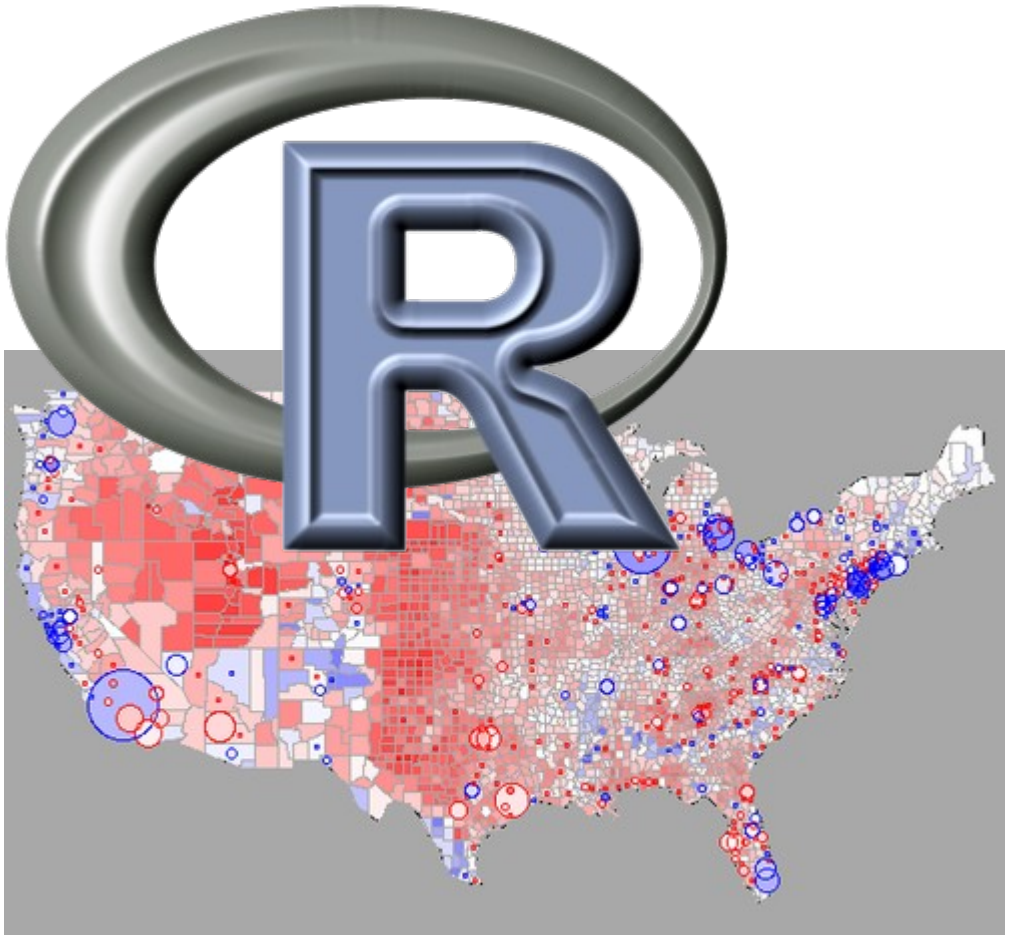
FIAP

PÓS-GRADUAÇÃO



MBA em Big Data

Introdução à Linguagem R

Prof. Antonio Henrique Pinto Selvatici

antoniohps@gmail.com

Versão 1 – 10/2014

Professor

- **Antonio Henrique Pinto Selvatici**

- É engenheiro eletrônico formado pelo Instituto Tecnológico de Aeronáutica (ITA), com mestrado e doutorado pela Escola Politécnica (USP), e passagem pela Georgia Institute of Technology em Atlanta (EUA). Desde 2002, atua na indústria em projetos nas áreas de robótica, visão computacional e internet das coisas, aliando teoria e prática no desenvolvimento de soluções baseadas em Machine Learning, processamento paralelo e modelos probabilísticos, tendo desenvolvidos projetos para Avibrás, Rede Globo, IPT e Systax. Foi professor do curso de Ciência da Computação da Uninove de 2009 a 2013. Em 2012, tendo ajudado a fundar a Selsantech, participou do desenvolvimento do CatSearch, uma solução para Data Mining preparada para o paradigma MapReduce. É professor do MBA do curso de Big Data da FIAP e trabalha na reformulação do sistema de pagamento on-line eWally.

O que esperar do curso

- Primeiro contato com a linguagem R
- Ter noções básicas de programação em R, tendo como base uma prévia noção de programação em outra linguagem
- Conhecer o potencial dessa linguagem e entender por que ela está sendo muito adotada no desenvolvimento de soluções de Big Data Analytics
- Tomar contato com alguns pacotes (bibliotecas) mais populares
- Aprender a usar pacotes de processamento paralelo

Organização do curso

- Encontro 1: Breve tutorial
  - Um pouco de história
  - Instalação e utilização do sistema (R + IDE)
  - Estruturas de dados e chamadas de funções
  - Básico sobre visualização
- Encontro 2: Visualização de dados
  - Outras estruturas de dados
  - Sintaxe da plotagem
  - Pacotes gráficos
- Encontro 3: Escrevendo scripts
  - Escrevendo e executando scripts
  - Estruturas de programação
  - Escrevendo funções
- Aula 4: Processamento paralelo
  - Interação com Hadoop e Cuda
  - Atividade prática

Avaliação da Disciplina

- Presença: 4,0
- Entrega de listas de exercícios: 3,0
- Atividade final: 3,0

1. Breve Tutorial de R

Resumo

- O que é R?
- Um pouco de história
- Instalação do software
- Executando e entendendo RStudio
- Hello World!
- Variáveis e tipos de dados
- Estruturas de dados fundamentais
- Importando tabelas de dados
- Visualização de dados
- Lista de exercícios

1 – Sobre a linguagem R

- R é uma linguagem de programação
  - Interpretada, de alto nível
  - Orientada a objetos, funcional
  - Extensível, através de adições de pacotes ou bibliotecas
  - Chamadas a funções de C e Fortran de forma simples
- R é um ambiente interativo de comando
  - Executa uma grande variedade de métodos estatísticos
  - Foram criadas IDEs que facilitam o uso desse ambiente (Rstudio, RCommander, etc)
  - Lembra outras plataformas, como SAS e Matlab (ou Octave)
- R é uma plataforma gráfica
  - Produz gráficos de alta qualidade
- R é *open source*

2 – Um pouco de história

- 1976 – Início do desenvolvimento da linguagem S
  - Já existiam SAS e SPSS
  - Bell Labs (desenvolvimento para uso interno)
  - Rick Becker, John Chambers and others
- 1981 – S versão 2
  - Rodando em Unix
  - Licenciamento para uso externo
- 1983 a 1992 – S versão 3 (S3)
  - Reformulação da linguagem
  - Orientação a objetos (tudo é objeto)
- 1995 – Início do desenvolvimento do R
  - Ross Ihaka & Robert Gentleman (Universidade de Auckland)
  - Open source
- 1995 a 1998 – S versão 4 (S4)
  - Algumas melhorias (classes com metadata, conectores)
  - Implementada em R e pelo S-Plus (Insightful, comprada pela TIBCO)

4 – Instalando R

- Site oficial: http://www.r-project.org/
- O ambiente interativo de R pode ser instalado em Linux, Mac ou Windows
- Possui algumas IDEs:
  - RCommander (www.rcommander.com)
  - RStudio (www.rstudio.com)
  - Revolution R Enterprise (http://www.revolutionanalytics.com/revolution-r-enterprise-developr)
  - Eclipse + StatET
- Instalação do RStudio (mais popular – vamos usar no curso):
  - Instalamos primeiro o ambiente do R
  - Depois, instalamos o RStudio
  - Possui pacotes prontos para Linux (.deb) e (.rpm)

5 – Hello World!

- Executar um comando no Rstudio pela primeira vez
  - File -> New File -> R Script
  - Digitar “Hello World!” (com as aspas) -> veja a saída
  - Teclar CTRL-Enter
  - Para apagar a saída, teclar CTRL-L
- As expressões a serem executadas são transportadas para o console. Dependendo da expressão, o sistema pode responder através de output no próprio console ou através de uma janela gráfica
- Selecionando mais de uma linha, elas são executadas juntas
- Podemos digitar diretamente no console, mas o controle sobre a sequência de comandos seria mais complicado. Manter a lista de comandos em separado é muito, como vamos ver...
- O ambiente interativo funciona como uma calculadora. Vamos executar algumas contas:
  - Ex: 5+3, 9/2, 4.24*3.13

6 – Expressões aritméticas e numéricas

- Operadores binários: +, -, *, /, ^ ou ** (exponenciação), %% (módulo)
- Operadores lógicos: >, >=, <, <=, ==, !=, identical()
- Funções matemáticas: abs, sqrt, log, exp, log10, factorial
- Funções trigonométricas: sin, cos, tan, asin, acos, atan
- Arredondamento: round, ceiling, floor, trunc, signif
- Quantidades: Inf, -Inf, NaN (not a number), pi, exp(1), 1i (unidade imaginária)
- Exemplos:
  - 5 %% 4
  - log(2)
  - cos(pi)
  - ceiling(3.2)
  - 0/0
  - 1/Inf
  - 1 ==2
  - 3 > -4

7 – Chamadas de funções

- Funções são objetos especiais do R que processam argumentos e podem gerar um valor de saída
- A chamada é feita pelo nome da função seguida dos argumentos entre parênteses.
- Há argumentos obrigatórios e argumentos opcionais.
  - Os obrigatórios devem ser inseridos na ordem em que foram definidos
  - A “novidade” é que os argumentos opcionais podem ser inseridos de duas formas:
    - » Na ordem em que foram definidos, OU
    - » Através da atribuição do nome do parâmetro através do operador “=“
    - » Os parâmetros de uma função podem ser observados através da função “args(name)”
- Exemplos:
  - args(matrix)
  - matrix(c(1,2,3,4,5,6),3)
  - matrix(c(1,2,3,4,5,6),ncol=3)

8 – Variáveis

- Variáveis em R são similares às variáveis das diferentes linguagens de programação interpretadas
  - Não-tipadas: o tipo da variável não precisa ser declarado e é definido na atribuição de valores
  - A variável é sempre uma referência para um objeto na memória (lembrando que tudo é objeto)
  - O operador de atribuição pode ser “=“ ou “<-”
  - Variáveis são sensíveis à caixa e não podem iniciar por números nem conter caracteres especiais
  - Curiosamente, não podem iniciar por “_”, mas podem conter “.” na grafia
- Executar:
  - x
  - x.1 <- 5+3
  - y2 = x.1 – 3
  - x.1 + y2

9 – Tipos de dados

- **Numeric**: dados numéricos, que, tecnicamente podem ser inteiros ou ponto flutuante (double), mas quase sempre correspondem ao último caso
- **Logical**: podendo assumir os valores TRUE ou FALSE (T ou F), são o resultado de operações lógicas
- **Character**: são as strings, definidas por texto entre ‘ ‘ ou “ “
- **Factor**: Variável categórica que pode assumir um valor dentre uma gama de “níveis” definidos. Lembra as enumerações da linguagem C.
- **Date**: variável indicando tempo
  - R possui objetos que representam apenas datas ou data e hora, dependendo de valor inicializado. Geralmente na forma “YYYY-MM-DD” (classe Date)
  - as.Date(“2014-10-01”)
  - ISOdatetime(1960,1,1,23,0,59)
- **Missing Data (NA):** variável especial que indica a falta de um valor definido

10 – Conjuntos (estruturas) de dados

- Sendo R um ambiente voltado para processamento estatístico, nosso maior interesse é trabalhar com conjuntos de dados
- Os conjuntos de dados suportados pelo R são:
  - Vetores: sequência de valores do mesmo tipo
  - Matrizes: valores do mesmo tipo organizados por linhas e colunas
  - Arrays: são generalizações das matrizes para várias dimensões
  - Listas: coleções de dados de diversos tipos
  - Dataframes: conjuntos de dados organizados como uma tabela de banco dados
- Para o nosso curso, as estruturas de dados de maior interesse são os vetores, matrizes e os dataframes

11 - Vetores

- São o tipo mais básico de estrutura de dados
- A função c(...) concatena seus argumentos para formar um vetor
  - c(1,3.4,-2.9,3)
  - c("A", "B", "C", "D")
- Para criar vetores com um padrão, podemos usar:
  - : (dois pontos) – cria sequências com incrementos ou decrementos de 1
    - » 2.3:5
  - seq() – cria uma sequência genérica
    - » seq(0, 2, by=0.5)
    - » seq(0, 3, len=6)
  - rep() – cria um vetor de réplicas
    - » rep(1:5, each=2)
    - » rep(1:5, times=2)

11 – Vetores >> Operações com vetores

- Para saber o tamanho de um vetor, usamos a função “length(x)”
- As operações e funções lógico-aritméticas são indistintamente aplicadas a vetores ou a quantidades escalares, de forma que estas podem ser encaradas como um “vetor unidimensional”
  - log(2)
  - log(c(1,2,exp(1),4))
  - c(1,2,3,4,5,6) > 3
  - c(1,2,3,4,5,6) > c(0,3)
- Podemos aplicar as operações aritméticas a dois vetores, que as executaram elemento a elemento.
  - c(2,3,4) + c(3,2,1)
  - c(2,2,2) ^(1:3)
- Caso os vetores sejam de tamanhos diferentes, o menor vetor tem seus elementos repetidos desde o começo
  - c(30,40,50,60) + c(1,2)

11 – Vetores >> acessando seus elementos

- A linguagem R é muito flexível no acesso a elementtos de um vetor.
- Através de colchetes [ ] podemos acessar um ou mais elementos
- Acesso a um único elemento:
  - x[4]
- Acesso a um conjunto de valores através de um vetor de índices:
  - x[1:3]
  - x[c(2,5,6)]
- Acesso a todos os elementos exceto os indicados através de índices negativos
  - x[-3]
  - x[-c(4,5)]
- Usando um vetor lógico para acessar os elementos
  - x[x>4]

11 – Vetores >> buscando elementos

- Há funções que fazem a busca em vetores e retornam seus índices
- **which()** : Retorna os índices de um vetor lógico onde os valores são TRUE
- **which.max() :** Retorna o índice da primeira ocorrência do valor máximo
- **which.min() :** Retorna o índice da primeira ocorrência do valor mínimo
  - A diferença para max() e min() é que eles retornam os valores máximo e mínimo do vetor
- **match() :** Retorna o índice da primeira ocorrência de um valor
- x <- c(4, 7, 2, 10, 1, 0)
- x>=4
- which(x>=4)
- which.max(x)
- x[which.max(x)]
- max(x)

11 – Vetores >> outras funções úteis

- **sum(x) / prod(x):** Soma/multiplicação dos elementos
- **cumsum(x) / cumprod(x):** Soma/multiplicação cumulativa dos elementos
- **min(x) / max(x):** Retorna o valor mínimo/máximo do vetor
- **mean(x) / median(x):** Retorna a média / mediana do vetor
- **range(x)**: Retorna o range (mínimo e máximo) valores do vetor
- **length(x)**: Retorna o número de elemtnos do vetor
- **unique(x):** Retorna um vetor sem repetições de valores
- **rev(x):** Reverte a ordem dos valores do vetor
- **sort(x):** Ordena os elementos do vetor
- **union(x, y)**: Faz a união dos vetores x e y
- **intersect(x, y):** Faz a intersecção dos vetores x e y
- **x %in% y:** para cada elemento de x, verifica se está presente no vetor y.
- **setdiff(x, y):** Retorna os elementos de x que não estão em y
- **setequal(x, y):** Informa se os vetores x e y possuem os mesmos elementos

12 – Data Frames

- São as “estruturas de dados” que guardam “dados estruturados”, semelhantes a uma tabela de um BD.
- Permitem a importação e exportação de dados com facilidade
- Guardam os dados inteiramente na RAM do computador, o que pode comprometer o desempenho do software
- O ambiente do R vem com vários conjuntos de dados de exemplo. Vamos usar o “state.x77”
  - data <- data.frame(state.x77)
- Ao clicar duplamente na variável “data”, podemos visualizar o data set, que está na forma de um data frame
- Vejam que ele possui linhas e colunas nomeadas, bem como um conjunto de dados numérico
- As colunas possuem dados de mesma natureza, enquanto cada linha se comporta como uma entrada em uma tabela de dados. Testar:
  - data[“Alabama”,] #corresponde a uma lista
  - data$Population #corresponde a um vetor

12 – Data Frames >> Importando dados

FIAP

- A linguagem R permite a importação de dados de tabelas em arquivos de textos ou outros formatos, bem como a exportação
- Isso torna possível integrar R a um stream de dados
- Alguns pacotes permitem a importação de arquivos xls e outros formatos
- No curso, as funções de importação de dados a partir de texto usadas serão:
  - read.table() : importa dados de texto na forma de tabela com várias opções de controle do formato do arquivo
  - read.csv() : é um caso particular de read.table(), importando dados do tipo Comma Separated Values
  - read.fwf() : importa dados de tabelas no formato texto com colunas de larguras definidas
  - Exemplo de read.csv
  - Exemplo de read.fwf

12 – Data Frames >> read.csv()

- Importa arquivos tipo Comma Separated Values
- Definição:
  - function (file, header = TRUE, sep = ",", quote = "\"", dec = ".", fill = TRUE, comment.char = "", ...)
    - » header: indica se a primeira linha do arquivo é de cabeçalhos
    - » sep: qual é o caractere de separação de valores
    - » quote: qual é a marca de citação (para valores textuais)
    - » dec: qual é o símbolo de separação das casas decimais
    - » fill: no caso de tabelas com linhas de tamanhos diferentes, indica se as colunas faltantes serão acrescentadas
    - » comment.char: caractere indicando linha de comentário

12 – Data Frames >> read.csv()

- Exemplo: importar arquivo com as informações abaixo

Data,Taxa de câmbio - R$ / US$ - comercial - venda - média - R$ - Banco Central do Brasil, Sistema Gerenciador de Séries Temporais (BCB outras/SGS) - GM366_ERV366,
02/01/1985,1.15781818181818E-09,
03/01/1985,1.15781818181818E-09,
04/01/1985,1.15781818181818E-09,
05/01/1985,
06/01/1985,
07/01/1985,1.17963636363636E-09,
08/01/1985,1.17963636363636E-09,
09/01/1985,1.17963636363636E-09,
10/01/1985,1.17963636363636E-09,
11/01/1985,1.20654545454545E-09,
12/01/1985,
13/01/1985,
14/01/1985,1.20654545454545E-09,
15/01/1985,1.20654545454545E-09,
16/01/1985,1.23381818181818E-09,

12 – Data Frames >> read.fwf()

- Importa arquivos tipo Fixed-Width Formatted data
- Definição:
  - function (file, widths, header = FALSE, sep = "\t", skip = 0, row.names, col.names, n = -1, buffersize = 2000, ...)
    - » widths: vetor com os comprimentos das colunas
    - » header: indica se a primeira linha do arquivo é de cabeçalhos
    - » sep: qual é o caractere de separação de valores
    - » skip: número de linhas a pular no início do arquivo
    - » row.names: indica os nomes das linhas, podendo ser um vetor com os nomes, o número da coluna que contém os nomes, ou o nome da coluna que contém os nomes. NULL força a numeração das linhas
    - » col.names: vetor de nomes para as colunas
    - » n: máximo número de entradas a serem lidas (-1 = sem limite)
    - » buffersize: máximo número de linhas a serem lidas de cada vez

12 – Data Frames >> read.fwf()

- Exemplo: importar arquivo com as informações abaixo
  - Pular 1 linha
  - Tamanhos das colunas: 2,8,2,12,3,12,10,3,4

```
00COTAHIST.1997BOVESPA 19991210

011997010202ACE 4      010ACESITA     PN *        R$
011997010202AVI 4      010ACOS VILL   PN *        R$
011997010202ALB 3      010ALBARUS     ON          R$
011997010202ALP 3      010ALPARGATAS  ON *        R$
011997010202ALP 4      010ALPARGATAS  PN *        R$
011997010202AMS 3      010AMER LEASINGON *ES      R$
011997010202BAS 4      010AMERICA SUL PN *ES      R$
011997010202ANT 3      010ANTARCTICA  ON          R$
```

12 – Data Frames >> Acessando dados

- É possível acessar os dados de um data frame através do acesso a
  - Um único elemento: através do par [<linha>,<coluna>] (nomes ou números):
    - » data[2,3]
    - » data[“Alabama”,”Population”]
  - Uma coluna (vetor de dados): através do símbolo $, [,<col>] ou de [<col>]
    - » data$Murder #retorna um vetor, o mesmo que data[,”Murder]
    - » data[“Murder”] #retorna o subset do data frame
  - Uma linha (lista de dados não homogêneos): através de [<linha>,]
    - » data[“New York”,]
  - Um subconjunto do data frame:
    - » data[c(2,4,6),c(“Murder”, “Population”)]

12 – Data Frames >> Acessando dados

- Algumas funções de visualização:
  - head(data) # Mostra as primeiras linhas
  - tail(data) # Mostra as últimas linhas
  - names(data) # Mostar os nomes de colunas
  - colnames(data) # Mostar os nomes de colunas
  - rownames(data) # Mostar os nomes de linhas
  - dim(data) # Dimensões do dataframe

12 – Data Frames >> Montando dados

- Se conseguimos descontruir uma data frame em suas partes, também conseguimos construir um data frame
- Se tivermos os vetores A, B e C como colunas de dados, podemos criar um data frame através de: data.frame(A,B,C). Exemplo:
  - A<- 1:5
  - B<- seq(10,50,by=10)
  - data <- data.frame(A,B)
- Para renomear as colunas (opcional), atribuir os valores a names(data)
  - names(data)<-c("Un.","Preço")
- Para renomear as linhas (opcional), atribuir os valores a rownames(data)
  - rownames(data)<-c("Um","Dois","Três","Quatro","Cinco")
- Exportação de data frames: para exportar um data frame para um arquivo csv, usar write.csv()
  - write.csv(data, "mydata.csv")

13 – Visualização de dados básica

- A linguagem R possui muitos recursos para visualização gráfica, através da customização dos elementos gráficos
- Sendo uma linguagem voltada para estatística, além dos gráficos convencionas o R fornece um modo fácil de criar representações estatísticas
- Plotagem básica
  - Density Plots (histogramas): `hist()`
  - Dot Plots: `dotchart()`
  - Bar Plots: `barplot()`
  - Line Charts: `lines()`
  - Pie Charts: `pie()`
  - Boxplots: `boxplot()`
  - Scatter Plots: `plot()`
- A maioria das funções de visualização precisam de um vetor numérico como argumento, recebendo outro vetor como rótulos dos dados
  - `plot(data$Income,data$Murder)`
  - `pie(data$Murder,rownames(data))`

14 - Documentação

- Os pacotes da linguagem R fornecem suporte à documentação, que pode ser acessada através da função help() ou do operador ?
  - ?<função>
  - help(<função>)
- Para ver a documentação de operadores, use aspas em torno do mesmo
  - ?“+"
- Para fazer uma busca de termos na documentação disponível, fazer
  - ??“<key words>"
  - help.search(“<key words>")
- Para executar o exemplo incluso na documentação, usar:
  - example(<função>) #por exmplo: example(plot)

15 - Lista de exercícios para fixação

- Baixar o pacote de exercícios de:
  - https://dl.dropboxusercontent.com/u/22050262/Lista1-LinguagemR.zip
- Importar ResultadosRTLS.csv
- Plotar as colunas LM-FT,MCMC-LM-FT (gráfico de linhas)
- Mostrar os índices em que a coluna LM é maior do que MCMC-LM
- Mostrar as médias das coluna LM e MCMC-LM
- Importar COTAHIST.A1997 no formato FWF. Usar as informações do arquivo SeriesHistoricas_Layout.pdf para definir o formato de importação. Limite o número de entradas lidas para 20000.
- Fazer o gráfico de dispersão (scatter plot) entre os valores de fechamento de duas ações à sua escolha.
  - Valor de fechamento: coluna PREULT
  - Para cada valor de DATA, o valor de uma ação é definida pelo seu CODNEG junto com um valor de CODBDI (usar CODBDI=2)
  - Para fazer o scatter plot entre as ações, é necessário determinar as datas em que ambas aparecem, e plotar os valores apenas dessas datas

Referências

- Norman Matloff. *The Art Of R Programming*. No Starch Press, São Francisco, CA, 2011.
- Joseph Adler. *R in a Nutshell*. O'Reilly Media, Inc., Sebastopol, CA, 2012
- Mark Gardener. Beginning R: The Statistical Programming Language. John Wiley & Sons, Indiana, IN, 2012.

FIAP

A MELHOR FACULDADE DE TECNOLOGIA