Análise de Carteiras usando o R

Bibliografia – BKM, cap. 5

Claudio Lucinda

FEA-RP/USP

Iniciando

Iniciando

- Vamos utilizar o R para fazer as análises de carteiras
- Acredito que é uma boa escolha, apesar de ter prós e contras:
- Prós
  - É mais flexível, tanto em termos de análise quanto em termos de fazer interface com outros softwares (o que significa um pouco menos de trabalho prévio).
  - Automaticamente documenta o que você está fazendo, o que significa que você pode rapidamente resolver qualquer problema e não fica perdido tentando lembrar todos os passos que você fez para chegar em um resultado.
  - É de graça
- Contras
  - É menos transparente. Nem sempre você consegue ver tudo da mesma forma que no Excel
  - Tem uma curva de aprendizado mais íngreme do que o Excel.

O que é necessário

- Ponto de partida: ter o R instalado no seu computador.
- Caso não tenha, recomendo que vá ao site e baixe a versão mais recente para o seu computador.
- IDE (*Integrated Development Environment*): Se você abrir o R, já dá pra fazer tudo. Só que é super desajeitado, na minha opinião, e existem muitas interfaces que facilitam muito o processo de elaboração dos códigos.
- A que utilizarei nestas aulas é o RStudio, que acho a mais flexível e legal das plataformas.

Trabalhando com o R

- Existem alguns arquivos que você verá no seu trabalho
- **R scripts** - São arquivos com a extensão R e que representam os seus códigos.
- **RData** - São arquivos que incluem os dados a serem utilizados.
- Dentro do R existem os chamados **environments** que são aonde os diferentes objetos estão acumulados.
- Especificamente com respeito ao nosso caso, devemos ter objetos da classe `data.frame`, da classe `xts` além de funções.
- Nota: Os objetos da classe `data.frame` e da classe `xts` são bancos de dados e, ao menos que você explicitamente especifique pra que ele esteja na memória com o comando `attach`, você precisa referenciar as variáveis em um banco com o cifrão.

Instalando Pacotes

- A maior fonte de flexibilidade do R está na sua enorme biblioteca de funções e funcionalidades contribuídas por outros pesquisadores.
- Para isso, é necessário o uso do comando `install.packages`. No nosso caso, utilizaremos bastante os pacotes `PortfolioAnalytics`, `quantmod`, `BatchGetSymbols` e `xlsx`.
- Com uma conexão à Internet, isso pode ser implementado da seguinte forma:

```
install.packages("quantmod")
install.packages("BatchGetSymbols")
install.packages("xlsx")
install.packages("PortfolioAnalytics")
```

libraries

- Instalar o pacote não significa que ele esteja automaticamente disponível.
- Para isso, é necessário o comando `library`, que disponibiliza as funções no seu ambiente global para sua análise

```
library(BatchGetSymbols)
library(PortfolioAnalytics)
library(xlsx)
library(quantmod)
```

- Algumas observações:
- Se você quer rodar apenas algumas linhas do seu arquivo R, é só usar `Ctrl+Enter`.
- O pacote `xlsx` permite importar e exportar dados em `xlsx`. Só que você precisa ter uma versão atualizada do Java no seu computador. Se não funcionar, entre no site do Java e atualize.

Várias Coisas

- Em muitos casos, é importante que você tenha certeza aonde o R salva as coisas – e vc precisa ter certeza também de onde os dados estão.
- Para saber a qual diretório o R está se referindo, é importante usar o comando `getwd()`. Se você quiser mudar este diretório, é o comando `setwd()`. Só que os endereços dos arquivos seguem a norma do Linux, com a barra ao contrário.
- Para limpar o ambiente no começo de um arquivo, o comando é `rm(ls=())`.
- Para limpar os gráficos, o comando é `graphics.off()`
- Se você quiser, no RStudio dá inclusive para fazer as apresentações integradas com o código. Para mais informações, procure por `RMarkdown`

Começando a Mexer com Dados Financeiros

Mexendo com os dados

- Vamos começar.
- O Arquivo em que estamos mexendo agora é o `Initial_Code.R` que está no mesmo GitHub que vocês baixaram este arquivo.
- Evidentemente, ele tem um monte de coisas que são muito específicas do meu jeito de fazer código. Vale a pena descobrir o seu.
- Eu começo com o básico, já supondo que os códigos estão instalados:

```
library(PortfolioAnalytics)
library(xlsx)
library(tidyverse)

rm(list=ls())
graphics.off()
```

Baixando os dados

- Eu crio uma função `conv_ts`, que transforma um `data.frame` em um `xts`
- Depois eu baixo os dados

```
PETR4<-read.xlsx("./Data01.xlsx",sheetName="PETR4")
BBAS3<-read.xlsx("./Data01.xlsx",sheetName="BBAS3")
IBOV<-read.xlsx("./Data01.xlsx",sheetName="IBOV")
CDI<-read.xlsx("./Data01.xlsx",sheetName="CDI")
PETR4[,3:4]<-NULL
PETR4<-PETR4[!is.na(PETR4$Data),]
```

Calculando os Retornos

- Calcular os retornos fica muito mais fácil com a função `Return.Calculate` do `PortfolioAnalytics`:
- Retiramos a primeira linha porque ela é sempre composta por `NA`

```
# Ações Individuais
ret_PETR4<-Return.calculate(PETR4)[-1,]
ret_BBAS3<-Return.calculate(BBAS3)[-1,]
ret_IBOV<-Return.calculate(IBOV)[-1,]
ret_CDI<-Return.calculate(CDI)[-1,]
```

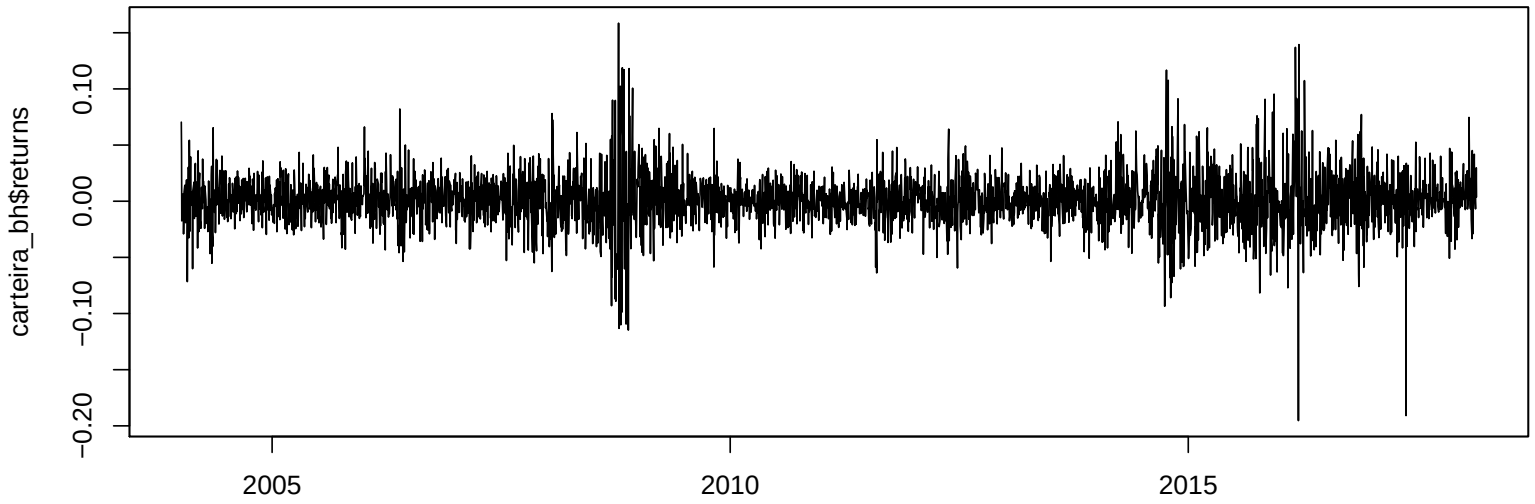
Avaliando o Retorno das Carteiras

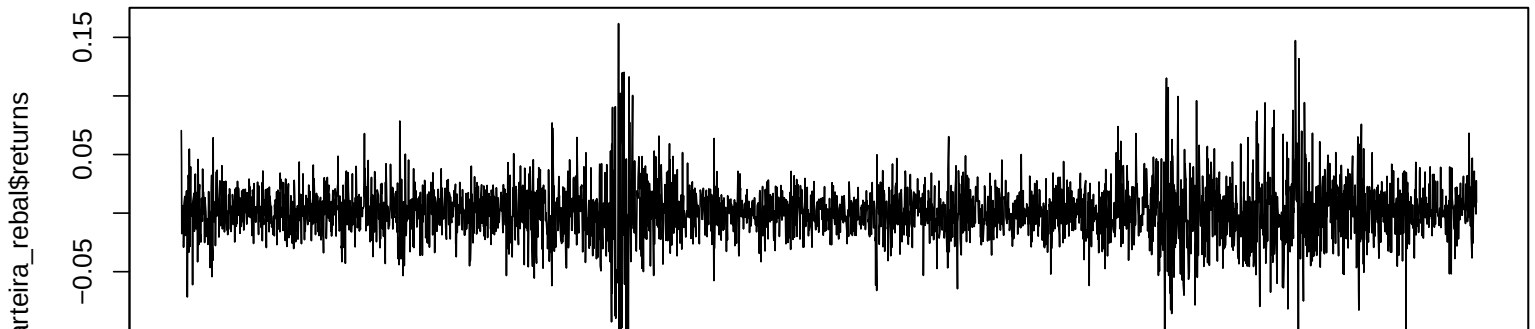
```
carteira<-merge.xts(ret_PETR4,ret_BBAS3)
peso_inicial<-c(.5,.5)

carteira_bh<-Return.portfolio(carteira,
                              weights=peso_inicial,
                              verbose=TRUE)
carteira_rebal<-Return.portfolio(carteira,
                                 weights=peso_inicial,
                                 rebalance_on="months",
                                 verbose=TRUE)
```

Retornos – Carteira Rebalanceada e não Rebalanceada

```
par(mfrow = c(2, 1), mar = c(2, 4, 2, 2))
plot.zoo(carteira_bh$returns)
plot.zoo(carteira_rebal$returns)
```

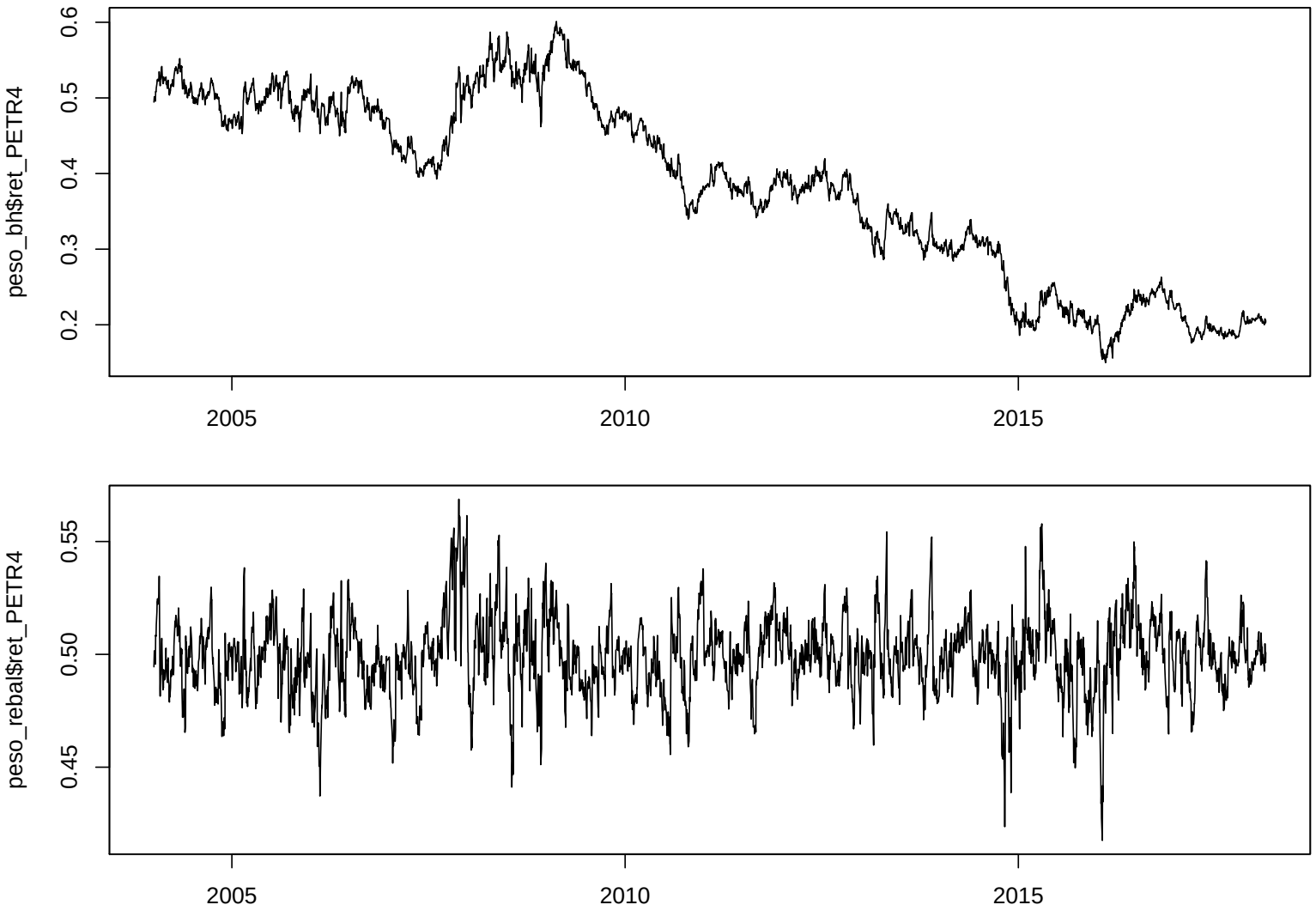




Pesos dos dois ativos na carteira

```
peso_bh<-carteira_bh$EOP.Weight
peso_rebal<-carteira_rebal$EOP.Weight
par(mfrow = c(2, 1), mar=c(2, 4, 2, 2))
plot.zoo(peso_bh$ret_PETR4)
plot.zoo(peso_rebal$ret_PETR4)
```

Pesos dos dois ativos na carteira – Graficamente



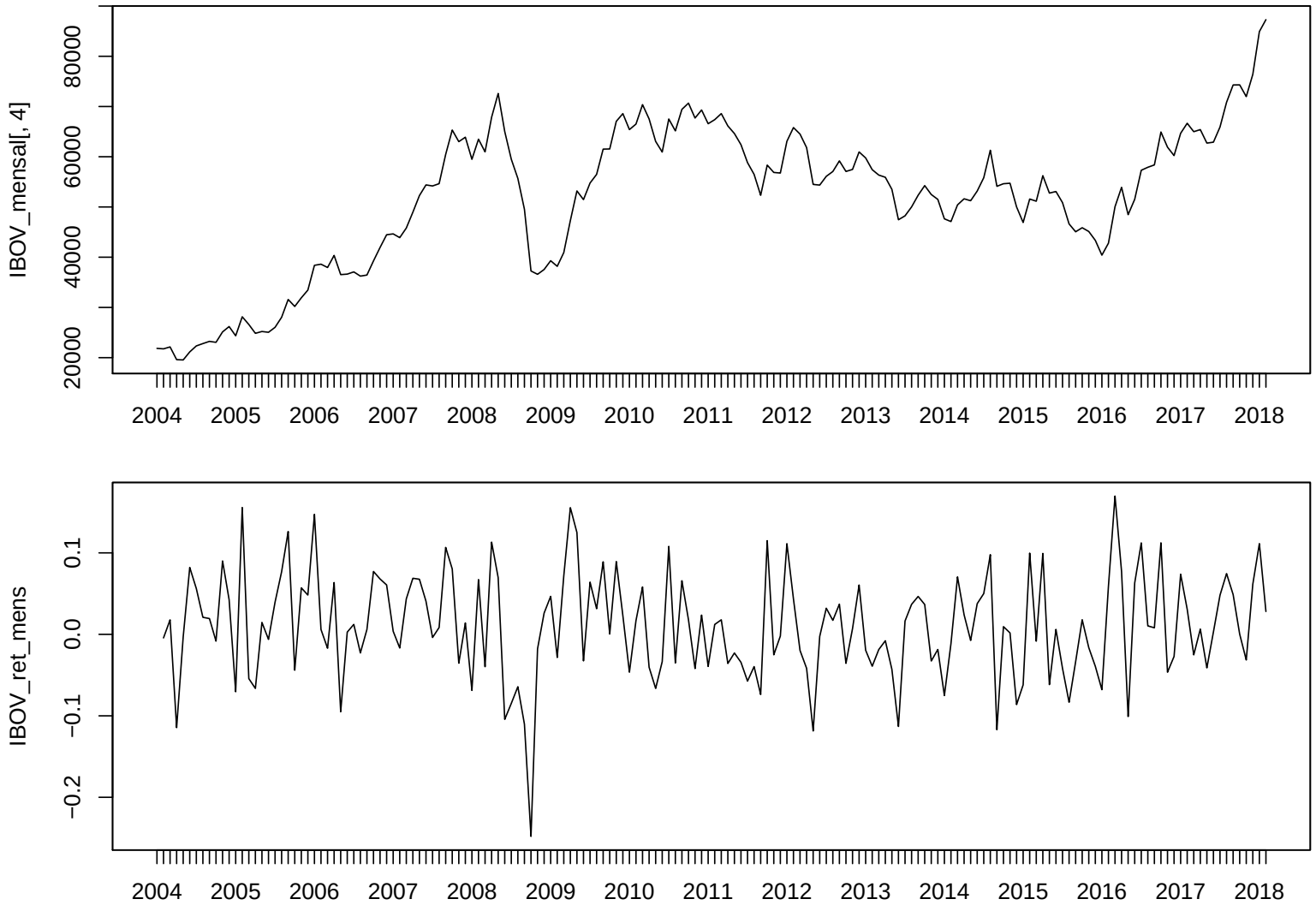
Analisando Risco e Retorno

Usando o Ibovespa

- Vamos fazer esta análise de risco e retorno utilizando os dados históricos do IBOVespa nos últimos 14 anos.
- Baixamos antes estes dados do Economática, mais adiante mostro um jeito de baixar diretamente da Internet.
- Inicialmente, olhando os retornos mensais a partir dos dados diários que já puxamos:

```
IBOV_mensal<-to.monthly(IBOV)
IBOV_ret_mens<-Return.calculate(IBOV_mensal[,4])
# Fechamento Mensal
plot.zoo(IBOV_mensal[,4])
# Retornos mensais
plot.zoo(IBOV_ret_mens)
```

Graficamente



Retornos Médios Aritméticos e Geométricos

- Quanto maior a volatilidade dos retornos, maior a diferença entre a média aritmética e a geométrica.
- Se os retornos vierem de uma distribuição normal, a diferença entre as duas médias é igual à metade do desvio-padrão, ou seja:

$$E(r_G) = E(r_A) - \frac{1}{2}\sigma^2$$

- Retornos geométricos também são conhecidos como *time weighted returns* porque cada instante passado do tempo ganha um peso igual.
- Retornos aritméticos podem ser uma estimativa do retorno esperado prospectivo, imaginando que cada retorno observado é um *draw* da distribuição subjacente dos retornos.

Retornos Médios Aritméticos e Geométricos

```
#Retornos Médios - Aritméticos
mean(IBOV_ret_mens, na.rm=TRUE)
```

```
## [1] 0.01029328
```

```
#Retornos Médios - Geométricos
mean.geometric(IBOV_ret_mens)
```

```
##                    IBOV.Close
## Geometric Mean 0.008229097
```

```
sd(IBOV_ret_mens,na.rm=T)
```

```
## [1] 0.06426524
```

Variância e Desvio-Padrão para Frequencias diferentes

- Não é a frequencia das observações que garantem melhores estimativas da média, e sim a duração da série de tempo.
- Isso nos dá a seguinte dica sobre os dados. Sempre utilize a amostra mais longa que você conseguir, desde que ela venha da mesma distribuição.
- Agora, estimativas mais precisas da variância podem ser conseguidas com amostras mais frequentes.

– Quando as observações são não correlacionadas, as variâncias estimadas para períodos mais longos são simplesmente a soma das variâncias estimadas para períodos mais curtos.

– Ou seja, a variância anual é 12 vezes a variância mensal: $\sigma_A^2 = 12 \times \sigma_M^2$

– O que implica que o desvio padrão cresce a um termo $\sqrt{T}$.

Índice de Sharpe

- Uma medida de retorno por unidade de risco
- Calculado como:

$$IS = \frac{E(r_P) - E(r_f)}{\sigma_P}$$

```
SharpeRatio.annualized(ret_IBOV,
                       Rf=mean(ret_CDI,na.rm=T),scale=12)
```

```
##                                            [,1]
## Annualized Sharpe Ratio (Rf=0.5%) -0.01294254
```

Índice de Sharpe – Base Diária

- O anualizado é bom para comparar diferentes fluxos de recursos. Mas você pode querer o Sharpe dos dados brutos

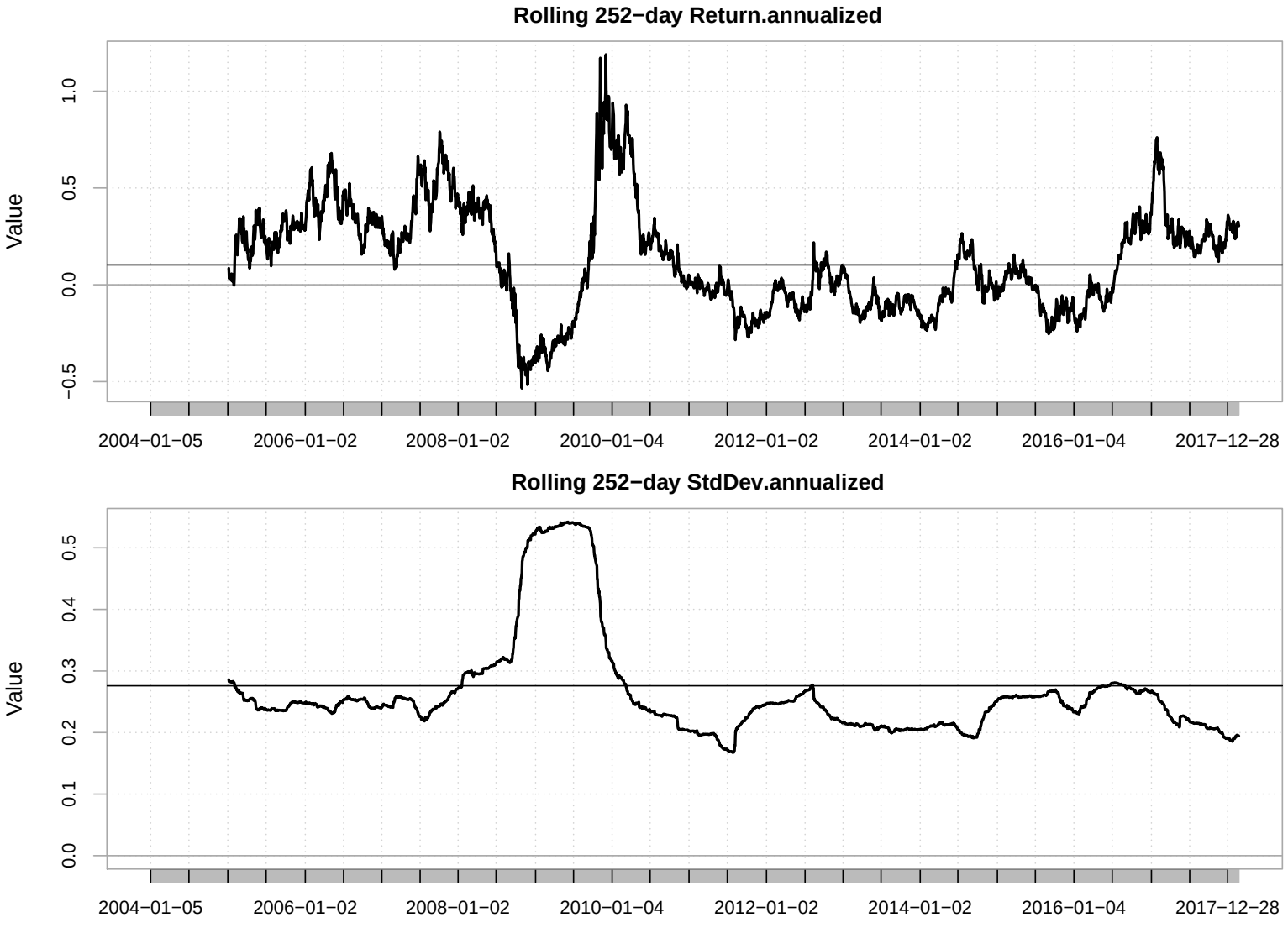
```
SharpeRatio(ret_IBOV,
            Rf=mean(ret_CDI,na.rm=T), FUN="StdDev")
```

```
##                                          [,1]
## StdDev Sharpe (Rf=0%, p=95%): 0.004948583
```

Calculando as medidas de desempenho em uma janela móvel de 252 dias

```
IBOV_anual<-Return.annualized(ret_IBOV,scale=252)
CDI_anual<-Return.annualized(ret_CDI,scale=252)
par(mfrow = c(2, 1), mar=c(2, 4, 2, 2))
chart.RollingPerformance(R = ret_IBOV,
                         width = 252,
                         FUN = "Return.annualized")
abline(h = IBOV_anual)
chart.RollingPerformance(R = ret_IBOV,
                         width = 252,
                         FUN = "StdDev.annualized")
abline(h = sd_IBOV_anual)
```

Graficamente



Índice de Sharpe

```
sharpe_IBOV_anual<-SharpeRatio.annualized(ret_IBOV,
                                          Rf=ret_CDI,
                                          scale=252)
chart.RollingPerformance(R = ret_IBOV, width = 252,
                         FUN = "SharpeRatio.annualized",
                         Rf=ret_CDI)
abline(h = sharpe_IBOV_anual)
```

Graficamente