Atividade 1 - Análise estatística

Carregando os arquivos do Yahoo! Finance

Vamos carregar as séries de preços de fechamento ajustados de ambas as séries em uma única estrutura de dados de séries temporais chamada `act1Prices.z`. Note que utilizamos o pacote `zoo`, logo é necessário tê-lo carregado previamente.

```
# SP500
sp500.df <- read.csv("GSPC-1962-1994.daily.csv", header = TRUE, stringsAsFactors =
FALSE)
sp500.df$Date <- as.Date(sp500.df$Date)
sp500.df <- sp500.df[order(sp500.df$Date), ]
sp500Prices.df <- sp500.df[, "Adj.Close", drop = FALSE]
rownames(sp500Prices.df) <- sp500.df[, "Date"]
colnames(sp500Prices.df) <- c("SP500")
sp500.prices <- as.zoo(sp500Prices.df)
index(sp500.prices) <- sp500.df[, "Date"]
head(sp500.prices)
```

```
##            SP500
## 1962-01-03 71.13
## 1962-01-04 70.64
## 1962-01-05 69.66
## 1962-01-08 69.12
## 1962-01-09 69.15
## 1962-01-10 68.96
```

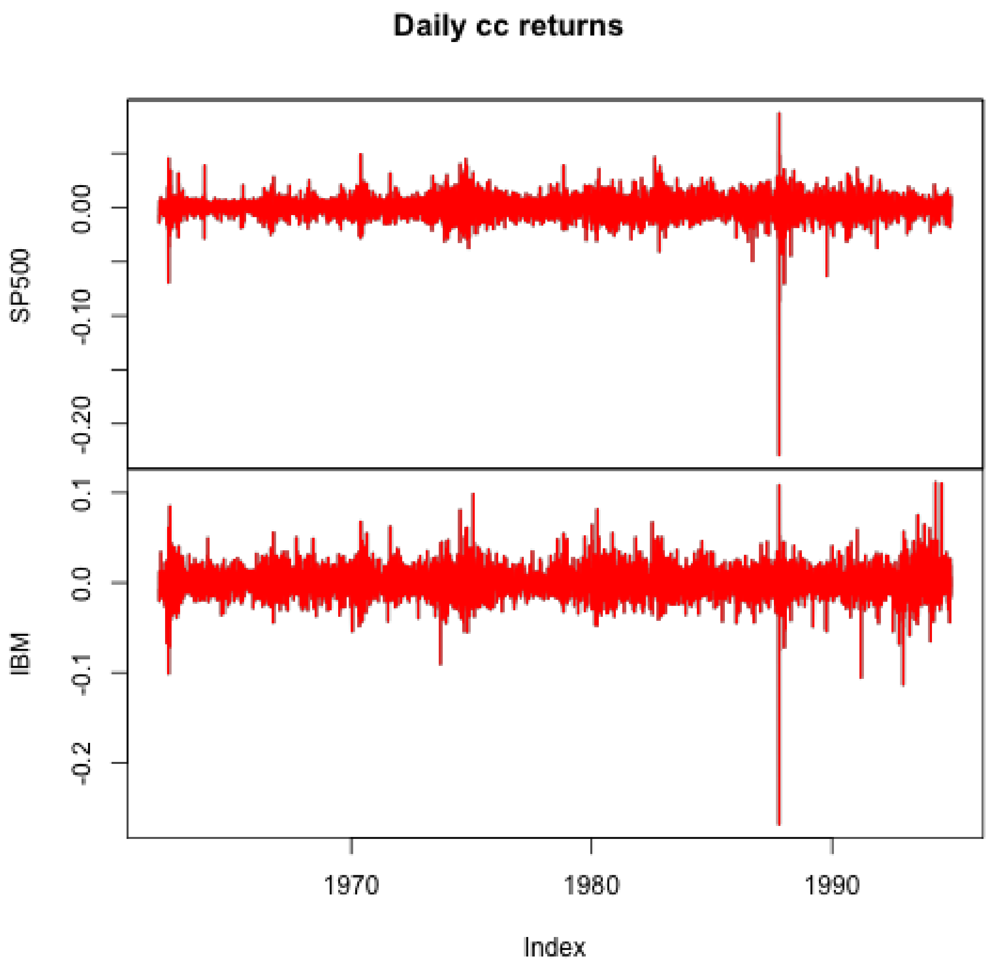
```
# IBM
ibm.df <- read.csv("IBM-1962-1994.daily.csv", header = TRUE, stringsAsFactors = FALSE)
ibm.df$Date <- as.Date(ibm.df$Date)
ibm.df <- ibm.df[order(ibm.df$Date), ]
ibmPrices.df <- ibm.df[, "Adj.Close", drop = FALSE]
rownames(ibmPrices.df) <- ibm.df[, "Date"]
colnames(ibmPrices.df) <- c("IBM")
ibm.prices <- as.zoo(ibmPrices.df)
index(ibm.prices) <- ibm.df[, "Date"]
head(ibm.prices)
```

```
##             IBM
## 1962-01-03 2.56
## 1962-01-04 2.54
## 1962-01-05 2.49
## 1962-01-08 2.44
## 1962-01-09 2.47
## 1962-01-10 2.47
```

```
# Merging series
act1Prices.z <- merge(sp500.prices, ibm.prices)
head(act1Prices.z)
```

```
##            SP500  IBM
## 1962-01-03 71.13 2.56
## 1962-01-04 70.64 2.54
## 1962-01-05 69.66 2.49
## 1962-01-08 69.12 2.44
## 1962-01-09 69.15 2.47
## 1962-01-10 68.96 2.47
```

```
# Calculando retornos
act1Returns.z <- diff(log(act1Prices.z))
plot(act1Returns.z, col = "red", lwd = 2, main = "Daily cc returns")
```



Bem, calculados os retornos, vamos dividir as séries em 4 partes iguais.

```
# Quantidade de linhas
n = nrow(sp500.prices)
# SP500
dim(sp500.prices) <- c(n/4, 4)
colnames(sp500.prices) <- paste("SP500", 1:4)
head(sp500.prices)
```

```
##            SP500 1 SP500 2 SP500 3 SP500 4
## 1962-01-03   71.13   79.44   98.44   234.4
## 1962-01-04   70.64   78.60   99.08   236.7
## 1962-01-05   69.66   77.85   99.54   235.8
## 1962-01-08   69.12   76.53  100.00   235.5
## 1962-01-09   69.15   75.44  100.68   235.9
## 1962-01-10   68.96   76.90  100.66   235.4
```

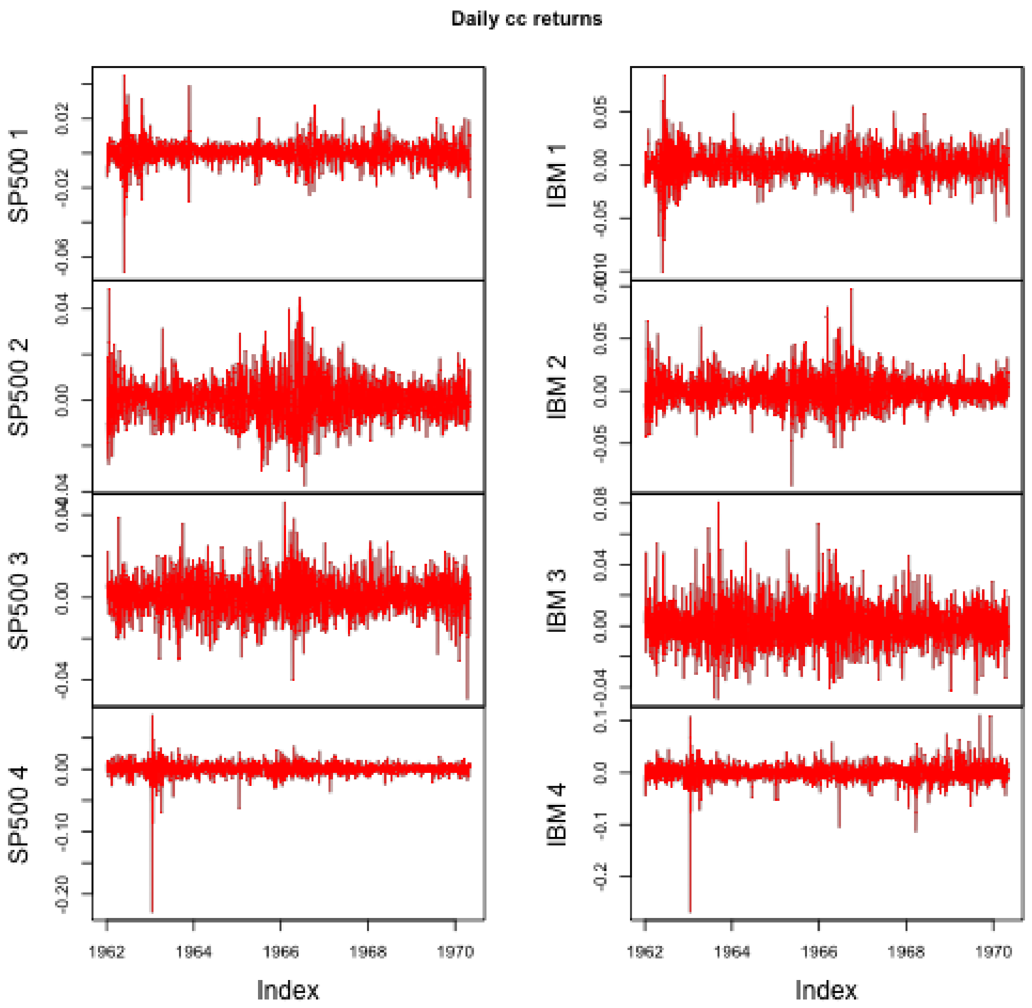
```
# IBM
dim(ibm.prices) <- c(n/4, 4)
colnames(ibm.prices) <- paste("IBM", 1:4)
head(ibm.prices)
```

```
##            IBM 1 IBM 2 IBM 3 IBM 4
## 1962-01-03  2.56  4.98  7.16 19.21
## 1962-01-04  2.54  4.90  7.17 19.11
## 1962-01-05  2.49  4.84  7.24 18.29
## 1962-01-08  2.44  4.72  7.31 18.53
## 1962-01-09  2.47  4.60  7.37 18.29
## 1962-01-10  2.47  4.64  7.32 18.12
```

```
# Merging series
act1partsPrices.z <- merge(sp500.prices[, 1:4], ibm.prices[, 1:4])
head(act1partsPrices.z)
```

```
##            SP500 1 SP500 2 SP500 3 SP500 4 IBM 1 IBM 2 IBM 3 IBM 4
## 1962-01-03   71.13   79.44   98.44   234.4  2.56  4.98  7.16 19.21
## 1962-01-04   70.64   78.60   99.08   236.7  2.54  4.90  7.17 19.11
## 1962-01-05   69.66   77.85   99.54   235.8  2.49  4.84  7.24 18.29
## 1962-01-08   69.12   76.53  100.00   235.5  2.44  4.72  7.31 18.53
## 1962-01-09   69.15   75.44  100.68   235.9  2.47  4.60  7.37 18.29
## 1962-01-10   68.96   76.90  100.66   235.4  2.47  4.64  7.32 18.12
```

```
act1partsReturns.z <- diff(log(act1partsPrices.z))
plot(act1partsReturns.z, col = "red", lwd = 1, main = "Daily cc returns")
```



Estatísticas em retornos simples

Infelizmente uma coisa *tricky*, as funções do core do R não funcionam adequadamente com objetos zoo de séries temporais, logo, devemos fazer uma cópia do objeto em forma de matriz.

```
# Serie completa
ret.mat <- coredata(act1Returns.z)
class(ret.mat)
```

```
## [1] "matrix"
```

```
colnames(ret.mat)
```

```
## [1] "SP500" "IBM"
```

```
head(ret.mat)
```

```
##            SP500       IBM
## [1,] -0.0069126 -0.007843
## [2,] -0.0139703 -0.019881
## [3,] -0.0077821 -0.020285
## [4,]  0.0004339  0.012220
## [5,] -0.0027514  0.000000
## [6,]  0.0059279  0.012073
```

```
# Para as partes
retparts.mat <- coredata(act1partsReturns.z)
```

Note que a estrutura de ordenação e colunas é mantida.

```
# Estatisticas para as series completas
apply(ret.mat, 2, mean)
```

```
##     SP500       IBM
## 0.0002247 0.0002086
```

```
apply(ret.mat, 2, sd)
```

```
##    SP500      IBM
## 0.008814 0.014511
```

```
myacf <- function(x) {
    acf(x, lag.max = 1, type = "correlation", plot = FALSE)
}
apply(ret.mat, 2, myacf)
```

```
## $SP500
##
## Autocorrelations of series 'x', by lag
##
##     0     1
## 1.000 0.125
##
## $IBM
##
## Autocorrelations of series 'x', by lag
##
##      0      1
##  1.000 -0.005
```

```
# Para as partes
apply(retparts.mat, 2, mean)
```

```
##    SP500 1    SP500 2    SP500 3    SP500 4      IBM 1      IBM 2 
##  5.566e-05  9.991e-05  4.193e-04  3.239e-04  3.200e-04  1.718e-04 
##      IBM 3      IBM 4 
##  4.958e-04 -1.377e-04
```

```
apply(retparts.mat, 2, sd)
```

```
##  SP500 1  SP500 2  SP500 3  SP500 4    IBM 1    IBM 2    IBM 3    IBM 4 
## 0.006487 0.008745 0.008828 0.010693 0.013117 0.013967 0.013716 0.016929
```

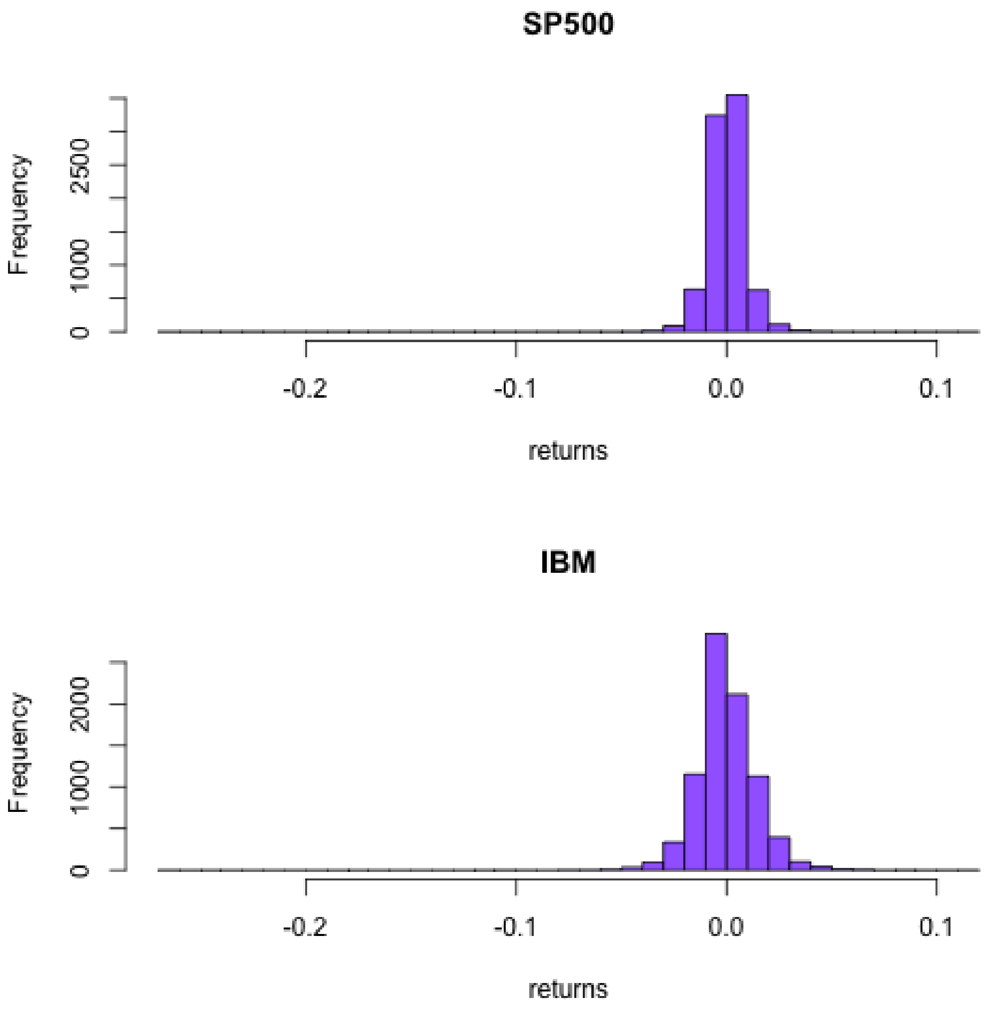
```
apply(retparts.mat, 2, myacf)
```

```
## $`SP500 1`
## 
## Autocorrelations of series 'x', by lag
## 
##     0     1 
## 1.000 0.183 
## 
## $`SP500 2`
## 
## Autocorrelations of series 'x', by lag
## 
##     0     1 
## 1.000 0.268 
## 
## $`SP500 3`
## 
## Autocorrelations of series 'x', by lag
## 
##     0     1 
## 1.000 0.097 
## 
## $`SP500 4`
```

```
##
## Autocorrelations of series 'x', by lag
##
##     0     1
## 1.000 0.027
##
## $`IBM 1`
##
## Autocorrelations of series 'x', by lag
##
##     0     1
## 1.000 0.034
##
## $`IBM 2`
##
## Autocorrelations of series 'x', by lag
##
##     0     1
## 1.000 0.067
##
## $`IBM 3`
##
## Autocorrelations of series 'x', by lag
##
##      0      1
##  1.000 -0.077
##
## $`IBM 4`
##
## Autocorrelations of series 'x', by lag
##
##     0     1
##  1.00 -0.03
```

Histogramas

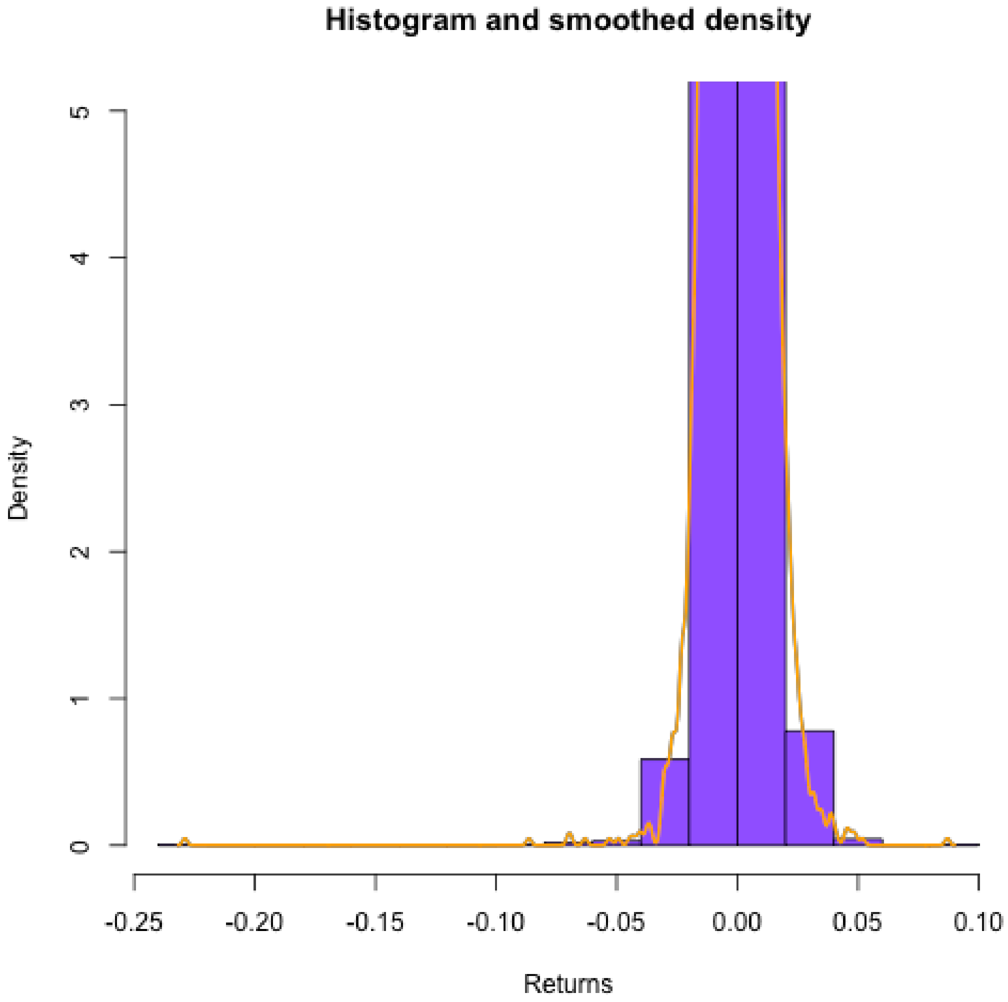
```
IBM.hist = hist(ret.mat[, 2], plot = FALSE, breaks = 30)
par(mfrow = c(2, 1))
hist(ret.mat[, 1], main = "SP500", col = "slateblue1", xlab = "returns", breaks =
IBM.hist$breaks)
hist(ret.mat[, 2], main = "IBM", col = "slateblue1", xlab = "returns", breaks =
IBM.hist$breaks)
```



```
par(mfrow = c(1, 1))
```

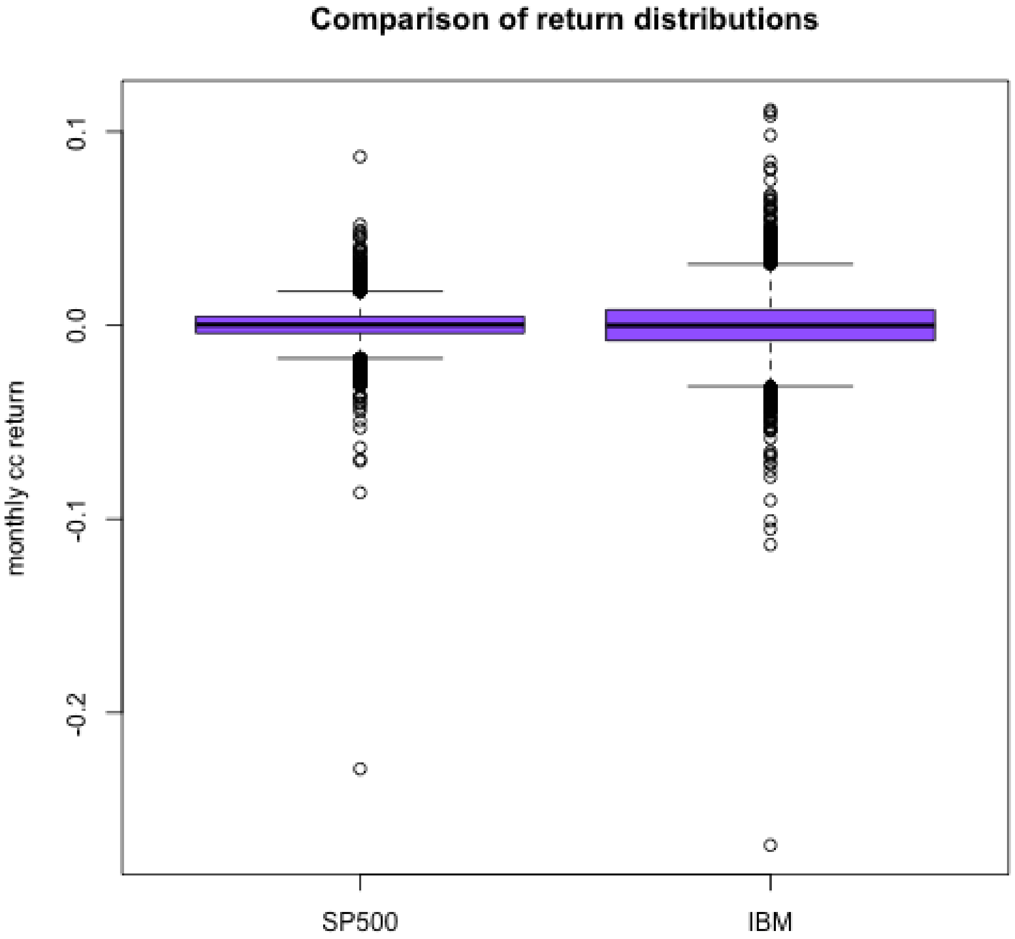
Histogramas *alisados* - `density`

```
SP500.density = density(ret.mat[, 1])
hist(ret.mat[, 1], main = "Histogram and smoothed density", col = "slateblue1",
    probability = TRUE, ylim = c(0, 5), xlab = "Returns")
points(SP500.density, type = "l", col = "orange", lwd = 2)
```


Histogram and smoothed density
Density
0
1
2
3
4
5
-0.25
-0.20
-0.15
-0.10
-0.05
0.00
0.05
0.10
Returns

Boxplot

```
boxplot(ret.mat[, 1], ret.mat[, 2], names = c("SP500", "IBM"), outchar = TRUE,
    col = "slateblue1", main = "Comparison of return distributions", ylab = "monthly
cc return")
```



Assimetria e kurtose

As funções *skewness* e *kurtose* pertencem ao pacote `PerformanceAnalytics`.

```
# Estatisticas para as series completas
apply(ret.mat, 2, skewness)
```

```
##   SP500     IBM
## -2.1937 -0.6042
```

```
apply(ret.mat, 2, kurtosis)
```

```
## SP500   IBM
## 60.27 18.34
```

```
# Para as partes
apply(retparts.mat, 2, skewness)
```

```
##  SP500 1  SP500 2  SP500 3  SP500 4    IBM 1    IBM 2    IBM 3    IBM 4
## -0.48146  0.26874  0.07835 -5.02392 -0.25401  0.36514  0.42704 -1.80938
```

```
apply(retparts.mat, 2, kurtosis)
```

```
## SP500 1 SP500 2 SP500 3 SP500 4   IBM 1   IBM 2   IBM 3   IBM 4
##  10.130   2.161   1.861 107.715   4.430   3.756   1.837  35.025
```