Baixando dados diários

Lucca Simeoni Pavan *João Carlos de Carvalho*

22 de novembro de 2016

Sumário


```
knitr::opts_chunk$set(echo = TRUE, cache = FALSE, warning = FALSE, message = FALSE,
    error = FALSE, tidy = TRUE, tidy.opts = list(width.cutoff = 70))
```

1 Ranking de negociações

```
library(GetHFData)
tickers_equity <- ghfd_get_available_tickers_from_ftp(my.date = "2016-10-30",
    type.market = "equity", max.dl.tries = 10)
```

```
##
## Reading ftp contents for equity (attempt = 1|10) Attempt 1 - File exists, skipping dl
```

```
head(tickers_equity, n = 10)
```

```
##    tickers n.trades                     f.name
## 1    PETR4    52393 ftp files/NEG_20161117.zip
## 2    JBSS3    45174 ftp files/NEG_20161117.zip
## 3    ITSA4    39200 ftp files/NEG_20161117.zip
## 4    ITUB4    30529 ftp files/NEG_20161117.zip
## 5    VALE5    30423 ftp files/NEG_20161117.zip
## 6    BVMF3    29099 ftp files/NEG_20161117.zip
## 7    BBDC4    26923 ftp files/NEG_20161117.zip
## 8    ABEV3    26786 ftp files/NEG_20161117.zip
## 9    BBAS3    26672 ftp files/NEG_20161117.zip
## 10   RUMO3    26274 ftp files/NEG_20161117.zip
```

Criando um vetor com as 6 ações mais negociadas em 30/10/2016.

```
top_6 <- c(as.character(head(tickers_equity$tickers)))
print(top_6)
```

```
## [1] "PETR4" "JBSS3" "ITSA4" "ITUB4" "VALE5" "BVMF3"
```

Baixando os dados

```r
dados_top6 <- ghfd_get_HF_data(top_6, type.market = "equity", first.date = as.Date("2014-11-03"),
    last.date = as.Date("2016-10-30"), first.time = "9:00:00", last.time = "18:00:00",
    type.output = "agg", agg.diff = "1 hour", dl.dir = "ftp files", max.dl.tries = 10,
    clean.files = FALSE)
save(dados_top6, file = "dados_top6.Rda")
head(dados_top6, n = 6)
```

```r
load("dados_top6.Rda")
dim(dados_top6)
```

```
## [1] 22667    13
```

```r
str(dados_top6)
```

```
## 'data.frame':    22667 obs. of  13 variables:
##  $ InstrumentSymbol: chr  "ABEV3" "ABEV3" "ABEV3" "ABEV3" ...
##  $ SessionDate     : Date, format: "2014-11-03" "2014-11-03" ...
##  $ TradeDateTime   : POSIXct, format: "2014-11-03 10:00:00" "2014-11-03 11:00:00" ...
##  $ n.trades        : int  1607 2055 3417 3686 3978 4707 5168 250 1602 1203 ...
##  $ last.price      : num  16.1 16.1 16.2 16.1 16.1 ...
##  $ weighted.price  : num  16.1 16.1 16.2 16.2 16.1 ...
##  $ period.ret      : num  -0.00864 0.00124 0.0056 -0.00124 -0.00372 ...
##  $ period.ret.volat: num  0.000325 0.000324 0.000278 0.000235 0.000263 ...
##  $ sum.qtd         : num  824900 926700 1408500 1034900 1141100 ...
##  $ sum.vol         : num  13291157 14907444 22757436 16729199 18362060 ...
##  $ n.buys          : int  579 1113 1888 2265 1972 1878 2309 23 659 526 ...
##  $ n.sells         : int  1028 942 1529 1421 2006 2829 2859 227 943 677 ...
##  $ Tradetime       : chr  "10:00:00" "11:00:00" "12:00:00" "13:00:00" ...
```

Agora irei criar um banco de dados para cada ação e depois obter os log retornos.

```r
library(dplyr)
dados_ITSA4 <- filter(dados_top6, InstrumentSymbol == "ITSA4") %>%
    select(SessionDate, weighted.price) %>% mutate(log_retorno = log(weighted.price) -
    lag(log(weighted.price)))
dados_PETR4 <- filter(dados_top6, InstrumentSymbol == "PETR4") %>%
    select(SessionDate, weighted.price) %>% mutate(log_retorno = log(weighted.price) -
    lag(log(weighted.price)))
dados_ITUB4 <- filter(dados_top6, InstrumentSymbol == "ITUB4") %>%
    select(SessionDate, weighted.price) %>% mutate(log_retorno = log(weighted.price) -
    lag(log(weighted.price)))
dados_BBDC4 <- filter(dados_top6, InstrumentSymbol == "BBDC4") %>%
    select(SessionDate, weighted.price) %>% mutate(log_retorno = log(weighted.price) -
    lag(log(weighted.price)))
dados_ABEV3 <- filter(dados_top6, InstrumentSymbol == "ABEV3") %>%
    select(SessionDate, weighted.price) %>% mutate(log_retorno = log(weighted.price) -
    lag(log(weighted.price))) %>% mutate(log_retorno = log(weighted.price) -
    lag(log(weighted.price)))
dados_BBSE3 <- filter(dados_top6, InstrumentSymbol == "BBSE3") %>%
    select(SessionDate, weighted.price) %>% mutate(log_retorno = log(weighted.price) -
    lag(log(weighted.price)))
```

Removendo NAs.

```r
dados_BBSE3 <- dados_BBSE3[2:3778, ]
dados_ABEV3 <- dados_ABEV3[2:3778, ]
dados_BBDC4 <- dados_BBDC4[2:3778, ]
dados_ITUB4 <- dados_ITUB4[2:3778, ]
dados_PETR4 <- dados_PETR4[2:3777, ]
dados_ITSA4 <- dados_ITSA4[2:3778, ]
```

Criando matriz com os dados diários.

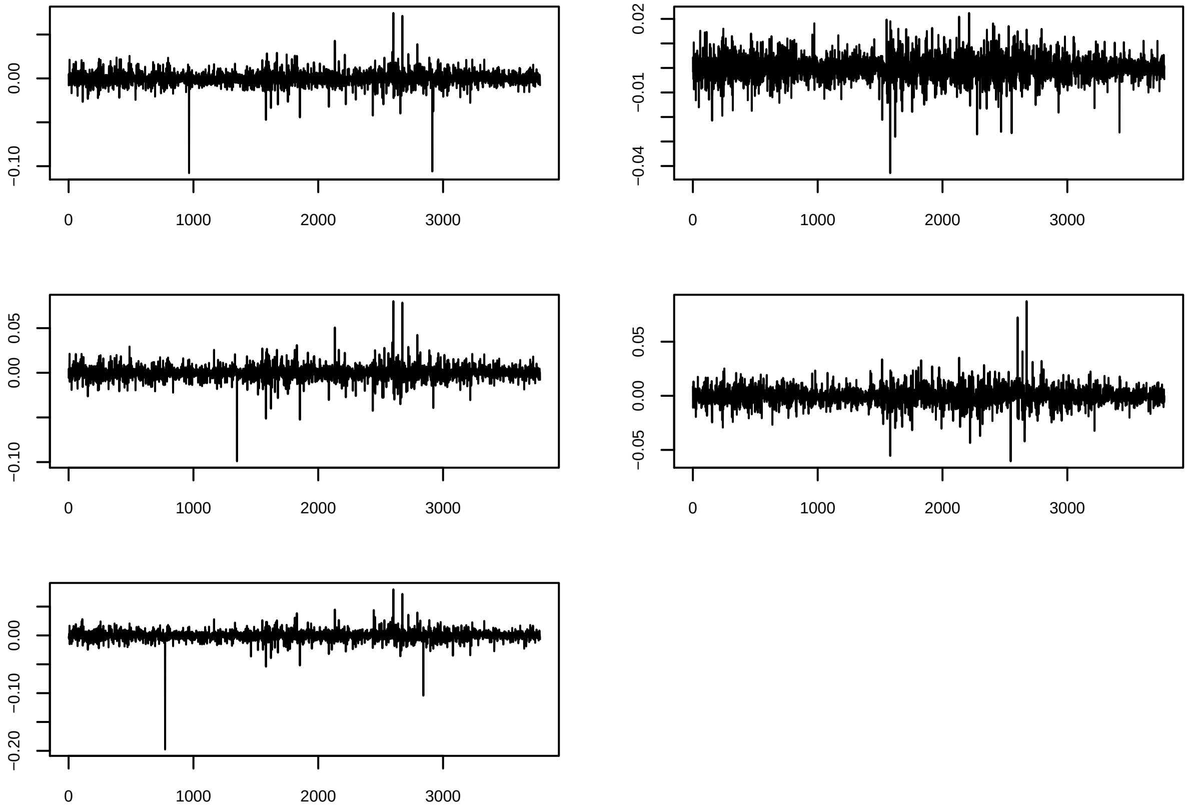
```r
matriz_logrtn <- data.frame(lrtn_ITSA4 = dados_ITSA4$log_retorno, lrtn_ITUB4 = dados_ITUB4$log_retorno,
    lrtn_BBDC4 = dados_BBDC4$log_retorno, lrtn_ABEV3 = dados_ABEV3$log_retorno,
    lrtn_BBSE3 = dados_BBSE3$log_retorno)
summary(matriz_logrtn)
```

```
##    lrtn_ITSA4           lrtn_ITUB4           lrtn_BBDC4
##  Min.   :-1.079e-01   Min.   :-9.911e-02   Min.   :-1.976e-01
##  1st Qu.:-3.412e-03   1st Qu.:-3.276e-03   1st Qu.:-3.698e-03
##  Median :-1.467e-04   Median :-5.178e-05   Median :-3.910e-06
##  Mean   :-4.198e-05   Mean   :-5.640e-06   Mean   :-5.718e-05
##  3rd Qu.: 3.154e-03   3rd Qu.: 3.205e-03   3rd Qu.: 3.599e-03
##  Max.   : 7.451e-02   Max.   : 8.019e-02   Max.   : 7.990e-02
##    lrtn_ABEV3           lrtn_BBSE3
##  Min.   :-4.289e-02   Min.   :-6.050e-02
##  1st Qu.:-2.287e-03   1st Qu.:-3.531e-03
##  Median : 7.486e-05   Median : 1.017e-05
##  Mean   : 5.483e-05   Mean   :-2.386e-05
##  3rd Qu.: 2.359e-03   3rd Qu.: 3.362e-03
##  Max.   : 2.241e-02   Max.   : 8.743e-02
```

```r
cor(matriz_logrtn)
```

```
##            lrtn_ITSA4 lrtn_ITUB4 lrtn_BBDC4 lrtn_ABEV3 lrtn_BBSE3
## lrtn_ITSA4  1.0000000  0.8441257  0.7255955  0.4612668  0.5651589
## lrtn_ITUB4  0.8441257  1.0000000  0.7891058  0.4916024  0.5817463
## lrtn_BBDC4  0.7255955  0.7891058  1.0000000  0.4387216  0.5068598
## lrtn_ABEV3  0.4612668  0.4916024  0.4387216  1.0000000  0.4038272
## lrtn_BBSE3  0.5651589  0.5817463  0.5068598  0.4038272  1.0000000
```

```r
library(MTS)
MTSplot(matriz_logrtn)
```




```r
head(matriz_logrtn)
```

```
##      lrtn_ITSA4    lrtn_ITUB4   lrtn_BBDC4    lrtn_ABEV3   lrtn_BBSE3
## 1 -0.0079058649 -0.0057653310 -0.004521064 -0.0016150278 -0.010986101
## 2  0.0050112343  0.0045462710 -0.002177734  0.0043904510  0.008955299
## 3 -0.0030588852 -0.0059963761 -0.001079848  0.0004751700 -0.004649955
## 4 -0.0036985681 -0.0022077264 -0.006330370 -0.0045601580 -0.009555026
## 5  0.0019715511  0.0028860314  0.003117587 -0.0008106029  0.013210150
## 6  0.0002158613 -0.0003814315 -0.004567171 -0.0029404438 -0.006192878
```

```r
ccm
```

```
## function (x, lags = 12, level = FALSE, output = T)
## {
##     if (!is.matrix(x))
##         x = as.matrix(x)
##     nT = dim(x)[1]
##     k = dim(x)[2]
##     if (lags < 1)
##         lags = 1
##     y = scale(x, center = TRUE, scale = FALSE)
##     V1 = cov(y)
##     if (output) {
##         print("Covariance matrix:")
##         print(V1, digits = 3)
```

```
##     }
##     se = sqrt(diag(V1))
##     SD = diag(1/se)
##     S0 = SD %*% V1 %*% SD
##     ksq = k * k
##     wk = matrix(0, ksq, (lags + 1))
##     wk[, 1] = c(S0)
##     j = 0
##     if (output) {
##         cat("CCM at lag: ", j, "\n")
##         print(S0, digits = 3)
##         cat("Simplified matrix:", "\n")
##     }
##     y = y %*% SD
##     crit = 2/sqrt(nT)
##     for (j in 1:lags) {
##         y1 = y[1:(nT - j), ]
##         y2 = y[(j + 1):nT, ]
##         Sj = t(y2) %*% y1/nT
##         Smtx = matrix(".", k, k)
##         for (ii in 1:k) {
##             for (jj in 1:k) {
##                 if (Sj[ii, jj] > crit)
##                   Smtx[ii, jj] = "+"
##                 if (Sj[ii, jj] < -crit)
##                   Smtx[ii, jj] = "-"
##             }
##         }
##         if (output) {
##             cat("CCM at lag: ", j, "\n")
##             for (ii in 1:k) {
##                 cat(Smtx[ii, ], "\n")
##             }
##             if (level) {
##                 cat("Correlations:", "\n")
##                 print(Sj, digits = 3)
##             }
##         }
##         wk[, (j + 1)] = c(Sj)
##     }
##     if (output) {
##         par(mfcol = c(k, k))
##         k0 = 4
##         if (k > k0)
##             par(mfcol = c(k0, k0))
##         tdx = c(0, 1:lags)
##         jcnt = 0
##         if (k > 10) {
##             print("Skip the plots due to high dimension!")
##         }
##         else {
##             for (j in 1:ksq) {
##                 plot(tdx, wk[j, ], type = "h", xlab = "lag",
##                   ylab = "ccf", ylim = c(-1, 1))
```

```
##                 abline(h = c(0))
##                 crit = 2/sqrt(nT)
##                 abline(h = c(crit), lty = 2)
##                 abline(h = c(-crit), lty = 2)
##                 jcnt = jcnt + 1
##                 if ((jcnt == k0^2) && (k > k0)) {
##                   jcnt = 0
##                   cat("Hit Enter for more plots:", "\n")
##                   readline()
##                 }
##             }
##         }
##         par(mfcol = c(1, 1))
##         cat("Hit Enter for p-value plot of individual ccm: ",
##             "\n")
##         readline()
##     }
##     r0i = solve(S0)
##     R0 = kronecker(r0i, r0i)
##     pv = rep(0, lags)
##     for (i in 1:lags) {
##         tmp = matrix(wk[, (i + 1)], ksq, 1)
##         tmp1 = R0 %*% tmp
##         ci = crossprod(tmp, tmp1) * nT * nT/(nT - i)
##         pv[i] = 1 - pchisq(ci, ksq)
##     }
##     if (output) {
##         plot(pv, xlab = "lag", ylab = "p-value", ylim = c(0,
##             1))
##         abline(h = c(0))
##         abline(h = c(0.05), col = "blue")
##         title(main = "Significance plot of CCM")
##     }
##     ccm <- list(ccm = wk, pvalue = pv)
## }
## <environment: namespace:MTS>
```

```
mq(matriz_logrtn)
```

```
## Ljung-Box Statistics:
##          m       Q(m)     df    p-value
##  [1,]     1       156      25         0
##  [2,]     2       202      50         0
##  [3,]     3       252      75         0
##  [4,]     4       265     100         0
##  [5,]     5       298     125         0
##  [6,]     6       318     150         0
##  [7,]     7       349     175         0
##  [8,]     8       408     200         0
##  [9,]     9       442     225         0
## [10,]    10       465     250         0
## [11,]    11       484     275         0
## [12,]    12       513     300         0
## [13,]    13       533     325         0
```

```
## [14,]   14   554   350     0
## [15,]   15   583   375     0
## [16,]   16   651   400     0
## [17,]   17   682   425     0
## [18,]   18   713   450     0
## [19,]   19   744   475     0
## [20,]   20   764   500     0
## [21,]   21   783   525     0
## [22,]   22   811   550     0
## [23,]   23   837   575     0
## [24,]   24   864   600     0
```