 em `arima(p, d, q)`):

```
fit_1 <- arima(x=daily_returns, order=c(1, 0, 2))
fit_2 <- arima(x=daily_returns, order=c(1, 0, 1))
BIC(fit_1, fit_2)
```

```
##       df       BIC
## fit_1  5 -5898.771
## fit_2  4 -5899.791
```

```
AIC(fit_1, fit_2)
```

```
##       df       AIC
## fit_1  5 -5923.872
## fit_2  4 -5919.872
```

No fit_1 realizado acima, usamos o modelo `ARMA(1, 2)` e no fit_2 usamos o `ARMA(1, 1)`. Analisando os valores de AIC notamos que o modelo do fit_1 é preferível ao do fit_2, visto que o AIC do fit_1 é menor do que o do fit_2. Já pelo critério BIC, o modelo do fit_2 é preferível ao do fit_1, pois o BIC do fit_2 é menor do que o do fit_1. Dessa maneira, a fim de trabalharmos com modelo menos complexo possível (`parcimônia`), optaremos por usar o modelo `ARMA(1, 1)` ou algum mais simples que este.

Abaixo temos a t-estatística de cada parâmetro, tanto do ARMA(1, 2) como do ARMA(1, 1) aqui considerado.

```
#install.packages("lmtest")
library("lmtest")
#?coeftest
coeftest(fit_1)
```

```
##
## z test of coefficients:
##
##              Estimate  Std. Error z value Pr(>|z|)
## ar1        0.13739900  0.21796520  0.6304 0.528452
## ma1       -0.29850628  0.21563093 -1.3843 0.166255
## ma2        0.13372317  0.04144990  3.2261 0.001255 **
## intercept  0.00023820  0.00049497  0.4812 0.630342
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
coeftest(fit_2)
```

```
##
## z test of coefficients:
##
##              Estimate  Std. Error z value  Pr(>|z|)
## ar1       -0.38028549  0.10684995 -3.5591 0.0003722 ***
## ma1        0.21385587  0.11080418  1.9300 0.0536026 .
## intercept  0.00024060  0.00045103  0.5334 0.5937255
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Vamos retirar o coeficiente correspondente ao maior valor de p das estatísticas acima para o ARMA(1, 1) e reestimar nosso modelo. Assim teremos um ARMA(1, 0).

```
#?arima
fit_3 <- arima(x=daily_returns, order=c(1, 0, 0))
coeftest(fit_3)
```

```
##
## z test of coefficients:
##
##              Estimate  Std. Error z value  Pr(>|z|)
## ar1       -0.16933822  0.02945483 -5.7491 8.973e-09 ***
## intercept  0.00024219  0.00043929  0.5513    0.5814
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Notamos pelas estatísticas acima que o *drift* (intercepto) do ARMA(1, 0) proposto possui valor de acima de 0.05. Logo, vamos consideremos um ARMA(1, 0) sem *drift*:

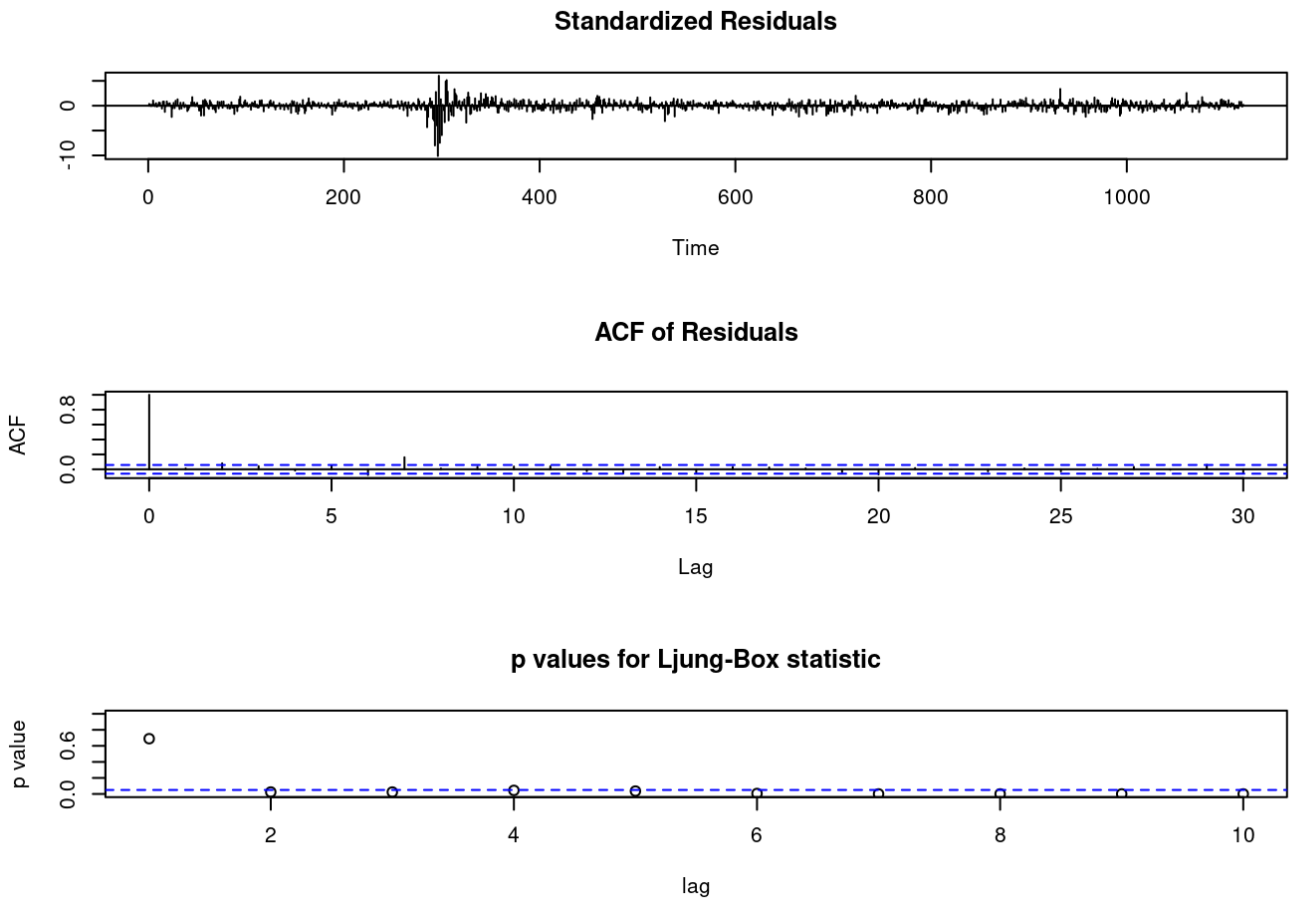
```
fit_4 <- arima(x=daily_returns, order=c(1, 0, 0), include.mean=FALSE)
coeftest(fit_4)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ar1 -0.169118   0.029456 -5.7414 9.392e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Agora o modelo proposto possui coeficiente com valor de p adquado (i.e., $p < 0.05$) de forma que escolhemos o modelo `ARMA(1, 0)` sem *drift*.

(d) Para verificar se o ARMA(1, 0) sem *drift* proposto no fit_4 é adequado, realizamos uma análise do seus resíduos.

```
tsdiag(fit_4)
```



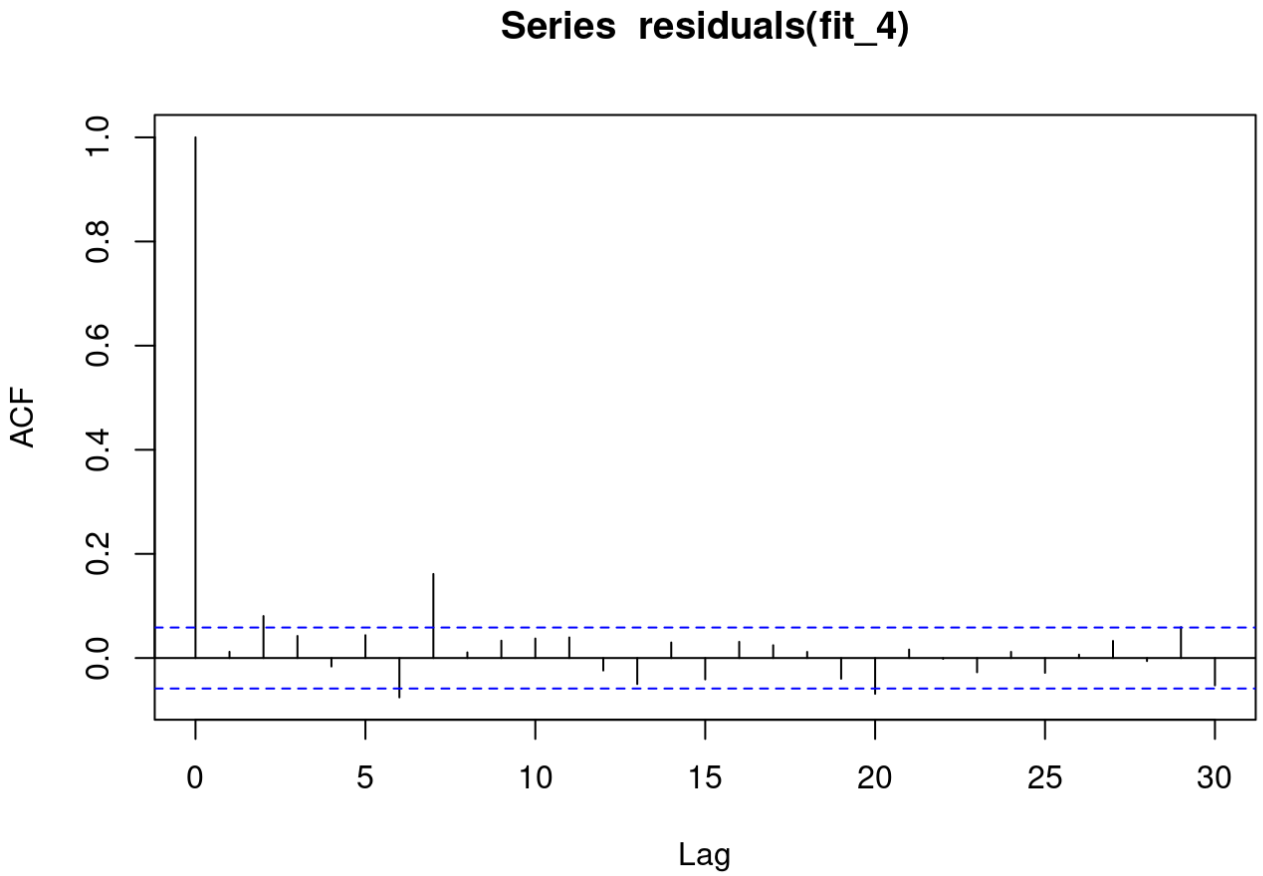

```
Box.test(fit_4$residuals, lag=7, fitdf=1)
```

```
##
##  Box-Pierce test
##
## data:  fit_4$residuals
## X-squared = 47.432, df = 6, p-value = 1.534e-08
```

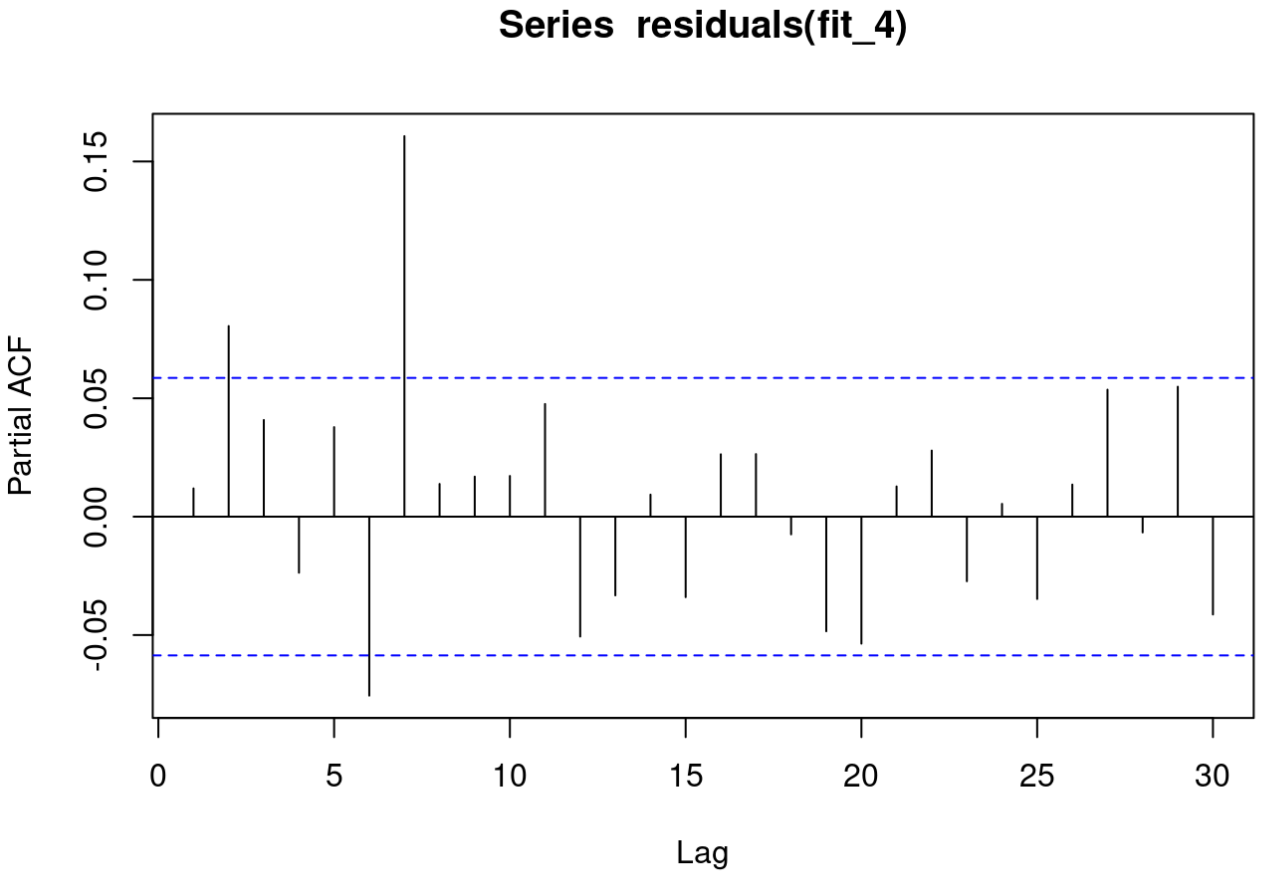
```
Box.test(residuals(fit_4), type="Ljung")
```

```
##
##  Box-Ljung test
##
## data:  residuals(fit_4)
## X-squared = 0.15969, df = 1, p-value = 0.6894
```

```
acf(residuals(fit_4))
```

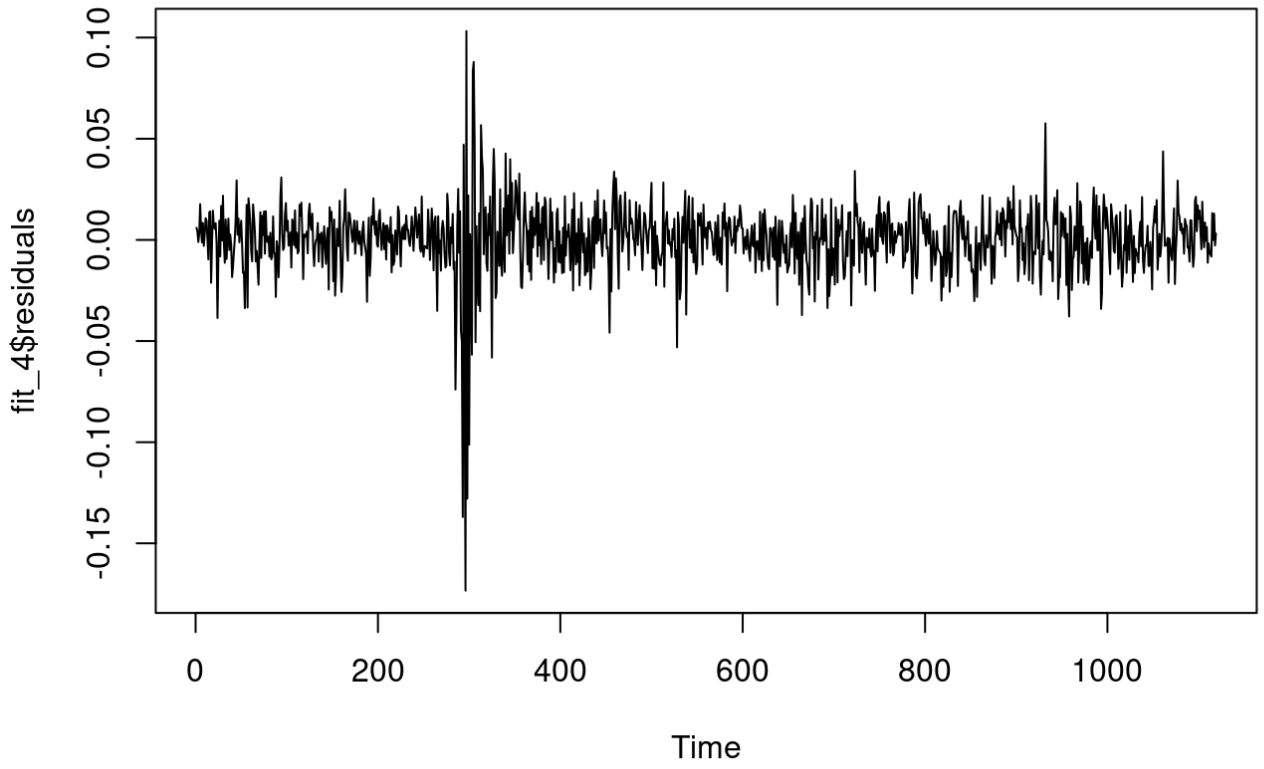



```
pacf(residuals(fit_4))
```



Usamos `fitdf=1` para corrigir o grau de liberdade do teste dado que a série testada é resultado de uma regressão com 1 termo AR. Os valores acima mostram que estatística de teste é igual a `47.432` com valor de p de `1.534e-08`. Portanto, não rejeitamos a hipótese nula e concluímos que os resíduos, até a sétima defasagem, são conjuntamente não correlacionados.

```
plot.ts(fit_4$residuals)
```



O gráfico acima sugere que a média dos resíduos é zero e sua variância é finita. Pelo que vimos anteriormente, a série dos resíduos do modelo ARMA(1, 0) sem *drift*, fit_4, é não autocorrelacionada. Estas características, sugerem que a série dos resíduos do modelo proposto é um ruído branco.

Questão 2

Para o modelo escolhido no exercício anterior, calcule as previsões para 5 períodos à frente, com seu intervalo de confiança correspondente. Lembre-se que a previsão é do tipo estático: apenas informações até o momento t são usadas para fazer previsões em t + k.

Resposta 2

O código abaixo realiza a predição para 5 períodos a frente, utilizando o modelo do exercício anterior.

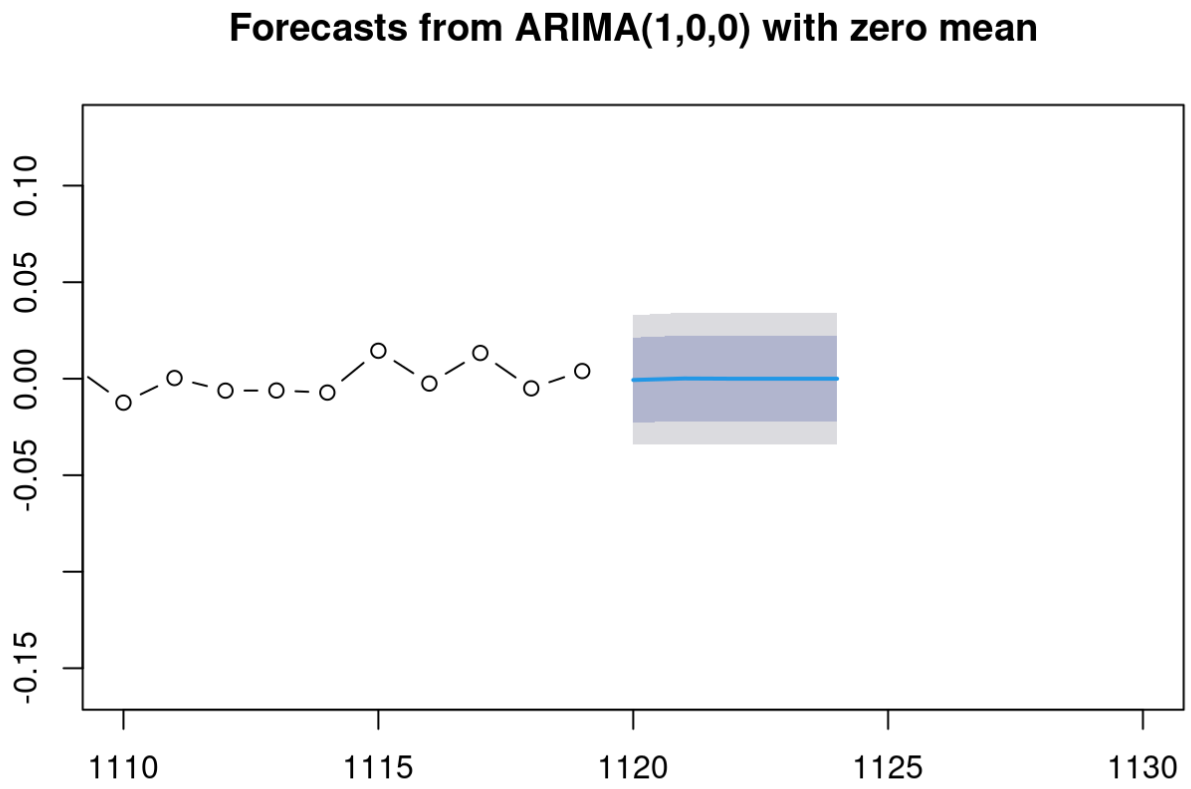
```
# ?predict
prev_5ahead <- predict(fit_4, n.ahead=5, se.fit=T, interval="confidence")
prev_5ahead
```

```
## $pred
## Time Series:
## Start = 1120
## End = 1124
## Frequency = 1
## [1] -6.704502e-04  1.133854e-04 -1.917556e-05  3.242940e-06 -5.484406e-07
##
## $se
## Time Series:
## Start = 1120
## End = 1124
## Frequency = 1
## [1] 0.01714377 0.01738721 0.01739412 0.01739432 0.01739432
```

```
forecast(fit_4, 5)
```

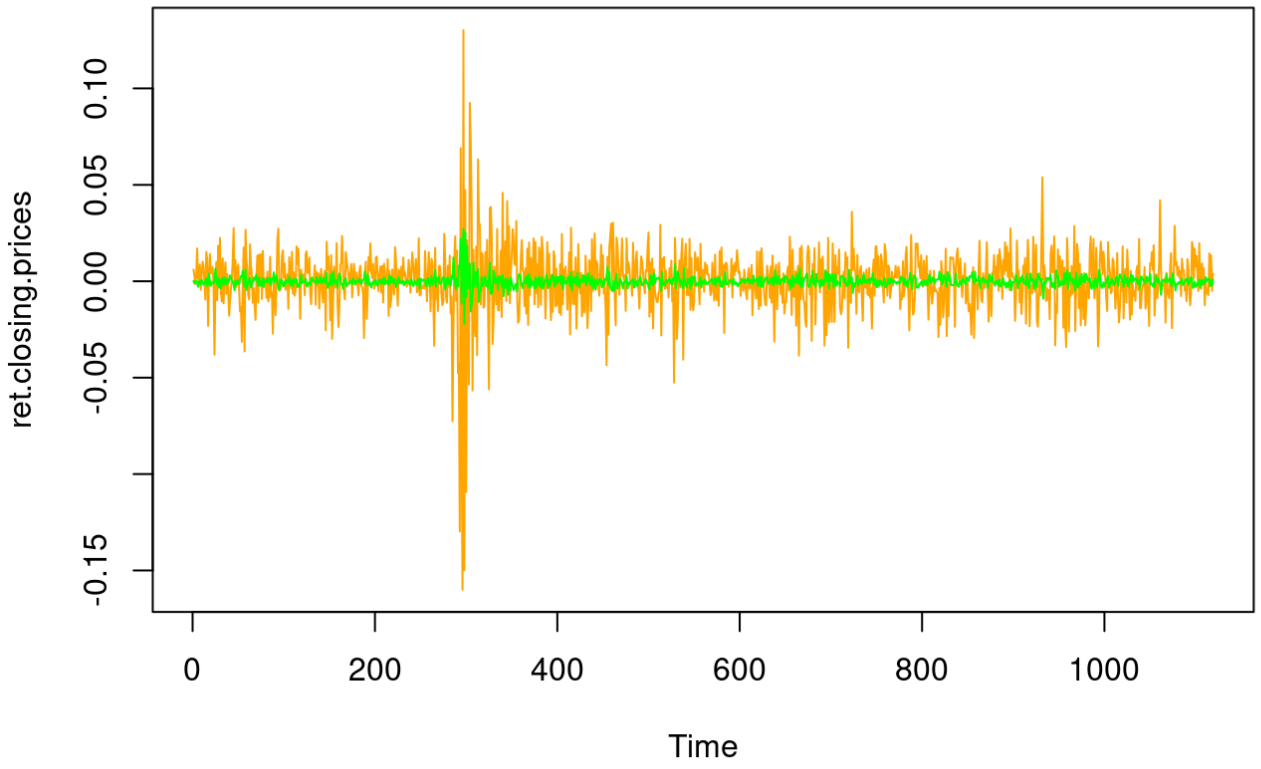
```
##      Point Forecast       Lo 80      Hi 80       Lo 95      Hi 95
## 1120  -6.704502e-04 -0.02264108 0.02130018 -0.03427162 0.03293072
## 1121   1.133854e-04 -0.02216922 0.02239599 -0.03396491 0.03419168
## 1122  -1.917556e-05 -0.02231064 0.02227228 -0.03411102 0.03407267
## 1123   3.242940e-06 -0.02228847 0.02229496 -0.03408899 0.03409548
## 1124  -5.484406e-07 -0.02229227 0.02229117 -0.03409279 0.03409170
```

```
plot(xlim=c(1110, 1130), forecast(fit_4, 5), type="b")
```



A seguir, plotamos a série observada (laranja) e a série estimada(verde).

```
#?lines
plot(daily_returns, col="orange")
lines(fitted(fit_4), col="green")
```



As predições exibidas nos gráficos mostram que os valores ajustados ficaram dentro do esperado.

Questão 3

Utilize função BatchGetSymbols::GetSP500Stocks para baixar dados de todas ações pertencentes ao atual índice SP500. Utilizando seus conhecimentos sobre dplyr, estime um modelo ARMA para os retornos de cada ação dos dados importados. No mesmo dataframe de saída, crie uma nova coluna com a previsão em t+1 de cada modelo. Qual ação possui maior expectativa de retorno?

Resposta 3

```
library(BatchGetSymbols, quietly=TRUE)
BatchGetSymbols::GetSP500Stocks
```

```
## function (do.cache = TRUE, cache.folder = file.path(tempdir(),
##     "BGS_Cache"))
## {
##     cache.file <- file.path(cache.folder, paste0("SP500_Composition_",
##         Sys.Date(), ".rds"))
##     if (do.cache) {
##         flag <- file.exists(cache.file)
##         if (flag) {
##             df.SP500Stocks <- readRDS(cache.file)
##             return(df.SP500Stocks)
##         }
##     }
##     my.url <- "https://en.wikipedia.org/wiki/List_of_S%26P_500_companies"
##     read_html <- 0
##     my.xpath <- "//*[@id=\"constituents\"]"
##     df.SP500Stocks <- my.url %>% read_html() %>% html_nodes(xpath = my.xpath) %
>%
##         html_table(fill = TRUE)
##     df.SP500Stocks <- df.SP500Stocks[[1]]
##     colnames(df.SP500Stocks) <- c("Tickers", "Company", "SEC.filings",
##         "GICS.Sector", "GICS.Sub.Industry", "HQ.Location", "Date.First.Added",
##         "CIK", "Founded")
##     if (do.cache) {
##         if (!dir.exists(cache.folder))
##             dir.create(cache.folder)
##         saveRDS(df.SP500Stocks, cache.file)
##     }
##     return(df.SP500Stocks)
## }
## <bytecode: 0x5582192101f8>
## <environment: namespace:BatchGetSymbols>
```

```
# define datas de início e fim
date_init <- "2019-01-01"
date_end <- "2023-07-06"
#date_end <- Sys.Date()


df.SP500 <- GetSP500Stocks()
#print(df.SP500$Tickers)

tickers <- df.SP500$Tickers

#assets_sp500 <- BatchGetSymbols(tickers=tickers[1:10],
assets_sp500 <- BatchGetSymbols(tickers=tickers,
                          first.date=date_init,
                          last.date=date_end,
                          type.return="log", # log retorno
                          freq.data="daily")

assets_sp500 <- assets_sp500[[2]]
# assets_sp500

# salva dados da SP500
saveRDS(df.SP500, file="dfSP500_tickers.rds")
saveRDS(df.SP500, file="assets_sp500.rds")

# carrega dados
# df.SP500 <- readRDS("dfSP500.rds")
# assets_sp500 <- readRDS("assets_sp500.rds")
```

Vejamos algumas informações sobre os tickets.

```
glimpse(assets_sp500)
```

```
## Rows: 561,083
## Columns: 10
## $ price.open          <dbl> 187.82, 188.28, 186.75, 191.36, 193.00, 193.25, 19…
## $ price.high          <dbl> 190.99, 188.28, 191.98, 192.30, 194.11, 193.94, 19…
## $ price.low           <dbl> 186.70, 182.89, 186.03, 188.66, 189.58, 191.38, 18…
## $ price.close         <dbl> 190.95, 183.76, 191.32, 190.88, 191.68, 192.30, 19…
## $ volume              <dbl> 2475200, 3358200, 2995100, 2162200, 2479800, 21636…
## $ price.adjusted      <dbl> 158.6111, 152.6387, 158.9184, 158.5529, 159.2174, …
## $ ref.date            <date> 2019-01-02, 2019-01-03, 2019-01-04, 2019-01-07, 2…
## $ ticker              <chr> "MMM", "MMM", "MMM", "MMM", "MMM", "MMM", "MMM", "…
## $ ret.adjusted.prices <dbl> NA, -0.0383813904, 0.0403172272, -0.0023027244, 0.…
## $ ret.closing.prices  <dbl> NA, -0.0383810692, 0.0403169286, -0.0023024732, 0.…
```

```
df_item <- assets_sp500 %>%
  filter(ticker=="MMM")
head(df_item)
```

```
##   price.open price.high price.low price.close  volume price.adjusted   ref.date
## 1     187.82     190.99    186.70      190.95 2475200       158.6111 2019-01-02
## 2     188.28     188.28    182.89      183.76 3358200       152.6387 2019-01-03
## 3     186.75     191.98    186.03      191.32 2995100       158.9184 2019-01-04
## 4     191.36     192.30    188.66      190.88 2162200       158.5529 2019-01-07
## 5     193.00     194.11    189.58      191.68 2479800       159.2174 2019-01-08
## 6     193.25     193.94    191.38      192.30 2163600       159.7324 2019-01-09
##   ticker ret.adjusted.prices ret.closing.prices
## 1    MMM                  NA                 NA
## 2    MMM        -0.038381390       -0.038381069
## 3    MMM         0.040317227        0.040316929
## 4    MMM        -0.002302724       -0.002302473
## 5    MMM         0.004182304        0.004182293
## 6    MMM         0.003229443        0.003229392
```

Agora vamos criar uma nova coluna com a previsão em t+1 de cada modelo:

```
library(forecast)

df <- assets_sp500
#head(assets_sp500)
df_res <- 0

for (x in tickers[1:3]) {
  #print(x)
  assets_item <- assets_sp500 %>%
    filter(ticker==x)

  # obtém retorno diário e série temporal de retornos
  daily_returns_item <- assets_item %>%
    select(ref.date, ret.closing.prices)

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

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

  daily_returns_item <- as.ts(daily_returns_item)

  # ajusta um ARMA(1, 0, 0)
  fit_item <- arima(x=daily_returns_item, order=c(1, 0, 0))

  prev_1ahead <- predict(fit_item, n.ahead=1, se.fit=T, interval="confidence")
  # prev_1ahead

  # adicionar nome na coluna, ou criar regra
  pred_forecast <- forecast(fit_item, 1)
  assets_item$prev.1ahead_aux1 <- pred_forecast$mean[1]

  if (df_res == 0) {
    df_res <- assets_item
  }
  else{
    df_res <- df_res %>%
      bind_rows(assets_item)
  }
}

df_res_top_1 <- df_res %>%
  select(ticker, prev.1ahead_aux1) %>%
  group_by(ticker) %>%
  distinct(ticker, prev.1ahead_aux1) %>%
  arrange(desc(prev.1ahead_aux1), .by_group=FALSE)
```

Portanto, a ação possui maior expectativa de retorno é:

```
head(df_res_top_1, 1)
```

```
## # A tibble: 1 × 2
## # Groups:   ticker [1]
##   ticker prev.1ahead_aux1
##   <chr>             <dbl>
## 1 MMM             0.00128
```

Questão 4

Separe os dados do SP500 em duas partes, etapa de estimação e etapa de previsão. Suponha que você queira, por exemplo, comprar a ação quando a previsão de retorno for positiva, vendendo-a no dia seguinte. As previsões dos modelos ARIMA permitem a construção de uma estratégia de negociação lucrativa?

Resposta 4

Consideremos a seguinte separação:

```
library(dplyr)
dim(df_res)
```

```
## [1] 3402   11
```

```
df_predicao <- df_res %>%
    slice(7500:dim(df_res)[1]-1)

fit_estimaticao <- arima(x=daily_returns, order=c(1, 0, 0), include.mean=FALSE)
coeftest(fit_4)
```

```
##
## z test of coefficients:
##
##      Estimate Std. Error z value  Pr(>|z|)
## ar1 -0.169118   0.029456 -5.7414 9.392e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
library(forecast)

df <- assets_sp500
#head(assets_sp500)
df_res <- 0

#for (x in tickers[1:7]) {
for (x in tickers) {
  # print(x)
  assets_item <- assets_sp500 %>%
    filter(ticker==x)

  # print(dim(assets_item))
  if (dim(assets_item)[1] > 0){
    # obtém retorno diário e série temporal de retornos
    daily_returns_item <- assets_item %>%
      select(ref.date, ret.closing.prices)

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

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

    daily_returns_item <- as.ts(daily_returns_item)

    # ajusta um ARMA(1, 0, 0)
    fit_item <- arima(x=daily_returns_item, order=c(1, 0, 0))

    prev_1ahead <- predict(fit_item, n.ahead=1, se.fit=T, interval="confidence")
    # prev_1ahead

    # adiciona coluna com predições
    pred_forecast <- forecast(fit_item, 1)
    assets_item$prev.1ahead_aux1 <- pred_forecast$mean[1]

    if (df_res == 0) {
      df_res <- assets_item
    }
    else{
      df_res <- df_res %>%
        bind_rows(assets_item)
    }
  }
}

df_res %>%
    group_by(ticker) %>%
    filter(prev.1ahead_aux1 == max(prev.1ahead_aux1))
```

```
## # A tibble: 561,083 × 11
## # Groups:   ticker [495]
##    price.open price.high price.low price.close  volume price.adjusted ref.date
##         <dbl>      <dbl>     <dbl>       <dbl>   <dbl>          <dbl> <date>
##  1       188.       191.      187.        191. 2475200           159. 2019-01-0
2
##  2       188.       188.      183.        184. 3358200           153. 2019-01-0
3
##  3       187.       192.      186.        191. 2995100           159. 2019-01-0
4
##  4       191.       192.      189.        191. 2162200           159. 2019-01-0
7
##  5       193        194.      190.        192. 2479800           159. 2019-01-0
8
##  6       193.       194.      191.        192. 2163600           160. 2019-01-0
9
##  7       191.       194.      189.        194. 1939300           161. 2019-01-1
0
##  8       192.       193.      191.        192. 2359900           160. 2019-01-1
1
##  9       191.       193.      190.        192. 1914400           160. 2019-01-1
4
## 10       189.       191.      188.        189. 2737300           157. 2019-01-1
5
## # ℹ 561,073 more rows
## # ℹ 4 more variables: ticker <chr>, ret.adjusted.prices <dbl>,
## #   ret.closing.prices <dbl>, prev.1ahead_aux1 <dbl>
```

```
df_res_top_n <- df_res %>%
  select(ref.date, ticker, prev.1ahead_aux1) %>%
  group_by(ticker) %>%
  distinct(ticker, prev.1ahead_aux1) %>%
  arrange(desc(prev.1ahead_aux1), .by_group=FALSE)

# ref: https://stackoverflow.com/questions/70139121/separating-positive-and-negati
ve-values-in-r-data-table-many-columns
df_res_top_n <- df_res_top_n %>%
  mutate(across(everything(),
                ~case_when(. < 0 ~ 0,TRUE ~ .),
                .names = "{col}_pos")) %>%
  mutate(across(-contains("pos"),
                ~case_when(. < 0 ~ ., TRUE ~ 0),
                .names = "{col}_neg"))

head(df_res_top_n)
```

```
## # A tibble: 6 × 4
## # Groups:   ticker [6]
##   ticker prev.1ahead_aux1 prev.1ahead_aux1_pos prev.1ahead_aux1_neg
##   <chr>              <dbl>                <dbl>                <dbl>
## 1 QRVO             0.0102               0.0102                   0
## 2 MPWR             0.00768              0.00768                  0
## 3 SWKS             0.00662              0.00662                  0
## 4 KLAC             0.00659              0.00659                  0
## 5 ADI              0.00596              0.00596                  0
## 6 LRCX             0.00553              0.00553                  0
```

Após a divisão, podemos organizar os dados conforme feito acima, de forma que os indíces cuja previsão de retorno for positiva podem ser comprados hoje e vendido no dia seguinte. Caso a predição com o modelo ARMA ajustado, feita para o dia segunite, fique de fato próxima do valor real observado para cada um dos tickers, essa estratégia seria lucrativa.

```
# seleciona tickers com predição de retorno positiva
pos_tickers <- df_res_top_n %>%
  filter(prev.1ahead_aux1_pos > 0) %>%
  arrange(ticker)
dim(pos_tickers)
```

```
## [1] 386   4
```

```
# adiciona coluna com predições para o dia seguinte
df_comparacao <- df %>%
  filter(ref.date == max(df$ref.date),
         ticker %in% pos_tickers$ticker) %>%
  arrange(ticker) %>%
  mutate(pred1ahead=pos_tickers$prev.1ahead_aux1_pos)

# adiciona coluna da diferença
df_comparacao <- df_comparacao %>%
  mutate(diferenca=ret.closing.prices-pred1ahead) %>%
  select(ticker, ret.closing.prices, pred1ahead, diferenca)

dim(df_comparacao)
```

```
## [1] 386   4
```

```
head(df_comparacao)
```

```
##   ticker ret.closing.prices   pred1ahead    diferenca
## 1      A        0.001507414 0.0004184259  0.001088988
## 2    AAL        0.012081383 0.0008502109  0.011231172
## 3   AAPL       -0.005888679 0.0024085622 -0.008297242
## 4    ABC       -0.001415228 0.0011943470 -0.002609575
## 5    ABT        0.001398029 0.0002580295  0.001140000
## 6   ACGL       -0.005013155 0.0014732134 -0.006486368
```

A tabela acima agrupa os resultados das predições para um dia a frente, considerando apenas as predições que tiveram valores positivos. Ainda nesta tabela, acrescentamos os valores observados para o dia da predição e uma coluna com a diferença entre o valor observado e o valor predito pelo modelo. Para saber se a estratégia que propomos foi lucrativa, podemos analisar a soma da coluna `diferença`. Se ela for positiva, nossa estratégia foi lucrativa, caso contrário, não foi. O código abaixo checa se isso ocorre:

```
sum(df_comparacao$diferenca)
```

```
## [1] -3.427035
```

Como o valor foi negativo, nossa estratégia não foi lucrativa. Se conseguissemos ajustar modelos ARMA mais precisos, talvez a estratégia proposta fosse lucrativa para este cenário.

Outra tentativa para termos uma estratégia lucrativa é selecionarmos apenas os índices cuja previsão se concretizou como sendo positiva. Dessa forma, as operações ocorreríam apenas em índices que o modelo prediz valores positivos para o retorno e que de fato tem ocorrido nos valores observados em dias passados.

Como há uma margem de confiança nas predições dos valores, também poderíamos nos concentrar nos tickers cuja as predições tiveram menores intervalos confiança, ou que toda faixa de intervalo de confiança seja positivos. Neste caso, seria esperado que as flutuações da predição para os tickers analisados tivessem maiores chances de retorno positivo. A margem de confiança seria tal que, mesmo com as flutuações da predição (em torno do valor médio predito), teríamos maiores chances de obter valores positivos para cada um dos tickers em questão. Dessa forma, os tickers selecionados pela estratégia talvez possam trazer lucros.

Referências

- Materiais das aulas (profa. Andreza Palma)
- CAP. 2 do livro “TSAY, Ruey S. *An introduction to analysis of financial data with R.* John Wiley & Sons, 2014.”