FIAP

MBA+

ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

FIAP

MBA+

PROGRAMANDO IA COM R

Prof. Elthon Manhas de Freitas
elthon@usp.br

2018

Sobre o R

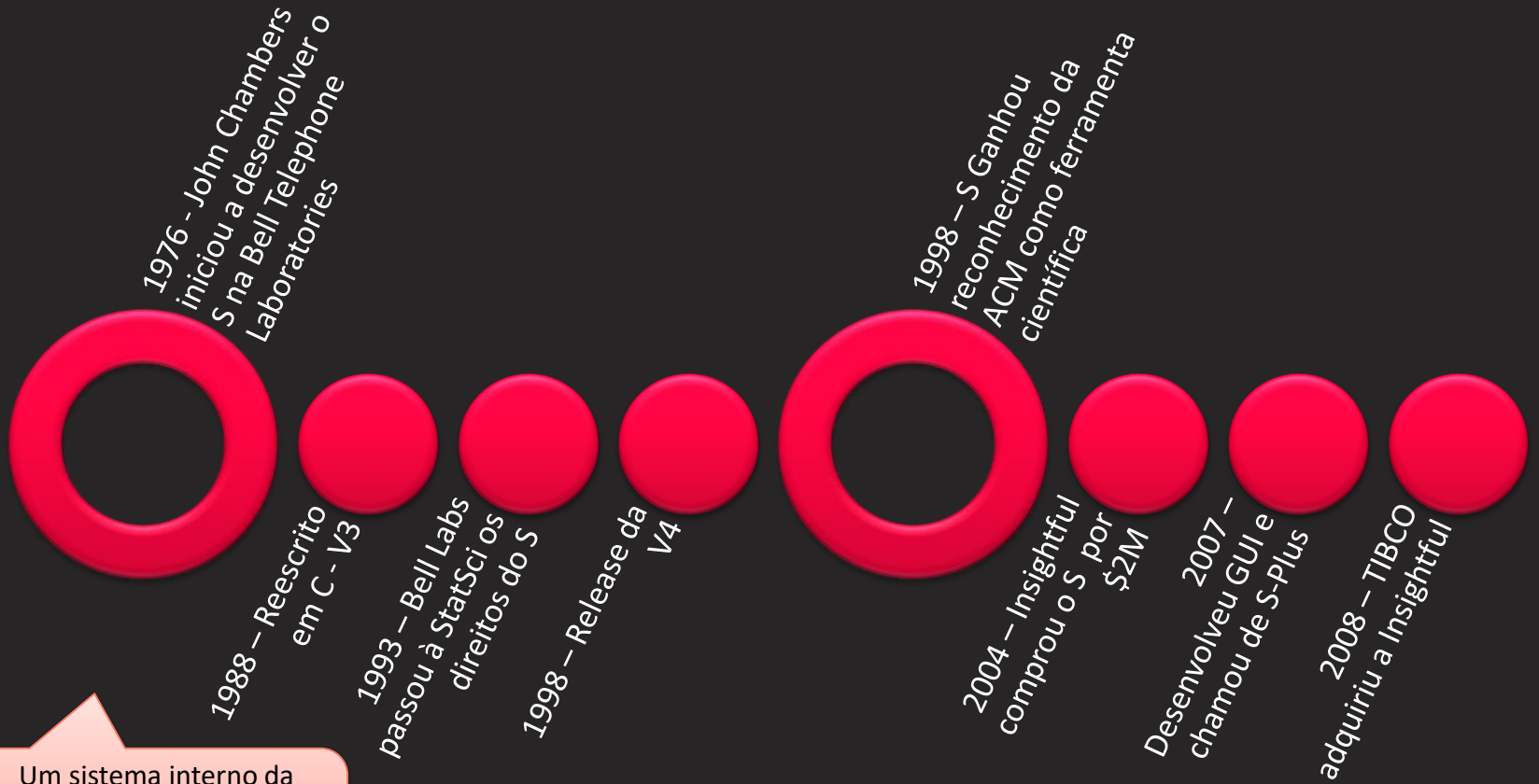
Conhecendo o R

O que é o R?

Uma variação do S!

WOW!

Histórico do S



Um sistema interno da AT&T para análises estatísticas.
Escrito em Fortran

JOHN M. CHAMBERS

https://statweb.stanford.edu/~jmc4/

Histórico do R

- Criado em 1991 por Ross Ihaka e Robert Gentleman (publicado em 1993) no departamento de estatísticas da universidade de Auckland.
- O S só estava disponível em pacotes comerciais.
- Em 1995 foi transformado em software livre sob a General Public License GNU
- Em 1996 foi criada uma lista de discussão com a comunidade.
- Em 1997 foi criado o Core Group com várias pessoas associadas ao S

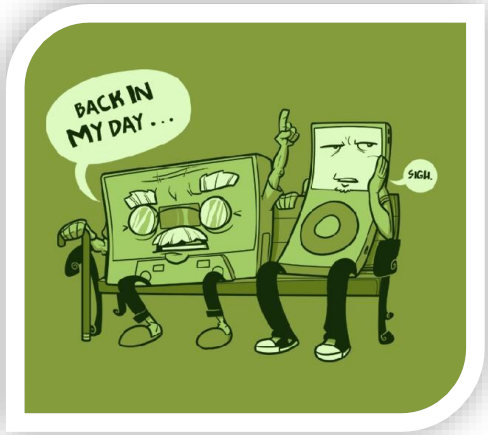
Principais características

- Software livre e Open Source
  - GNU General Public License
- Distribuição em Linux, Windows, Mac
  - Pode ser compilado para outras plataformas

- Linguagem interpretada, e não compilada
  - Ambiente exploratório sem compilação
- Tipagem Dinâmica
- Integração com bibliotecas de alto desempenho, compiladas em C, C++ e Fortran

Limitações importantes sobre o R

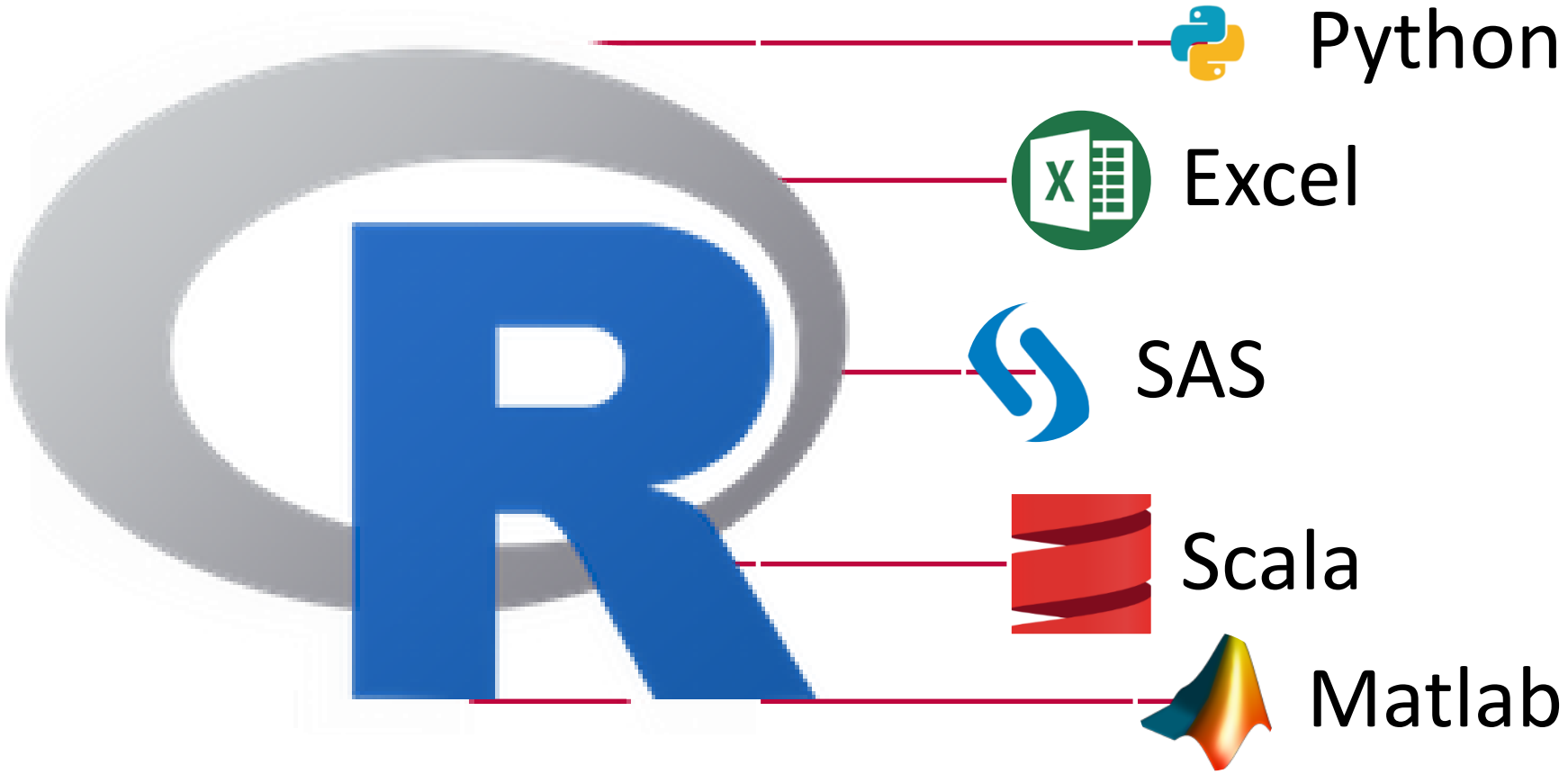
- Não há garantia de funcionamento e continuidade
  - Não há um contrato com uma empresa, como quando você compra uma licença de software
- Uma linguagem de mais de 40 anos
- Todos os dados ficam em memória



- Laços muito lentos (loop)
  - Velocidade de processamento muito lenta, se comparada a linguagens compiladas

ENTRE AS PRINCIPAIS FERRAMENTAS DOS CIENTISTAS DE DADOS

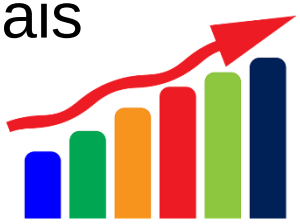
Principais ferramentas / alternativas ao R



O que é possível fazer com R

FIAP

- Analytics
  - Matemática e estatística do básico ao mais avançado
  - Modelagem estatística, simulações de ambientes e otimização
  - Aprendizado de Máquina – Machine Learning
  - Integração e análise com Big Data
- Gráficos e Visualização
  - Plotagens gráficas, simples a profissionais
  - Gráficos dinâmicos
  - Reprodução geográfica



E mais …

O que é possível fazer com R

FIAP

- Programação

  - Criação de scripts complexos
  - Documentação de processos
    - (já tentou reproduzir algo no excel?)

- Distribuição
  - Possibilidade de criar e compartilhar pacotes
  - Documentação
  - Aplicações rodando em Back-End
  - Front-end interativo

Curiosidade sobre o R

- O que é esperado quando se escreve um código em R?
  - Que o código não erre: `R <- !R`

The Comprehensive R Archive Network

FIAP

CRAN – Manuais oficiais

http://cran.r-project.org

- Uma introdução ao R
  http://cran.r-project.org/doc/manuals/r-release/R-intro.html
- R Data Import/Export
  http://cran.r-project.org/doc/manuals/r-release/R-data.html
- Writing R Extensions: Como escrever e organizar pacotes R
  http://cran.r-project.org/doc/manuals/r-release/R-exts.html
- R Installation and Administration: Como compilar o R a partir do Código fonte
  http://cran.r-project.org/doc/manuals/r-release/R-admin.html
- R Internals: Manuais da estrutura de baixo nível para desenvolvedores R e membros do “R Core”
  http://cran.r-project.org/doc/manuals/r-release/R-ints.html
- Definição da linguagem R: Documentos para desenvolvedores
  http://cran.r-project.org/doc/manuals/r-release/R-lang.html

Instalar o R

FIAP

- Página para Download:

  https://cran.r-project.org/

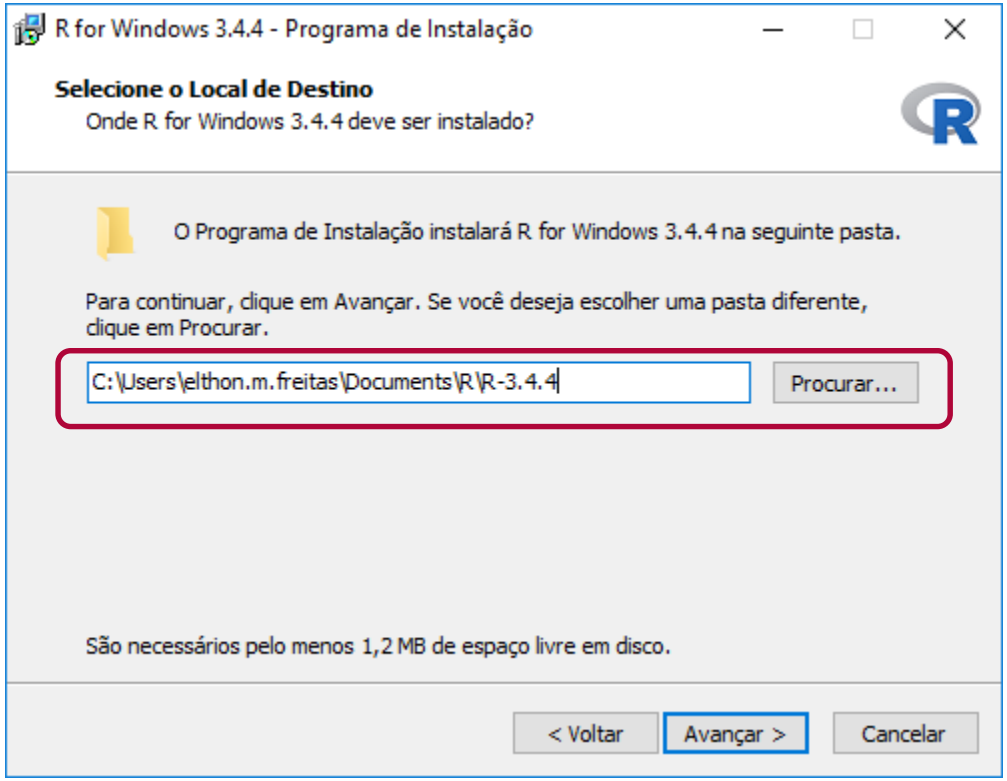
  - Ou pelo R-Project:
  - www.r-project.org

  - Alternativo:
  - https://cran.r-project.org/mirrors.html

  - Microsoft R:
  - https://mran.microsoft.com/download

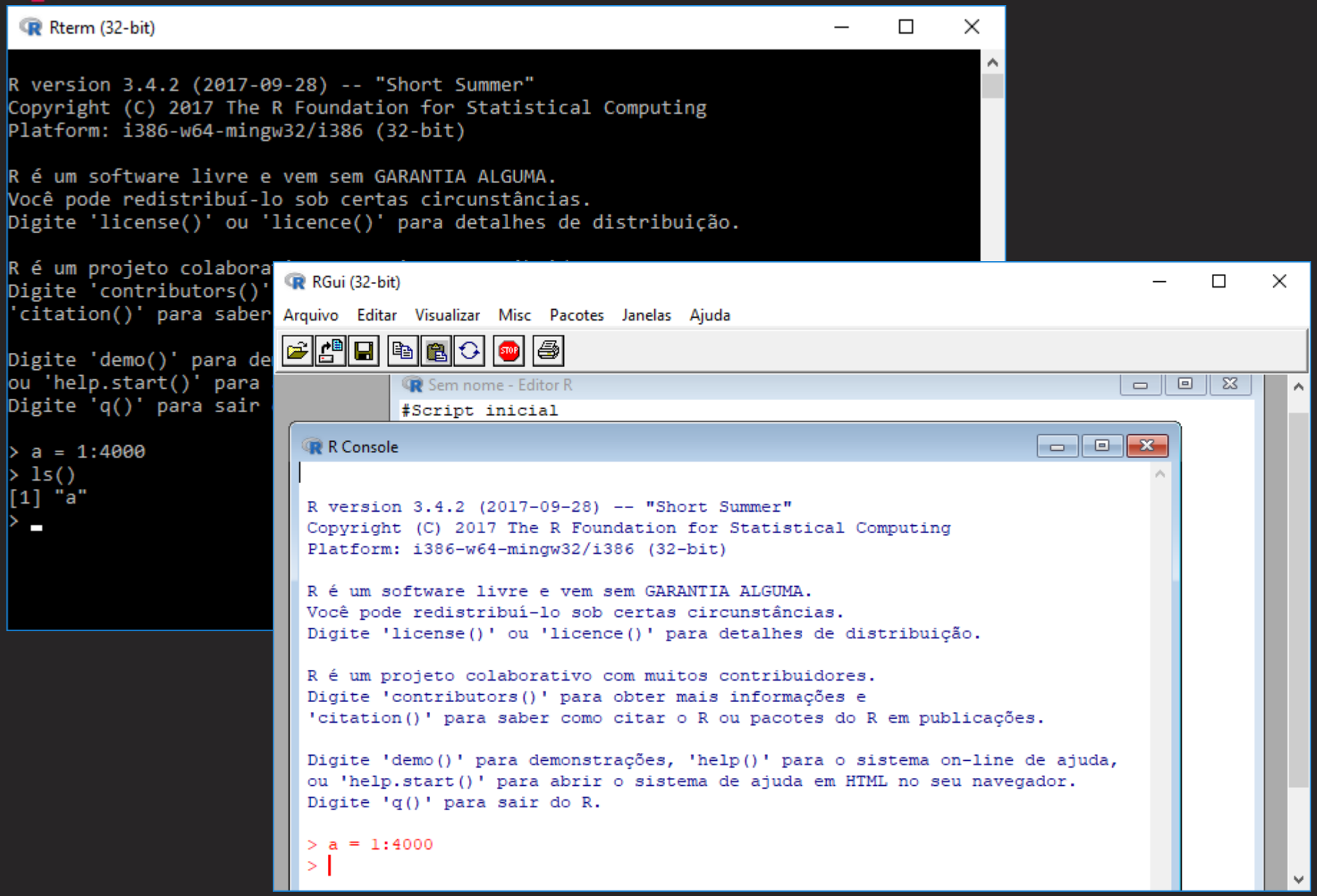
Instalando o R

- Seguir o assistente
  - Avançar + ... + Avançar + Concluir



Obs.: As máquinas do laboratório já possuem o R instalado.

Interfaces clássicas do R

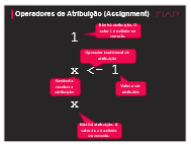


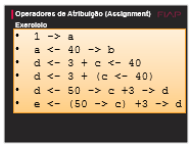
Olá mundo! (Hello World)

- Localizar instalação do R
  - Pasta “C:\Program Files\R\R-3.4.4\”
    - Sub-pasta “bin\i386” ou “bin\x64” (32 ou 64 bits)

- Programas a executar (cada um deles)
  - R.exe *(executar, olhar e sair >q();)*
  - Rgui.exe *(usar este para as próximas práticas)*

- R-Markdown o professor (Basic Building Blocks)
  - Conteúdo : A01.01-Basic BB





Instalar o RStudio e o RBuildTools

- Página para Download

  https://www.rstudio.com/products/rstudio/download/

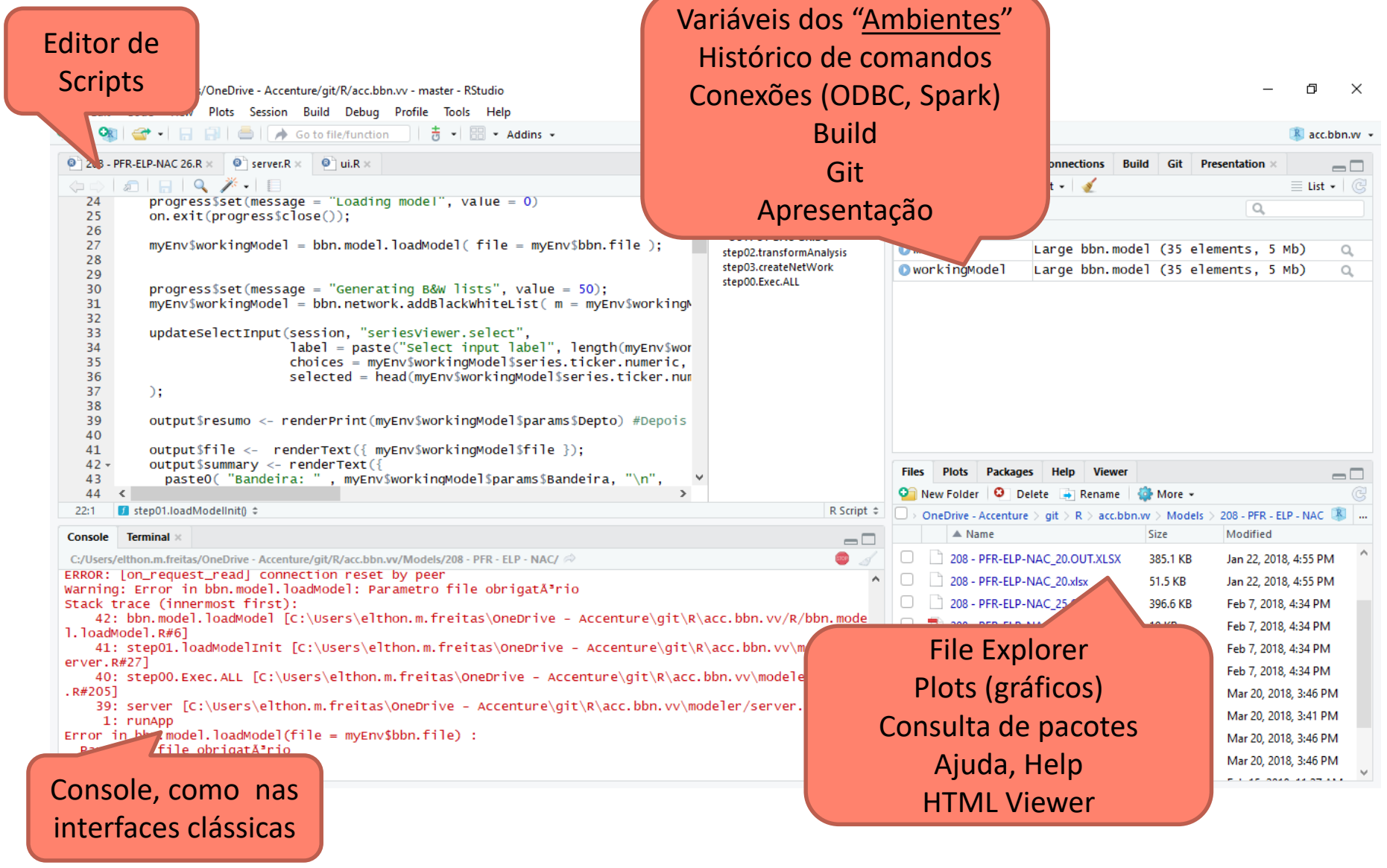
- R Build Tools

  O Build Tools é instalado e configurado a partir do RStudio, ao se criar um projeto ou pelo link:

  https://cran.r-project.org/bin/windows/Rtools/Rtools33.exe

Obs.: As máquinas do laboratório já possuem o Rstudio instalado. **É necessário instalar o RBuildTools.**

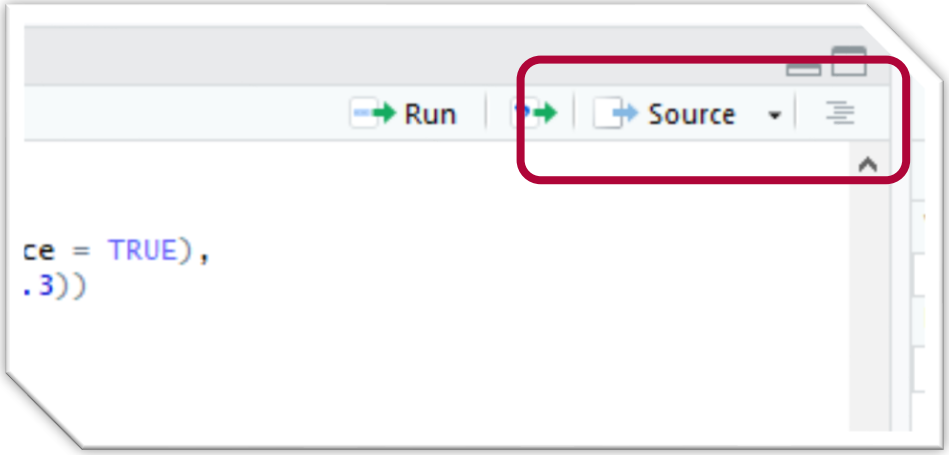
RStudio – Sobre o ambiente



RStudio – Criar um Script

- Salvar um arquivo com a extensão “.R”

- Abrir e clicar no botão “Source”



- Linha de código:

```
> source( “nome do script.R”)
```

Diretório de trabalho (WD-Working Dir)

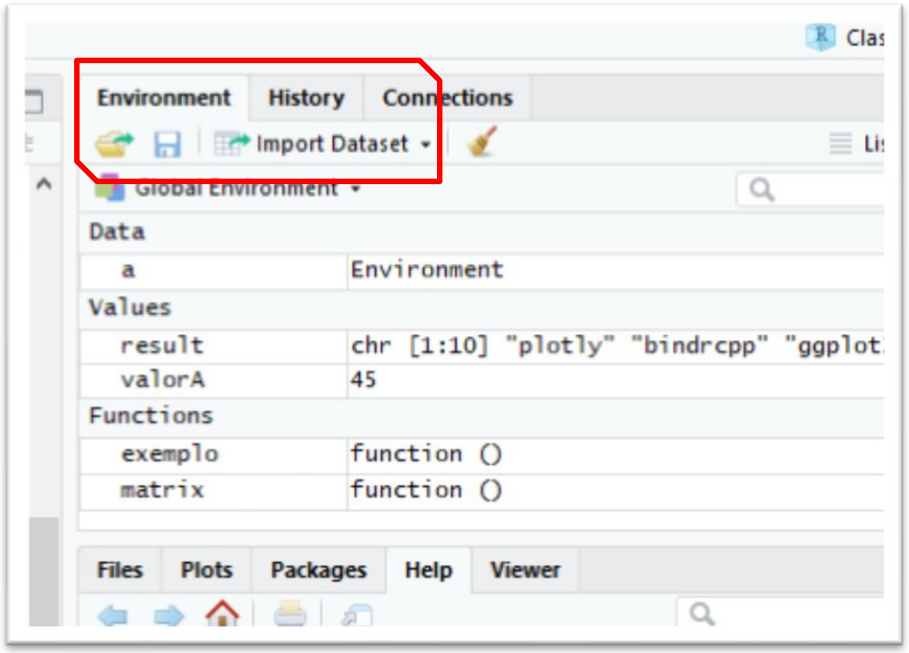
FIAP

- É o diretório em que a sua sessão irá salvar e abrir arquivos por padrão.
- Ao abrir ou salvar arquivos do WD, não é necessário informar o caminho inteiro.

- Consultar o WD atual:
  - `getwd()`
- Alterar o WD para outro local:
  - `setwd(dir = "caminho")`

RStudio – Salvar memória de trabalho

- Utilizar os ícones presents na aba Environment.
- Alternativas por linha de código:




```
> save.image("arquivo")
```

Salva todas as variáveis (Workspace)

```
> save(variáveis, "arquivo")
```

Salva variáveis específicas

```
> load("arquivo")
```

Carrega as variáveis presentes no workspace

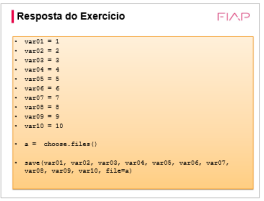
A extensão padrão para arquivos de dados é ".RData"

Exercício

- Criar script que:
  - Crie 10 variáveis em memória
  - Salve as 10 variáveis em arquivo a ser escolhido.

  - Para escolher o arquivo de destino, use a função `choose.files( )`

Resposta:



Pacotes

Bibliotecas , Libraries

Principais pacotes - Core Team (RStudio)

Ciclo de vida de um pacote

Instalação do pacote

Carregamento em memória

Descarregar da memória

Remoção do pacote

- install.packages("abc.data")
- library("abc.data")
- require("abc.data")

- detach("package:abc.data", unload=TRUE)
- remove.packages("abc.data")

Gerenciamento de pacotes pela UI

1

install.packages("abc.data")

2

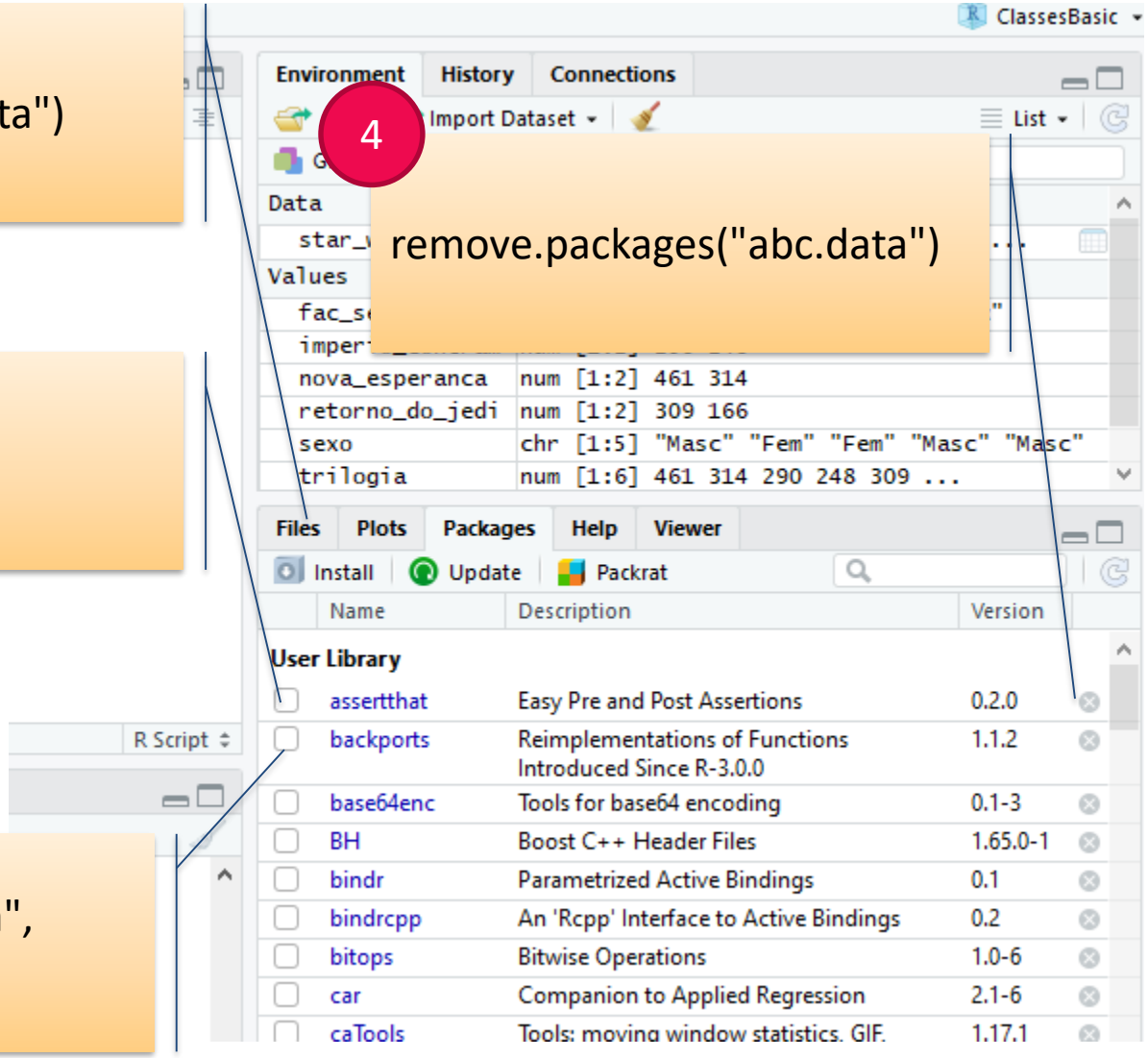
library("abc.data")
require("abc.data")

3

detach("package:abc.data", unload=TRUE)

4

remove.packages("abc.data")



Exercício

FIAP

- Instalar os pacotes:

stringr	tidyr	shiny
data.table	readxl	swirl
dplyr	openxlsx	xlsx
ggplot2	sqldf	rjson
h2o	dt	devtools
knitr	zoo	curl
plotly	lubridate	visNetwork
plyr	rmarkdown	

A maioria destes pacotes serão usados ao longo do curso

Usar um pacote sem carregá-lo

FIAP

- É possível acessar um componente de um pacote sem carregá-lo na memória.

- Para isso:

```
<nome_do_pacote>::<componente>
```

- Exemplo:

```
ggplot2::qplot(1)
```

Tipos de dados

Tipos de dados primitivos

FIAP

Abrangência

- logical : Números booleanos
  - `var_logical <- TRUE` (TRUE, FALSE, T, F)
- integer : Números inteiros
  - `var_integer <- 45L`
- numeric : Números decimais
  - `var_numeric <- 45.6`
- character : Textos, strings
  - `var_character <- "Aula inicial"`
    - `(funciona com aspas simples ou duplas)`

- Verificar o tipo de uma variável
  - `class(var_logical)`

Tipos de dados primitivos

- complex : Números imaginários

```
• a <- 1.3 + 2.45i
```

Componente real

Componente imaginário

- Inventados por Gauss. Usado para solucionar o problema das equações que não possuem raizes no conjunto dos números reais $\mathbb{R}$.
- Ele convencionou que a Raiz Quadrada de -1 é i.

i = raiz de -1

$i^2 = -1$

$i^3 = -i$

$i^4 = +1 = (i^2)^2$

```
• a <- sqrt( as.complex(-1) )
• a
• a ** 2
• a ** 3
• a ** 4
```

Verificação e conversão dos dados

- Para verificar se uma variável é de um determinado tipo, é possível:
  - Verificar a class

```
• class(x)
```

  - Usar as funções “is”

```
• is.integer(x)
• is.numeric(x)
• is.character(x)
• ... etc
```

- Para converter existem as funções “as”:

```
• as.integer(x)
• as.numeric(x)
• as.character(x)
• ... etc
```

Estrutura de dados : Vetores

FIAP

- Para criar um vetor, utilizamos a função “c”

  c significa “combine”

```
• vet_int <- c(1, 2, 3, 10)

• vet_bool <- c(TRUE, TRUE, TRUE, FALSE)

• vet_str <- c(“F”, “I”, “A”, “P”)

• vet_num <- c( 12.1, 14, 78.5, FALSE)
```

Estrutura de dados : Vetores

- É possível combinar em um vetor, os componentes de outros vetores

```
• vet_int <- c(vet_int, -1, vet_int)

• vet_bool <- c(FALSE, vet_bool)

• vet_str <- c(10, vet_str, vet_int )
```

Estrutura de dados : Vetores

- Prática com o professor
  - Conteúdo do Markdown: A01.02-Vetores

Estrutura de dados : Vetores

FIAP

- Resumo:

  - Criar vetores
  - Combinar elementos dos vetores
  - Nomear elementos do vetor
  - Operações com 2 vetores
  - Funções especiais
    - sort, sum, min, max, mean, median, quantile

Estrutura de dados : Matrizes

FIAP

- Para criar uma matriz, utilizamos a função “matrix”

```
matrix(c(1,2,3,4,5,6,7,8,9), nrow = 3)
```

- Prática com R-Markdown
  - Conteúdo do : A01.03-Matrizes

Estrutura de dados : Array

FIAP

- Arrays são estruturas homogêneas de várias dimensões.
- Podemos dizer que vetores são arrays de 1 dimensão e matrizes são arrays de 2 dimensões.

```
array(1:18, dim = c(3,3,2))
```

Estrutura de dados : Fatores

FIAP

- Servem para categorizer dados
- Utilizados em dados que se repetem com freqüencia.
- Exemplo sem ordem:

```
sexo <- c("Masc","Fem","Fem","Masc","Masc")
fac_sexo <- factor(sexo)
fac_sexo
```

- Exemplo ordenado:

```
temperatura <- c("alta", "baixa",
"alta","baixa", "media")
temperaturaf <- factor(temperatura, order =
TRUE, levels = c("baixa", "media", "alta"))
temperaturaf
```

Estrutura de dados : Data Frame

FIAP

- A estrutura de dados mais usada em análise de dados.

- Similar às matrizes, mas com uma série de funções específicas, dentre elas:
  - Buscas, indexação
  - Cada coluna pode ter um tipo diferente de dados.

```
• mtcars
• class(mtcars)
```

Estrutura de dados : Listas

FIAP

- Muito utilizada para guardar dados estruturados.

- Similar aos vetores, com as seguintes particularidades:
  - Cada elemento pode ter um tipo diferente de dados.

```
• vetA <- 1:10
• matA <- matrix(1:9, ncol = 3)
• dfA <- mtcars[1:10,]

• minha_lista <- list(vetA, matA, dfA)
```

Exercício

- Criar 5 vetores com exatamente 4 elementos para cada tipo de dado:
  - Lógico, Inteiro, Numérico, Texto, Imaginário

- Criar uma lista com 5 elementos. Cada elemento deve ser um vetor de 1 tipo de dado
- Criar uma matriz 4 x 4 com 16 números inteiros pares

- Criar um data.frame em que cada coluna é um dos vetores criados neste exercício.
  - Nomear as colunas (a seu critério)
  - Nomear as linhas com “L1, L2, L3 e L4”

Operadores lógicos

- Operadores de comparação (relacionais)
  - == Igual
  - != Diferente
  - > Maior
  - >= Maior ou igual
  - < Menor
  - <= Menor ou igual

- Operadores lógicos
  - ! - Não (nega a sentença)
  - && - E
  - || - OU

MBA+