Aula 1 - Introdução ao R

Vitor Rios

10 de outubro de 2017

Bem vindos

Cronograma

Dia 1 - 21/11 - Introdução, histórico e como usar o R e Rstudio, boas práticas de programação
Dia 2 - 22/11 - Criando e e manipulando dados, noções de lógica de programação I
Dia 3 - 23/11 - Leitura de arquivos, limpeza e manipuação de dados, noções de lógica de programação II
Dia 4 - 28/12 - Análise exploratória e funções matemáticas
Dia 5 - 05/12 - Análise estatística básica no R Dia 6 - 06/12 - Gráficos
Dia 7 - 07/12 -
Dia 8 - 12/12 -
Dia 9 - 13/12 - Programação para iniciantes: da idéia à análise final
Dia 10 - 14/12 - Versionamento de código: noções de git

Aulas: teoria (1 a 2 h) + prática (wiki do curso)

O que é o R?

- R é uma linguagem de programação voltada para análises estatísticas e manipulação de dados
- RStudio é um ambiente de desenvolvimento integrado (IDE) que extende as capacidades do R e facilita o uso

Jeito R de ser

- Praticidade
- Reprodutibilidade
- Simplicidade (sim, você leu direito)

Ao fazer o tratamento e análise dos dados dentro do R, você não altera os dados originais, e pode reproduzir tudo, bastando ter o script e os dados originais. Não é necessário nem salvar os gráficos.

Porque Usar R?

- Uma vastidão de pacotes de análises prontas

CRAN Task Views

Bayesian	Bayesian Inference
ChemPhys	Chemometrics and Computational Physics
ClinicalTrials	Clinical Trial Design, Monitoring, and Analysis
Cluster	Cluster Analysis & Finite Mixture Models
DifferentialEquations	Differential Equations
Distributions	Probability Distributions
Econometrics	Econometrics
Environmetrics	Analysis of Ecological and Environmental Data
ExperimentalDesign	Design of Experiments (DoE) & Analysis of Experimental Data
ExtremeValue	Extreme Value Analysis
Finance	Empirical Finance
FunctionalData	Functional Data Analysis
Genetics	Statistical Genetics
Graphics	Graphic Displays & Dynamic Graphics & Graphic Devices & Visualization
HighPerformanceComputing	High-Performance and Parallel Computing with R
MachineLearning	Machine Learning & Statistical Learning
MedicalImaging	Medical Image Analysis
MetaAnalysis	Meta-Analysis
Multivariate	Multivariate Statistics
NaturalLanguageProcessing	Natural Language Processing
NumericalMathematics	Numerical Mathematics
OfficialStatistics	Official Statistics & Survey Methodology
Optimization	Optimization and Mathematical Programming
Pharmacokinetics	Analysis of Pharmacokinetic Data
Phylogenetics	Phylogenetics, Especially Comparative Methods
Psychometrics	Psychometric Models and Methods
ReproducibleResearch	Reproducible Research
Robust	Robust Statistical Methods
SocialSciences	Statistics for the Social Sciences
Spatial	Analysis of Spatial Data
SpatioTemporal	Handling and Analyzing Spatio-Temporal Data
Survival	Survival Analysis
TimeSeries	Time Series Analysis
WebTechnologies	Web Technologies and Services
gR	gRaphical Models in R

Figure 1: CranTaskView

- **Interfaces amigáveis que não precisam de programação**
  - RCommander <- fácil acesso às análises principais
  - Action <- análises integradas ao Excel
  - RStudio <-ambiente completo de programação
- Possibilidade de programar suas próprias análises, de acordo com suas necessidades
- Integração com outros softwares
  - GIS
  - Git <– **Salvação da humanidade**
  - Excel
  - C++
  - Python
  - Esta aula foi feita em R no RStudio

Primeiros passos no R

Abrir o RStudio

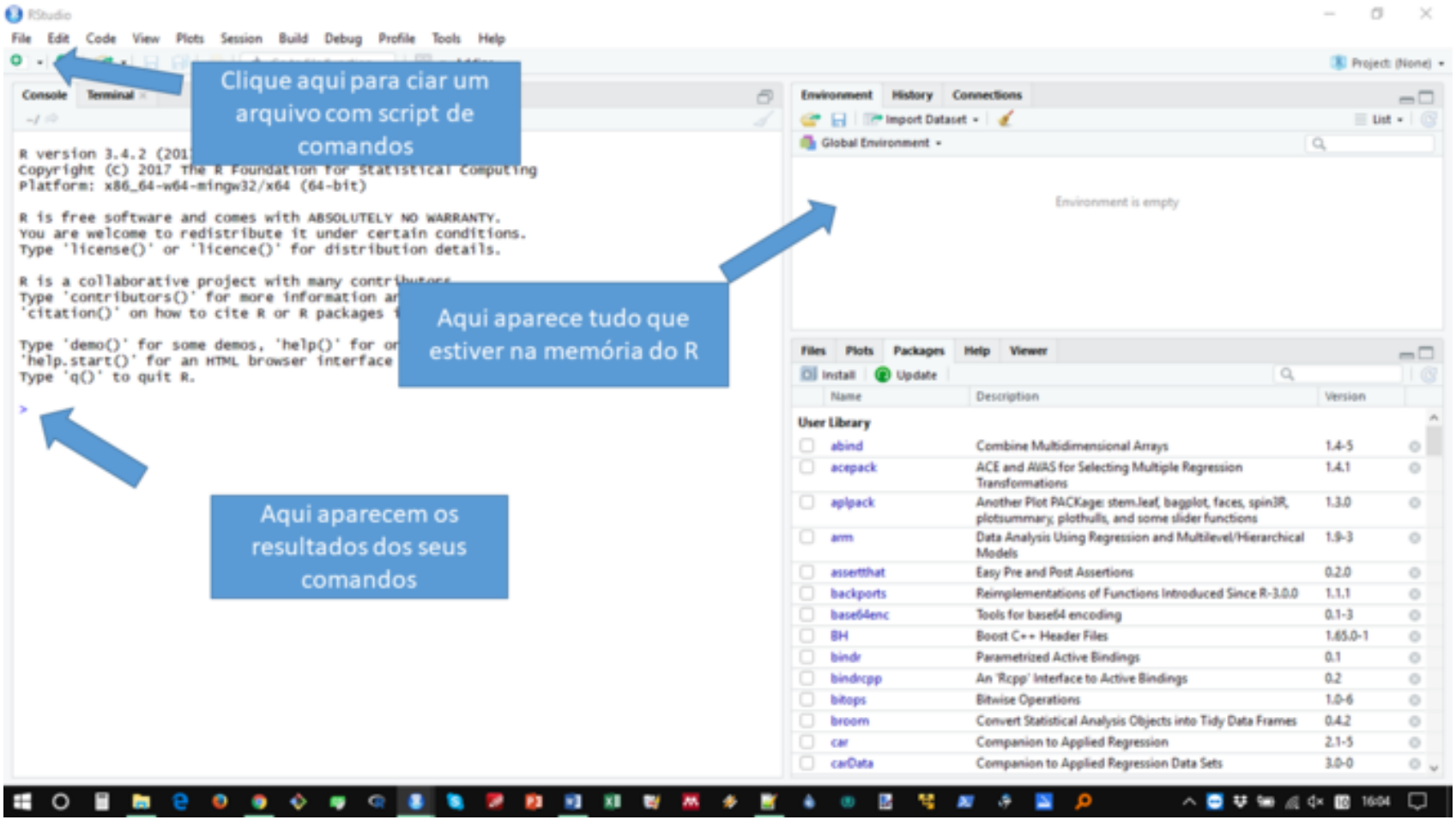


Figure 2:

No RStudio

Digite o comando abaixo na aba console, onde está o caractere “>” , e depois tecle **enter**

```
4 + 12
```

```
## [1] 16
```

O resultado aparece logo abaixo, na própria aba, em cor diferente

O R é uma calculadora completa, tente várias combinações.

Qualquer texto após # é ignorado pelo R

```
1 + 1  # soma
```

```
## [1] 2
```

```
3 - 74 ## subtração
```

```
## [1] -71
```

```
4 * 12 ## multiplicação
```

```
## [1] 48
```

```
3 / 7 ## divisão
```

```
## [1] 0.4285714
```

O R também aceita comandos mais complicados

```
pi ^ 2 # potência
```

```
## [1] 9.869604
```

```
((4 * 7) + 45) / 2
```

```
## [1] 36.5
```

O que está dentro dos parênteses é avaliado de dentro para fora, e o resultado é usado para a operação seguinte, e assim por diante

Logaritmos, exponenciações, raízes

```
log(10, base=2)
```

```
## [1] 3.321928
```

```
exp(1)
```

```
## [1] 2.718282
```

```
sqrt(2)
```

```
## [1] 1.414214
```

Funções

Quando temos um nome seguido de parentese com alguma coisa dentro, temos uma **função** Função é um conjunto de comandos que faz uma operação em um objeto
sqrt(x) calcula a raiz quadrada de x
Funções podem ter opçoes, chamadas argumentos

```
log(10, base=2)
```

calcula o logaritmo de 10 na base 2

Quase tudo que vocêe vai usar no R são funções

Muito cuidado com a digitação: para o R, maiúsculas e minúsculas são diferentes, e um espaço no meio da palvra atralha

O R é burro, qualquer erro de digitação faz com que ele não encontre o que você procura

Ajuda das funções

você pode pedir socorro ao R em qualquer momento

```
?log()
?exp()
?plot()
help(library)
```

As funções geralmente vem em conjuntos especializados voltados para aulguma análise, chamados de *pacotes*

Outros lugares para pedir ajuda

Quase todos os problemas que você tem com R, alguém já teve e resolveu

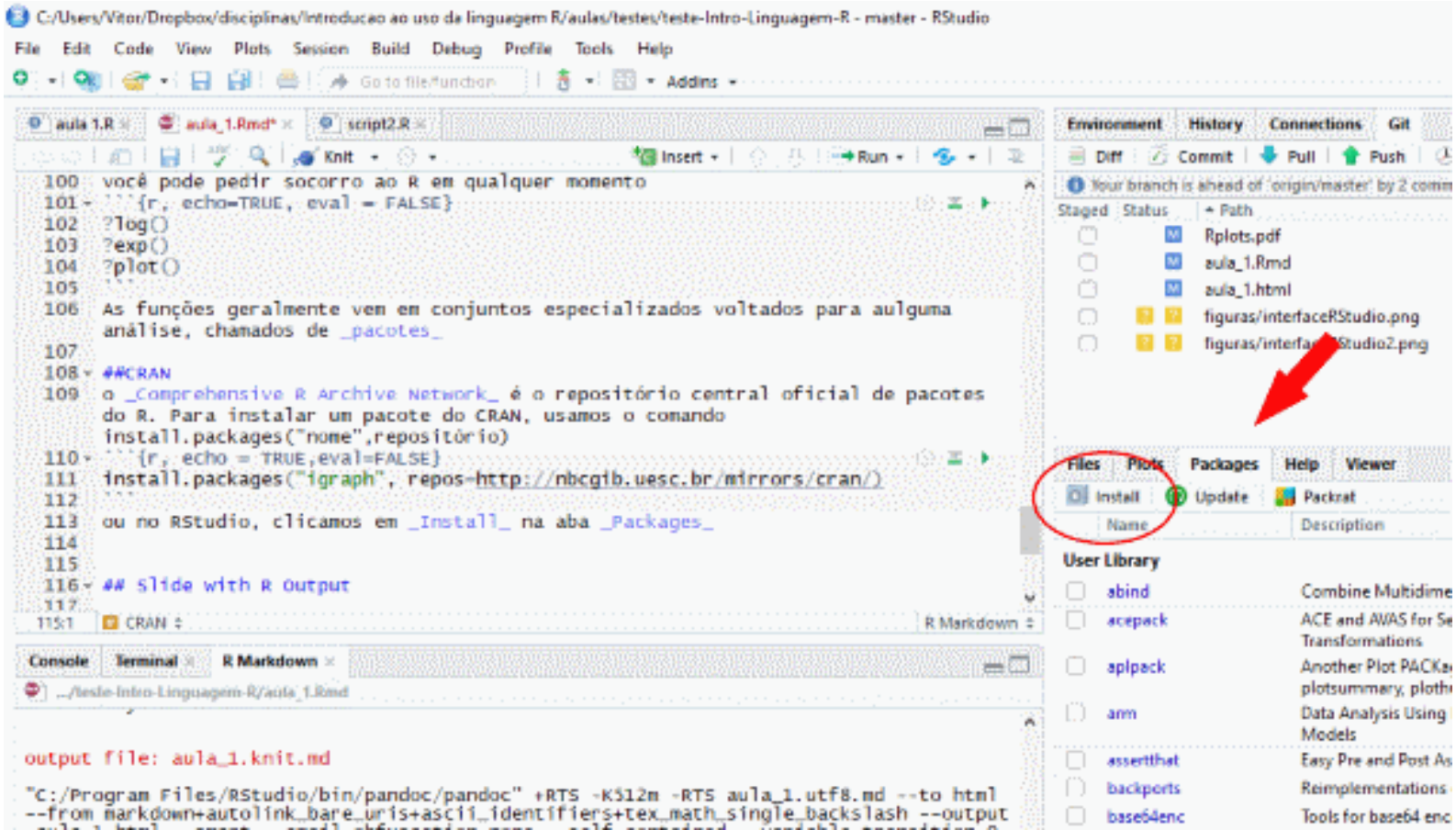
Google <- busque por "R manova" ou "R phylogenetics" ou "R cluster analysis", etc. Stack Overflow <- geralmente é o primeiro resultado do Google

CRAN

o *Comprehensive R Archive Network* é o repositório central oficial de pacotes do R. Para instalar um pacote do CRAN, usamos o comando install.packages("nome",repositório)

```
install.packages("igraph", repos=http://nbcgib.uesc.br/mir
```

ou no RStudio, clicamos em *Install* na aba *Packages*



Para usar um pacote

```
library(igraph)
```

```
##
## Attaching package: 'igraph'
```

```
## The following objects are masked from 'package:stats':
##
##     decompose, spectrum
```

```
## The following object is masked from 'package:base':
##
##     union
```

agora as funções do *igraph* estão disponíveis para serem usadas

Dados e variáveis

R é uma linguagem de manipulação de objetos

No R, podemos criar caixas para guardar coisas que iremos usar depois, que chamamos de *variáveis* Para isso usamos o sinal $=$ ou $<$- .
$<$- existe por motivos de compatibilidade para teclados que nao possuíam $=$ (lá dos tempos jurássicos da computação). Use $=$

```
pote.de.sorvete = "feijão"
```

Na linha acima o R interpreta o que está dentro das apas como texto, e guarda esse valor dentro de um pedaço da memória que recebe o nome "pote.de.sorvete".

O nome das variáveis não deve conter espaços, acentos,ou caracteres especiais como ç, ~, !, ?, /, |

Ao digitarmos *pote.de.sorvete* no console, o R nos mostra o que está guardado nesse pedaço de memória

```
pote.de.sorvete
```

```
## [1] "feijão"
```

Escolha nomes que façam sentido para suas variáveis

Lembre que você vai ter de ler seus códigos depois, e advinhar o que você quis dizer com *a*, *aa*, *x*, *X*, *a7589*, *MyVar* e outras bizarrices que pareciam fazer sentido.

Aproveite que o RStudio tem autocompletar!

Para trocar o que está dentro dessa variável, basta usar o $=$ novamente

```
pote.de.sorvete = "sorvete"
pote.de.sorvete
```

```
## [1] "sorvete"
```

porque o código baixo nao funciona?

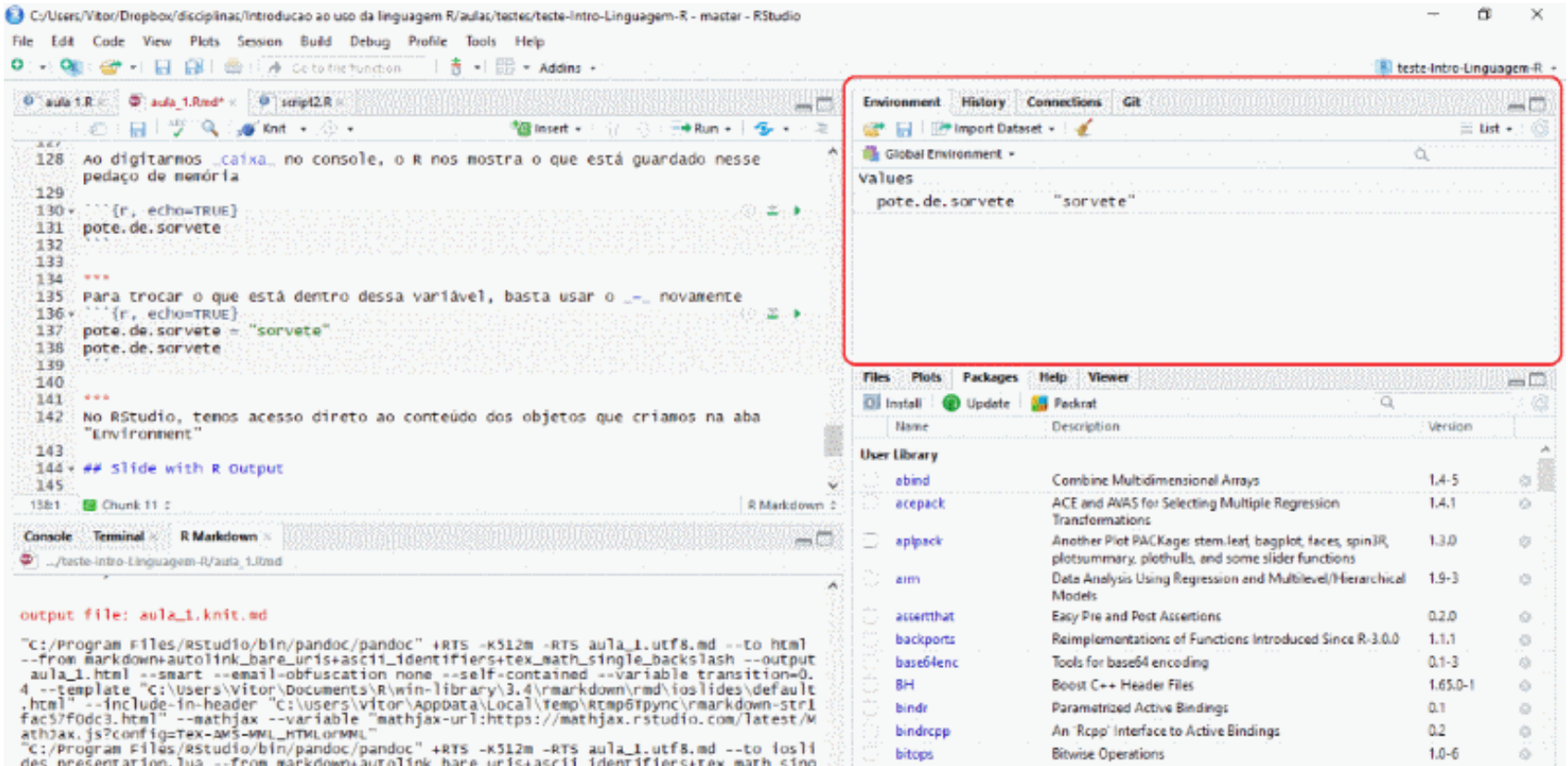
```
pote .de.sorvete = "chuchu"
```

Para ver o que tem na memória do R, use o comando *ls()*

```
ls()
```

```
## [1] "pote.de.sorvete"
```

No RStudio, podemos ver o conteúdo dos objetos que criamos na aba "Environment"



Variáveis também podem guardar conjuntos de dados

Chamamos estas variáveis de *vector* ou *vetor*. Criamos elas com a função *concatenate*, *c()*

```
geladeira = c("sorvete","cebola","agua", "miojo","macarrão"
geladeira
```

```
## [1] "sorvete"          "cebola"           "agua"
## [4] "miojo"            "macarrão"         "pacote.de.l
```

Vetores só guardam objetos iguais: ou texto, ou números, nunca os dois juntos
Veja o que as funções *seq()* e *rep()* fazem!

Vetor é a unidade básica do R

Quase todas as funções do R são baseadas em manipulação de vetores

```
peso.real = c(100, 45, 77, 60)
peso.no.perfil = c(70, 50, 60, 55)
```

Ao somar, multiplicar, subtrair ou dividir vetores, o R pega os valores na mesma posição e faz a operação em questão

regra da reciclagem Quando um vetor é menor que o outro, o R repete o vetor menor até completar o maior

```
peso.real = c(100, 45, 77, 60, 50, 77.5)
ganho.apos.almoco = c(2, 5.3)
peso.gordo= peso.real + ganho.apos.almoco
peso.gordo
```

```
## [1] 102.0  50.3  79.0  65.3  52.0  82.8
```

Sempre confira o tamanho dos seus vetores!

Tamanho e conteúdo dos vetores

Função *length()* retorna o tamanho do vetor

```
peso.real = c(100, 45, 77, 60, 50, 77.5)
length(peso.real)
```

```
## [1] 6
```

Para acessar o elemento *i* do vetor, use *nome[i]*

```
peso.real = c(100, 45, 77, 60, 50, 77.5)
peso.real[3]
```

```
## [1] 77
```

Dataframes

Dataframes são o formato de tabela mais comum do R. No dataframe você pode ter colunas com tipos de dados diferentes, como numérico e texto.
Para criar o dataframe,, temos 3 jeitos principais

Salvando ~~sua vida~~ seu código

Se você fechar o R, vai perder tudo que você fez

Solução: salve seu código em um script!

Script é um arquivo de texto sem formatação, com a extensão *.R* ou *.r*, ao invés de .txt. Pense no script como a receita, o R como a cozinha, e a análise o jantar.

Você pode editar no bloco de notas, Notepad++, Gedit,Vim, EMACS, etc, mas evite usar Word, ele insere formatações que atrapalham tudo

Use o RStudio!

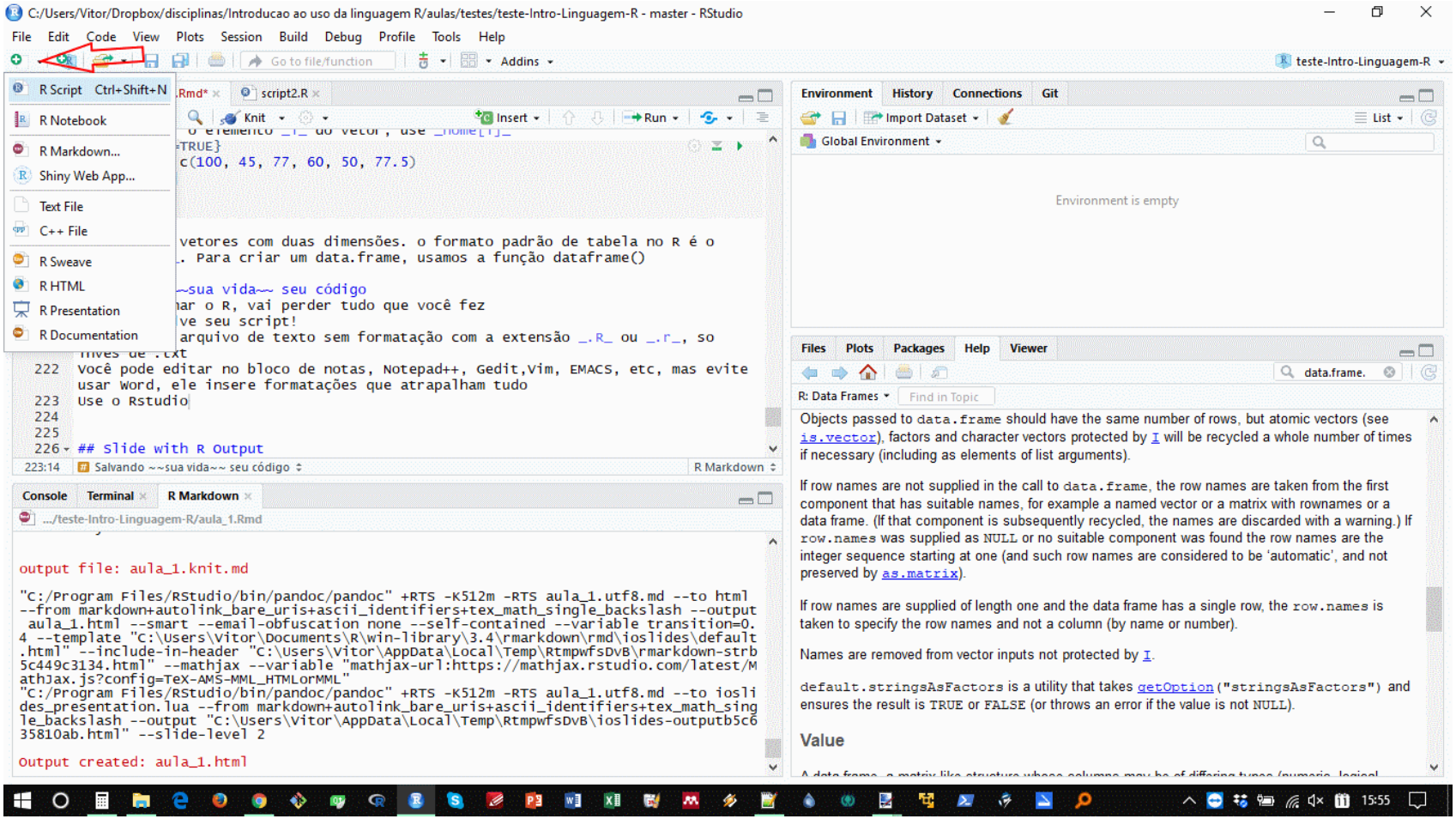

C:/Users/Vitor/Dropbox/disciplinas/Introducao ao uso da linguagem R/aulas/testes/teste-Intro-Linguagem-R - master - RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
teste-Intro-Linguagem-R
R Script Ctrl+Shift+N
R Notebook
R Markdown...
Shiny Web App...
Text File
C++ File
R Sweave
R HTML
R Presentation
R Documentation
script2.R
Knit
Insert
Run
c(100, 45, 77, 60, 50, 77.5)
vetores com duas dimensões. o formato padrão de tabela no R é o
. Para criar um data.frame, usamos a função dataframe()
~sua vida~~ seu código
ar o R, vai perder tudo que você fez
ve seu script!
arquivo de texto sem formatação com a extensão _.R_ ou _.r_, so
222 Você pode editar no bloco de notas, Notepad++, Gedit,Vim, EMACS, etc, mas evite
usar Word, ele insere formatações que atrapalham tudo
223 Use o Rstudio
226 ## Slide with R Output
223:14 Salvando ~~sua vida~~ seu código
R Markdown
Console Terminal R Markdown
.../teste-Intro-Linguagem-R/aula_1.Rmd
output file: aula_1.knit.md
"C:/Program Files/RStudio/bin/pandoc/pandoc" +RTS -K512m -RTS aula_1.utf8.md --to html --from markdown+autolink_bare_uris+ascii_identifiers+tex_math_single_backslash --output aula_1.html --smart --email-obfuscation none --self-contained --variable transition=0.4 --template "C:\Users\Vitor\Documents\R\win-library\3.4\rmarkdown\rmd\ioslides\default.html" --include-in-header "C:\Users\Vitor\AppData\Local\Temp\RtmpwfsDvB\rmarkdown-str5c449c3134.html" --mathjax --variable "mathjax-url:https://mathjax.rstudio.com/latest/MathJax.js?config=TeX-AMS-MML_HTMLorMML"
"C:/Program Files/RStudio/bin/pandoc/pandoc" +RTS -K512m -RTS aula_1.utf8.md --to ioslides_presentation.lua --from markdown+autolink_bare_uris+ascii_identifiers+tex_math_single_backslash --output "C:\Users\Vitor\AppData\Local\Temp\RtmpwfsDvB\ioslides-output5c635810ab.html" --slide-level 2
Output created: aula_1.html
Environment History Connections Git
Import Dataset
List
Global Environment
Environment is empty
Files Plots Packages Help Viewer
data.frame.
R: Data Frames
Find in Topic
Objects passed to data.frame should have the same number of rows, but atomic vectors (see is.vector), factors and character vectors protected by I will be recycled a whole number of times if necessary (including as elements of list arguments).
If row names are not supplied in the call to data.frame, the row names are taken from the first component that has suitable names, for example a named vector or a matrix with rownames or a data frame. (If that component is subsequently recycled, the names are discarded with a warning.) If row.names was supplied as NULL or no suitable component was found the row names are the integer sequence starting at one (and such row names are considered to be 'automatic', and not preserved by as.matrix).
If row names are supplied of length one and the data frame has a single row, the row.names is taken to specify the row names and not a column (by name or number).
Names are removed from vector inputs not protected by I.
default.stringsAsFactors is a utility that takes getOption("stringsAsFactors") and ensures the result is TRUE or FALSE (or throws an error if the value is not NULL).
Value
15:55

Figure 4:

Mas salvar onde?

O R usa uma coisa chamada diretório de trabalho. Ele encontra automaticamente tudo que esteja nesse diretório e salva as coisas nesse diretório.

```
getwd() ##exibe o diretório de trabalhoda sessão atual
```

```
## [1] "C:/Users/Vitor/Dropbox/disciplinas/Introducao ao us
```

Para mudar o diretório de trabalho, use a função *setwd()*

```
setwd("C:/Curso_R/aula1") ##muda o diretório de trabalho da
```

ATENÇÃO O endereço deve vir sempre em aspas, e use / ao invés de \ no Windows

Cada coisa, um script!

Separe suas funções e análises em scripts diferentes, vai facilitar *muito* sua vida.

Use nomes curtos e descritivos. *Funcao1.R, funcao2.R, regressao.R, regressaoLogisticaMultivaridaAmostra1BTSversão1.r* é pedir pra sofrer.

Uma pasta para cada coisa:
~/doutorado/ - /dados
- /analises - /figuras - /manuscrito

Você pode chamar um script de dentro do outro, usando o comando *source(endereço/script.R)*. O endereço pode inclusive ser um site

Criando seu script

Note que o RStudio muda automaticamente as cores do código. Isto é chamado *syntax highlighting*, e serve pra facilitar a leitura. o que cada cor quer dizer?

O Rstudio também insere automaticamente um), } ou] quando você digita (, {, ou [, o que diminui a maior parte dos erros de digitação

Crie um script com os codigos que usamos até aqui, comentando cada linha com um # após o comando, dizendo o que ele faz

Salvando tudo junto

Se você quiser salvar tudo que está na memória do R, use a função *save.image()*, com a extensão *.RData*

```
save.image(file="tudoJunto.RData")
```

Lendo arquivos

O R consegue ler arquivos externos, se você disser a ele o que esperar

```
help("read.table")
```

Existem algumas funções com opções padrão que facilitam o uso da *read.table()*

read.csv() lê arquivos usando como padrão o ponto (.) como sepador de decimais e a vírgula (,) como separador de colunas.

read.csv2() usa a vírgula como separador de decimais e o ponto-e-vírgula (;) como separador de colunas

Existem pacotes específicos para ler e escrever .docx, .xlsx, .pptx, SPSS, SAS, shapefiles, e praticamente qualquer tipo de arquivo

Uma das principais habilidades para se usar o R é encontrar o pacote certo pra fazer o que você quer

Argumentos importantes

```
#argumento são sempre separados por vírgulas na chamada da
#no começo, é bom colocar cada argumento em uma linha para
#TRUE habilita a opção, FALSE desabilita
arquivoLido =read.csv(file = "arquivo.csv",
                      header = TRUE, #interpreta a primeira li
                      as.is = TRUE , # não altera interpreta o
                      sep = ",", # define o caractere que deve
                      dec = ".", #define o caractere que deve
)
```

O comando acima cria uma variável chamada *arquivoLido*, com o conteúdo de *"arquivo.csv"*

O arquivo original não é alterado, não importa o que você fizer dentro do R

Se você quiser salvar suas alterações em um arquivo, use write.table(), write.csv() ou write.csv2()

```
write.csv(x= arquivoLido, #objeto a ser gravado
```