

Conhecendo o R

História e popularidade

Prof. Dr. **Walmes Zeviani**

2021-10-28

O que é o R?

Conhecendo a linguagem

Definição



Logo do R.

Detalhes

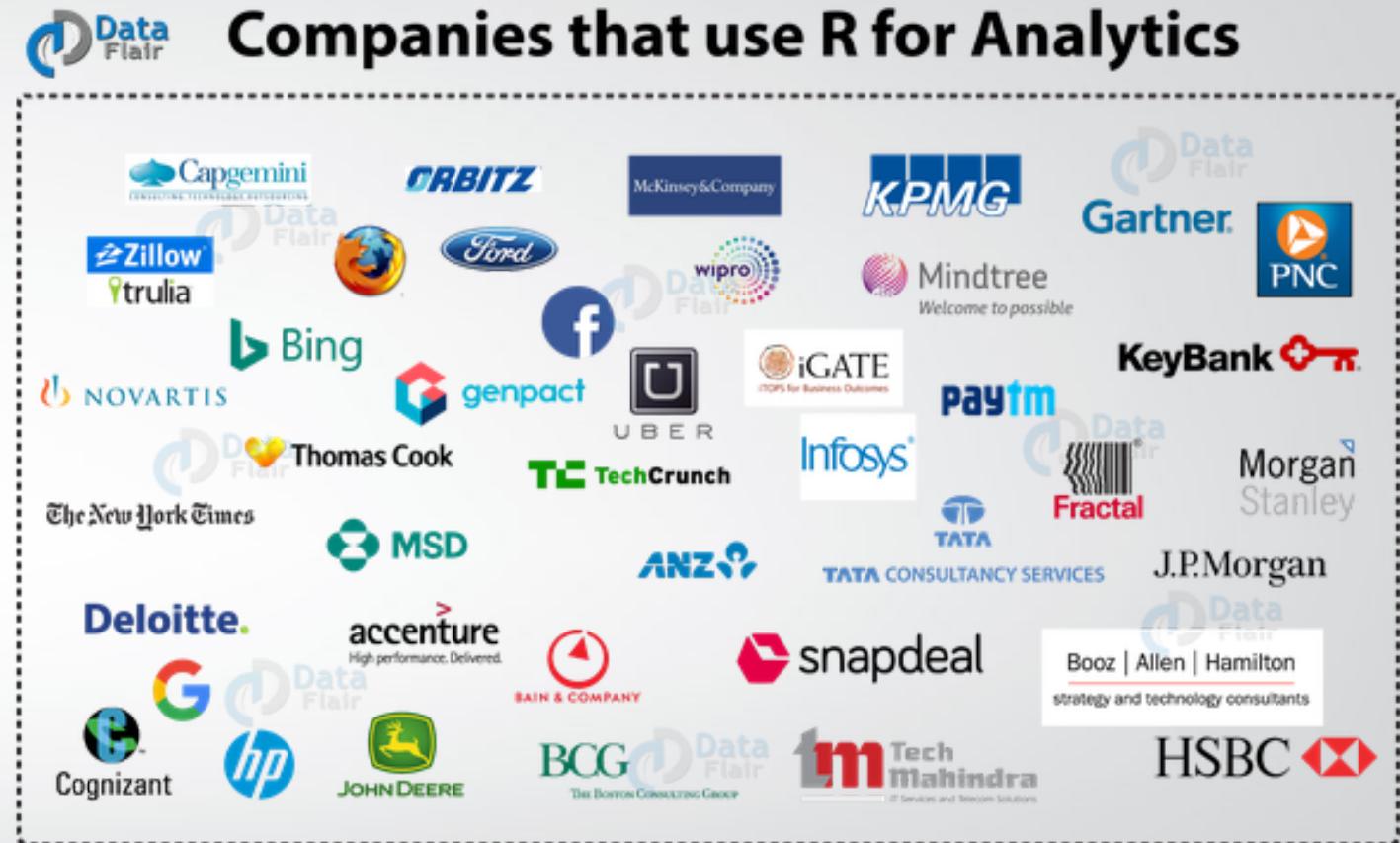
- ▶ É uma linguagem de programação **livre** e de **código aberto** (um projeto GNU).
- ▶ É um ambiente de software para **computação estatística e gráficos**.
- ▶ <https://www.r-project.org/about.html>.

Quem usa o R?

- ▶ Tem imensa popularidade acadêmica e científica.
 - ▶ Universidades e escolas.
 - ▶ Institutos de pesquisa.
 - ▶ Órgãos governamentais.
- ▶ Com a Era da Informação ou Era Big Data:
 - ▶ Start Ups.
 - ▶ Empresas privadas.
 - ▶ Grandes empresas de informação.

R nas empresas

Empresas que usam o R para análise de dados⁶.



Como obter o R?



[CRAN
Mirrors](#)
[What's new?](#)
[Task Views](#)
[Search](#)

[About R](#)
[R Homepage](#)
[The R Journal](#)

[Software](#)
[R Sources](#)
[R Binaries](#)
[Packages](#)
[Other](#)

[Documentation](#)
[Manuals](#)
[FAQs](#)
[Contributed](#)

The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- [Download R for Linux](#)
- [Download R for \(Mac\) OS X](#)
- [Download R for Windows](#)

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2021-03-31, Shake and Throw) [R-4.0.5.tar.gz](#), read [what's new](#) in the latest version.
- Sources of [R alpha and beta releases](#) (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are [available here](#). Please read about [new features and bug fixes](#) before filing corresponding feature requests or bug reports.
- Source code of older versions of R is [available here](#).
- Contributed extension [packages](#)

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

Pacotes agrupados por tarefas/áreas

Topics	
Bayesian	Bayesian Inference
ChemPhys	Chemometrics and Computational Physics
ClinicalTrials	Clinical Trials and Studies
Cluster	Clustering and Classification
Databases	Databases and Data Management
DifferentialEquations	Differential Equations
Distributions	Distributions and Probability
Econometrics	Econometrics and Statistical Methods
Environmetrics	Environmental Monitoring and Statistics
ExperimentalDesign	Experimental Design and Analysis
ExtremeValue	Extreme Value Theory
Finance	Financial Modelling and Analysis
FunctionalData	Functional Data Analysis
Genetics	Genetic Data Analysis
Graphics	Statistical Graphics and Visualization
HighPerformanceComputing	High Performance Computing
Hydrology	Hydrological Modelling and Analysis
MachineLearning	Machine Learning and Data Mining
MedicalImaging	Medical Imaging and Signal Processing
MetaAnalysis	Meta-Analysis and Evidence Synthesis
MissingData	Handling Missing Data
ModelDeployment	Model Deployment and Shiny Apps
Multivariate	Multivariate Statistics and Data Analysis
NaturalLanguageProcessing	Natural Language Processing
NumericalMathematics	Numerical Mathematics and Computing
OfficialStatistics	Official Statistics and 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 and Replication
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
TeachingStatistics	Teaching Statistics and Education
TimeSeries	Time Series Analysis
WebTechnologies	Web Technologies and Services
gR	Graphical Models in R

<https://cran.r-project.org/web/views/>

Oportunidades de emprego no Brasil

The screenshot shows a LinkedIn job search interface. The search bar at the top has 'Brazil' selected. Below the search bar are several filter options: Date Posted, Experience Level, Job Type, Company, and More Filters. A search button and a magnifying glass icon are also present. The main results section displays 134 results for "R Programming in Brazil" (3 new). Two job listings are visible:

- Senior Project Engineer · Data Scientist** at Choice Technologies Holding Sàrl, Rio de Janeiro Area, Brazil. Key Knowledge, Skills and Experience - Minimum qualifications. 10 years of relevant work experience (e.g., as a ...). Posted 3 weeks ago · Easy Apply. This listing is highlighted with an orange oval around the job title.
- Programa de Férias ou Estágio · Analytics** at Accenture Brasil, São Paulo, BR. This new vacation program offers the professional the opportunity to know and be part of the Analytics world and experience the reality ... Posted 1 day ago.

To the right of the second listing, there is a sidebar with the text "CHOICE" and a "CHOICE" logo. It also lists the job title, company, and location again, along with the posting date and number of applicants. Below this, it says "Easy Apply". At the bottom, it says "Posted by" and shows a profile picture of Fernanda Casado, Human Resources Director at Choice Technologies Holding Sàrl.

Clique no link para estatísticas recentes: <https://www.linkedin.com/jobs/search?keywords=R%20Programming>

Algumas funcionalidades de destaque

Comunicação de dados

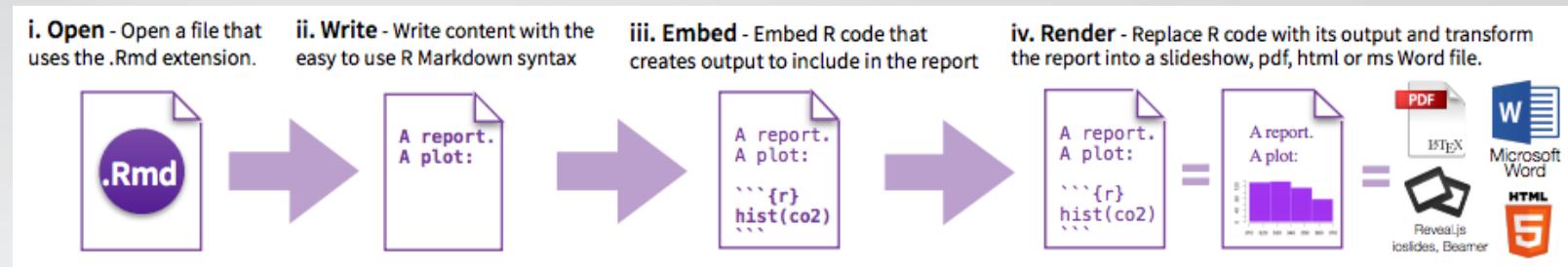
Recursos gráficos

- ▶ Gráficos estáticos com `ggplot2`⁸.
- ▶ Recursos interativos⁹.
- ▶ Estatísticas da Hansenise no Paraná:
<http://www.leg.ufpr.br/~paulojus/hansenise/>.
- ▶ Vídeos sobre recursos gráficos:
<https://youtu.be/KvWCppWip7I>.

Dashboards em Shiny

- ▶ Dashboards com R feitos em Shiny.
- ▶ Monitoramento de Covid19 - Brasil e mundo:
<http://shiny.leg.ufpr.br/elias/covid19time/>.
- ▶ Monitoramento de Covid19 - Estados brasileiros:
<http://leg.ufpr.br/~wagner/covid/Estados.html>.

Documentos dinâmicos



Tutorias em vídeo

- ▶ Introdução ao Rmarkdown · Fernando Mayer.
- ▶ Pesquisa reproduzível com R Markdown · Fernando Mayer.

Supremo em Ação

- ▶ Relatório anual: <https://www.cnj.jus.br/pesquisas-judiciais/supremo-em-acao/>
- ▶ Aplicação Web: <http://dpj.cnj.jus.br/supremo-em-acao/>.

Métodos estatísticos

- ▶ É a maior área de competência do R.
- ▶ Ajuste de modelos estatísticos e de aprendizado de máquina.
- ▶ Procedimentos de simulação Monte Carlo (reamostragens e afins).
- ▶ Métodos multivariados, métodos não paramétricos, etc, etc, etc.



Material didáticos indicados

São as principais referências dos nossos cursos

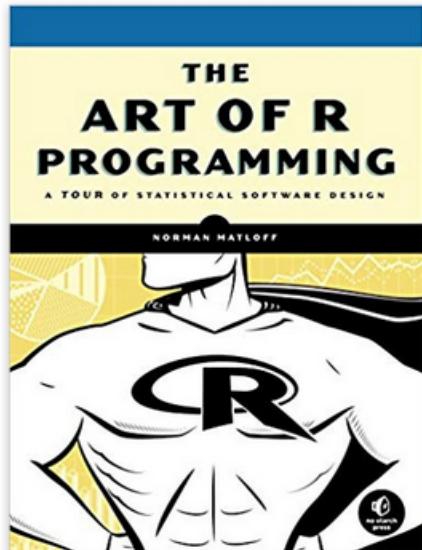
The Art of R Programming: A Tour of Statistical Software Design 1st Edition

by Norman Matloff (Author)



135 customer reviews

Look inside ↓



ISBN-13: 978-1593273842

ISBN-10: 9781593273842

Why is ISBN important? ▾

Kindle \$31.70

Paperback \$18.00 - \$27.47

Other Sellers

See all 3 versions

Rent

Due Date: Dec 17, 2019 [Rental Details](#)

- FREE return shipping at the end of the semester.
- Access codes and supplements are not guaranteed with rentals.

In Stock. Rented from eCampus_, Fulfilled by Amazon

\$18.00

List Price: \$39.95

Save: \$21.95 (55%)

Deliver to Brazil

Add to Rental Cart

Buy used

\$23.52

Buy new

\$27.47

More Buying Choices

150 offers from \$10.17

[See All Buying Options](#)

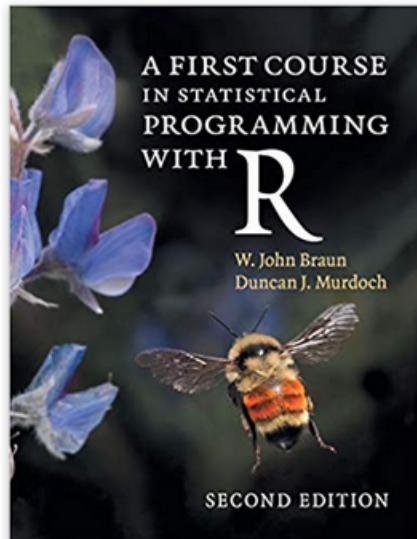
<https://www.amazon.com/Art-Programming-Statistical-Software-Design/dp/1593273843>

A First Course in Statistical Programming with R 2nd Edition

by W Braun (Author)

★★★★★ 1 customer review

[Look inside](#) ↓



ISBN-13: 978-1107576469

ISBN-10: 1107576466

[Why is ISBN important?](#) ▾

Kindle

Paperback

Other Sellers

\$35.45

\$31.81

See all 4 versions

Buy new

\$31.81

List Price: \$44.99

Save: \$13.18 (29%)

23 New from \$31.81

In Stock.

Ships from and sold by Amazon.com.

This item ships to Brazil. Want it Wednesday, Aug. 7? Order within 2 hrs 25 mins and choose AmazonGlobal Priority Shipping at checkout. [Learn more](#)

[Deliver to Brazil](#)

Qty: 1 ▾



Add to Cart



Buy Now

More Buying Choices

45 offers from \$31.81

23 New from \$31.81 | 21 Used from \$34.18 | 1 Rentals from \$31.91

[See All Buying Options](#)

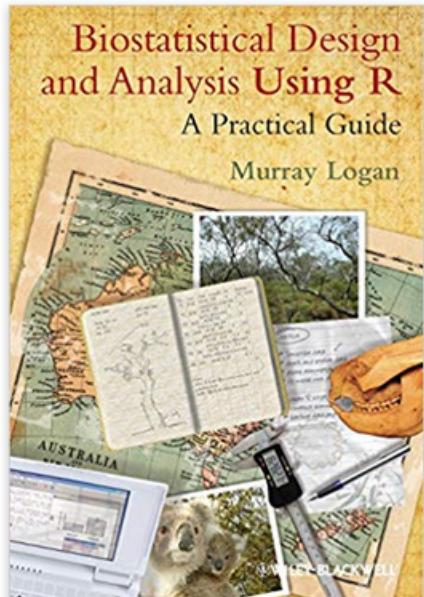
<https://www.amazon.com/First-Course-Statistical-Programming-ebook/dp/B01HTT8YLG>

Biostatistical Design and Analysis Using R: A Practical Guide 1st Edition

by Dr Murray Logan (Author)

★★★★★ 7 customer reviews

Look inside ↓



ISBN-13: 978-1405190084
ISBN-10: 9781405190084

eTextbook	Hardcover	Paperback	Other Sellers
\$69.39	\$126.94	\$37.00 - \$80.94	See all 9 versions
<input checked="" type="radio"/> Rent		\$37.00	
	Due Date: Dec 17, 2019 Rental Details		List Price: \$85.25 Save: \$48.25 (57%)
	<ul style="list-style-type: none">FREE return shipping at the end of the semester.Access codes and supplements are not guaranteed with rentals.		<input checked="" type="checkbox"/> Deliver to Brazil
	In Stock. Rented from apex_media , Fulfilled by Amazon		
	Want it Wednesday, Aug. 7? Order within 17 hrs 51 mins and choose AmazonGlobal Priority Shipping at checkout. Details		
<input type="radio"/> Buy used		\$57.96	
<input type="radio"/> Buy new		\$80.94	
More Buying Choices		34 offers from \$37.00	
16 New from \$52.87 17 Used from \$57.96 1 Rentals from \$37.00			See All Buying Options

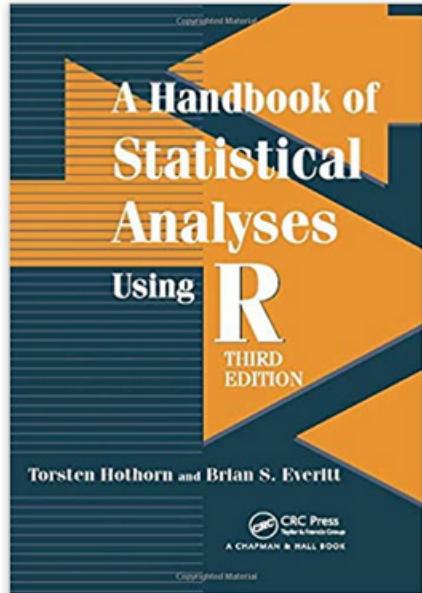
<https://www.amazon.com/Biostatistical-Design-Analysis-Using-Practical-ebook/dp/B005TI32C6>

A Handbook of Statistical Analyses using R, Third Edition 3rd Edition

by Torsten Hothorn ▾ (Author), Brian S. Everitt (Contributor)

★★★★★ ▾ 12 customer reviews

Look inside ↓



ISBN-13: 978-1482204582

ISBN-10: 1482204584

eTextbook	Hardcover	Paperback	Other Sellers
\$56.88	\$193.96	\$47.48 - \$66.80	See all 6 versions
<input type="radio"/> Buy used			\$47.48
<input checked="" type="radio"/> Buy new			\$66.80
In Stock. Ships from and sold by Amazon.com.		List Price: \$72.95 Save: \$6.15 (8%)	
This item ships to Brazil. Want it Wednesday, Aug. 7? Order within 17 hrs 52 mins and choose AmazonGlobal Priority Shipping at checkout. Learn more			
<input type="radio"/> Deliver to Brazil			
Qty: 1 ▾			
 Add to Cart			
 Buy Now			

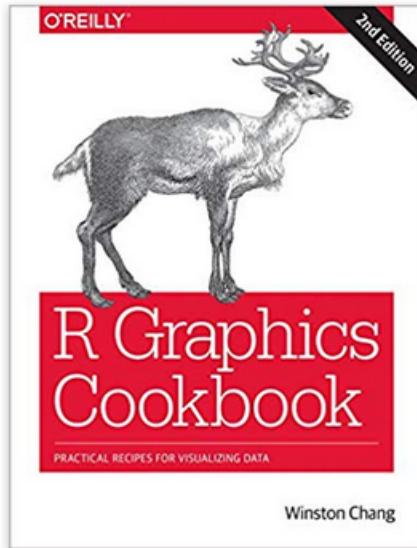
<https://www.amazon.com/Handbook-Statistical-Analyses-using-Third/dp/1482204584>

R Graphics Cookbook: Practical Recipes for Visualizing Data 2nd Edition

by Winston Chang ▾ (Author)

★★★★★ ▾ 1 customer review

Look inside ↓



Kindle \$60.47

Paperback
\$24.34 - \$48.67

Other Sellers
See all 2 versions

Rent

Due Date: Dec 17, 2019 [Rental Details](#)

- FREE return shipping at the end of the semester.
- Access codes and supplements are not guaranteed with rentals.

In Stock. Rented from [Amazon Warehouse](#), Fulfilled by Amazon

\$24.34

List Price: \$69.99

Save: \$45.65 (65%)

Deliver to Brazil

Add to Rental Cart

Buy new

\$48.67

More Buying Choices

44 offers from \$24.34

32 New from \$48.67 | 11 Used from \$48.72 | 1 Rentals from \$24.34

[See All Buying Options](#)

ISBN-13: 978-1491978603

ISBN-10: 1491978600

[Why is ISBN important?](#) ▾

prime student

College student? Get FREE shipping and exclusive deals

[LEARN MORE](#)

<https://www.amazon.com/Graphics-Cookbook-Practical-Recipes-Visualizing/dp/1491978600>

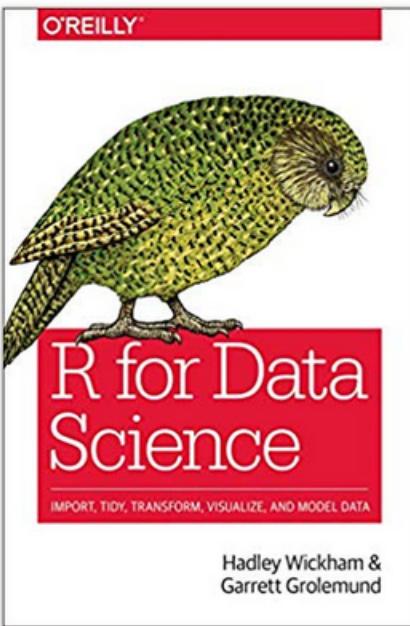
R for Data Science: Import, Tidy, Transform, Visualize, and Model Data 1st Edition

by Hadley Wickham ▾ (Author), Garrett Grolemund ▾ (Author)

★★★★★ ▾ 124 customer reviews

#1 Best Seller in Data Processing

Look inside ↓



Kindle

\$21.89

Paperback

\$18.17

Other Sellers

See all 3 versions

Buy new

In Stock.

Ships from and sold by Amazon.com.

\$18.17

List Price: \$49.99

Save: \$31.82 (64%)

52 New from \$18.17

© Deliver to Brazil



Add to Cart



Buy Now

More Buying Choices

89 used & new from \$18.17

See All Buying Options

52 New from \$18.17 | 37 Used from \$25.99

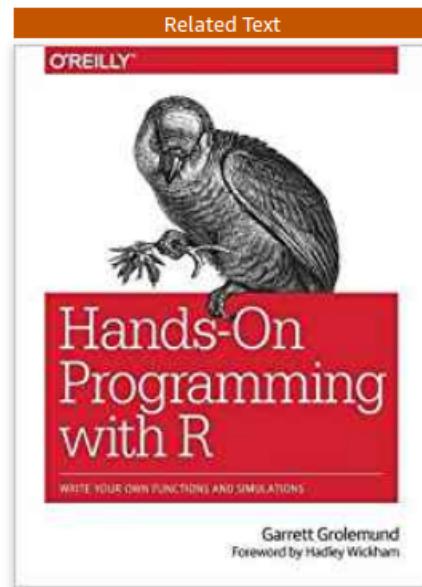
<https://www.amazon.com/Data-Science-Transform-Visualize-Model/dp/1491910399>

Hands-On Programming with R: Write Your Own Functions and Simulations

by Garrett Grolemund (Author), Hadley Wickham (Foreword)

4.5 out of 5 stars 19 customer reviews

[Look inside](#) ↓



Kindle \$14.74

Paperback
\$23.98 - \$31.76

Other Sellers
[See all 4 versions](#)

Buy used

\$23.98

Buy new

\$31.76

In Stock.

List Price: \$39.99
Save: \$8.23 (21%)

Ships from and sold by Amazon.com.

39 New from \$31.76

This item ships to Brazil. Want it Tuesday, Aug. 6? Order within 12 hrs 24 mins and choose AmazonGlobal Priority Shipping at checkout. [Learn more](#)

[Deliver to Brazil](#)

Qty: 1



Add to Cart



Buy Now

ISBN-13: 978-1449359010

ISBN-10: 1449359019

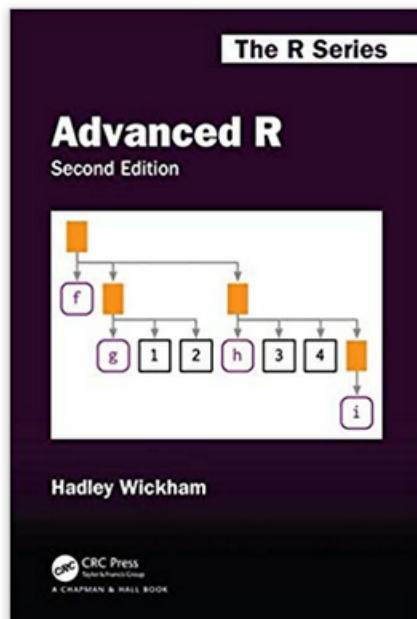
<https://www.amazon.com/Hands-Programming-Write-Functions-Simulations/dp/1449359019>

Advanced R, Second Edition (Chapman & Hall/CRC The R Series) 2nd Edition

by Hadley Wickham (Author)

★★★★★ 3 customer reviews

[Look inside](#)



ISBN-13: 978-0815384571

Kindle \$56.01

Hardcover \$136.41

Paperback \$54.64

Other Sellers

[See all 3 versions](#)

Buy new

\$54.64

List Price: \$59.95

Save: \$5.31 (9%)

24 New from \$54.64

In Stock.

Ships from and sold by Amazon.com.

This item ships to Brazil. Want it Tuesday, Aug. 6? Order within 10 hrs 54 mins and choose AmazonGlobal Priority Shipping at checkout. [Learn more](#)

Deliver to Brazil

Qty: 1



Add to Cart



Buy Now

31 used & new from \$54.64

[See All Buying Options](#)

More Buying Choices

24 New from \$54.64 | 7 Used from \$59.79

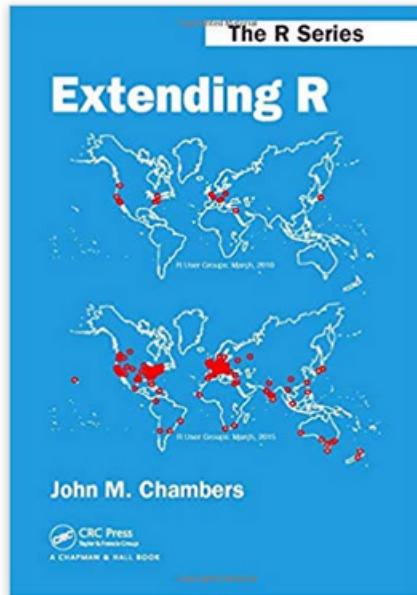
<https://www.amazon.com/Advanced-Second-Chapman-Hall-CRC/dp/0815384572>

Extending R (Chapman & Hall/CRC The R Series) 1st Edition

by John M. Chambers (Author)

★★★★★ 4 customer reviews

Look inside ↓



ISBN-13: 978-1498775717
ISBN-10: 1498775713

eTextbook	Hardcover	Paperback	Other Sellers
\$74.14	\$122.03 - \$132.65	\$57.66	See all 5 versions
Buy new		\$57.66	
In Stock.		List Price: \$74.95 Save: \$17.29 (23%)	
Ships from and sold by Amazon.com.		29 New from \$57.47	
This item ships to Brazil. Want it Wednesday, Aug. 7? Order within 14 hrs 23 mins and choose AmazonGlobal Priority Shipping at checkout. Learn more			
<input checked="" type="radio"/> Deliver to Brazil			
Qty: 1 ▾			
<input type="button" value="Add to Cart"/>			
<input type="button" value="Buy Now"/>			
More Buying Choices		45 used & new from \$57.46	
29 New from \$57.47 16 Used from \$57.46		See All Buying Options	

<https://www.amazon.com/Extending-Chapman-Hall-John-Chambers-ebook/dp/B01GRHCLG0>

Considerações Finais

- ▶ R é uma linguagem e ambiente para computação estatística e gráficos livre e de código aberto.
- ▶ É extensível, escalável, interopera com várias outras linguagens/softwares.
- ▶ É cada vez mais popular na comunidade acadêmica.
- ▶ Tem sido adotado exponencialmente na comunidade profissional para atividades de análise de dados.
- ▶ O R pode ser empregado em várias etapas do processo de análise de dados:
 - ▶ Aquisição, importação e tratamentos dos dados.
 - ▶ Confecção de visualizações estáticas e dinâmicas (gráficos, mapas, etc).
 - ▶ Elaboração de relatórios dinâmicos/reproduzível.
 - ▶ Disponibilização de conteúdo via APIs e dashboards.
 - ▶ Ajuste de modelos estatísticos (diagnósticos, preditivos e prescritivos).
 - ▶ Uso de métodos de aprendizado de máquinas.



Logo do R.

Perguntas?



Configuração do ambiente de trabalho

Downloads e instalações

Prof. Dr. **Walmes Zeviani**

2021-10-28

Download e instalação

Configurando o ambiente de trabalho

Download e instalação do R

Roteiro

1. Instalar a linguagem R.
 - ▶ É a **linguagem de programação**.
 - ▶ Ela que faz as "contas" e recebe o crédito.
2. Instalar um editor/IDE de preferência.
 - ▶ Editores/IDEs são facilitadores para o desenvolvimento de código.
 - ▶ IDEs: ambiente integrado de desenvolvimento.
 - ▶ RStudio.
 - ▶ Tinn-R.
 - ▶ Editores: são ferramentas gerais habilitadas para as linguagens usando plugins.
 - ▶ GNU Emacs, Spacemacs ou Doom-Emacs usando ESS.
 - ▶ Vim.
 - ▶ Visual Studio Code (VS Code).

Links para downloads

- ▶ Downloads do R:
<http://cran.r-project.org/>.
- ▶ Windows
 - ▶ Baixar o `*.exe` e instalar com qualquer outro software.
 - ▶ Instruções em: **base** para Windows.
 - ▶ Print screens do processo aqui: [r-installation-screenshots](#).
- ▶ Mac OS X:
 - ▶ Seguir instruções no CRAN: [Mac OS X](#).
 - ▶ Tutorial de instalação (en): [install-r-and-rstudio-on-mac](#)
- ▶ GNU Linux:
 - ▶ Conforme a distribuição, seguir instruções em: [GNU Linux](#).

Instalação do R no Ubuntu GNU Linux

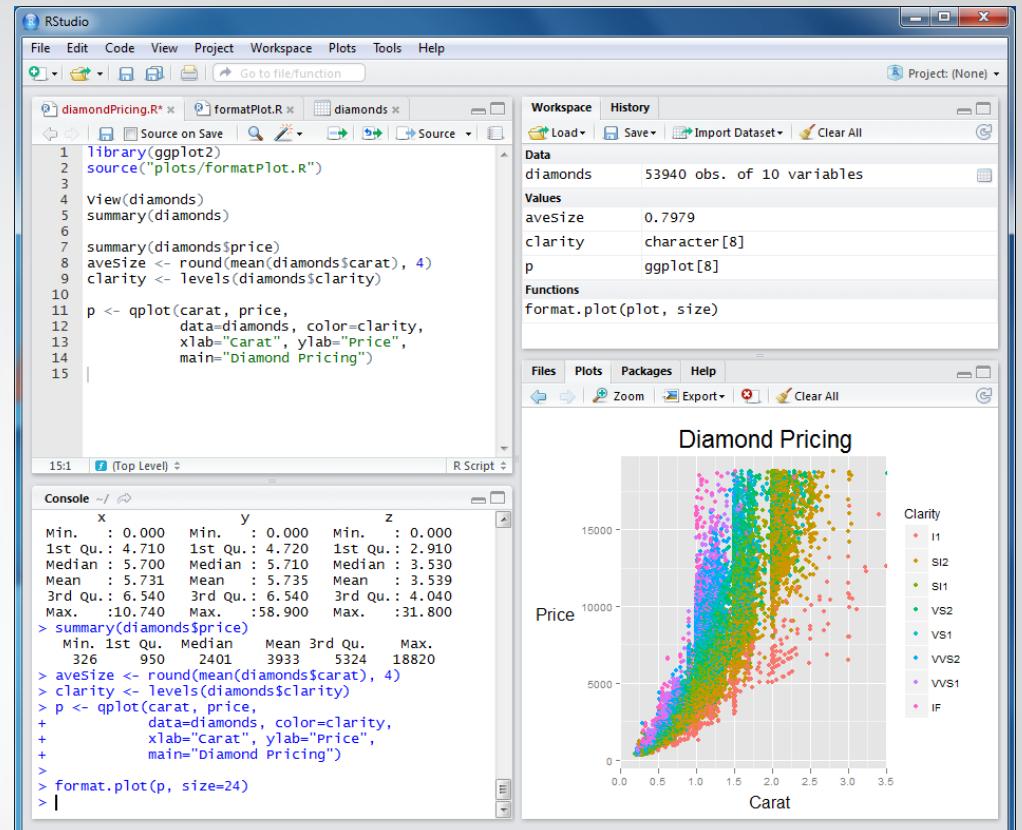
- ▶ Instalação conforme documentação oficial do CRAN.
- ▶ Executar em um Terminal.

```
# Adiciona chave.  
apt-key adv --keyserver keyserver.ubuntu.com \  
    --recv-keys E298A3A825C0D65DFD57CBB651716619E084DAB9  
  
# Adiciona endereço à lista de repositórios.  
sudo add-apt-repository \  
    "deb https://cloud.r-project.org/bin/linux/ubuntu $(lsb_release -cs)-cran40/"  
  
# Atualiza lista de repositórios.  
sudo apt-get update  
  
# Faz a instalação do R.  
sudo apt-get build-dep -y r-base
```

RStudio IDE

Detalhes

- É um ambiente mais popular para trabalhar com a linguagem R.
- É certamente o mais rápido de aprender.
- A IDE vem totalmente habilitada para o uso do R em todos os aspectos.
- Download do RStudio (recomendado):
[https://www.rstudio.com/products/rstudio/download/.](https://www.rstudio.com/products/rstudio/download/)

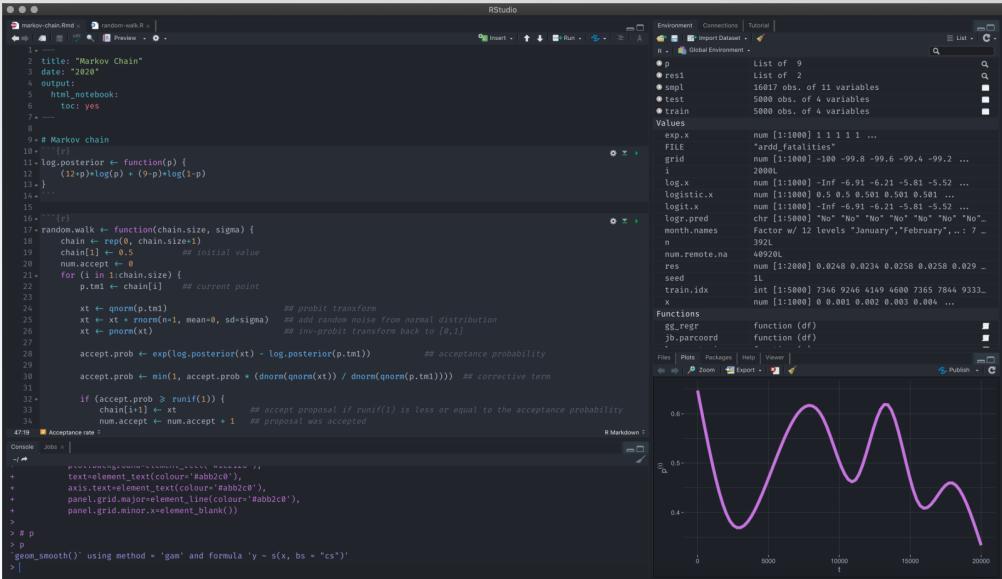


Print screen do RStudio.

RStudio IDE

Instalação

- ▶ Windows: baixar `*.exe` e instalar.
- ▶ Mac OS X: descrição em [install-r-and-rstudio-on-mac](#).
- ▶ GNU Linux:
 - ▶ Ubuntu: baixar o `*.deb` e instalar:
`sudo dpkg -i <nome>.deb`.
 - ▶ Outras distros: instalar via `tar.gz` ou `*.rpm`.



Print screen do RStudio em tela toda e ambiente escuro.

Fonte.

Outros editores para R

Não custa pensar fora da caixinha

Outros editores · Tinn-R (IDE)

- ▶ [https://tinn-r.org/pt/.](https://tinn-r.org/pt/)
- ▶ IDE para Windows.
- ▶ Desenvolvida por brasileiros.
 - ▶ Prof. Dr. José Cláudio Faria e colaboradores no LEC/UESC.

The screenshot shows the Tinn-R IDE interface. On the left is the R script editor with code in R and LaTeX. The middle section is the terminal window displaying R command-line output. The right section is a help viewer showing the documentation for the `sd` function.

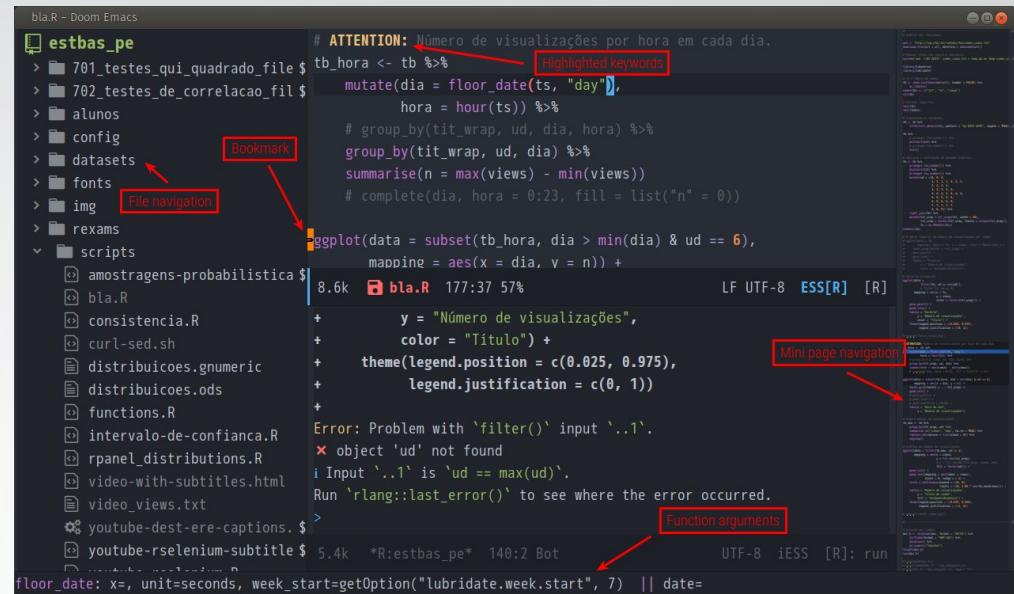
```
\usepackage{geometry}
\usepackage{sectsty}
\allsectionsfont{\sffamily\raggedright}
\geometry{verbose, a4paper,
tmargin=1.5cm, bmargin=1.5cm,
lmargin=1.5cm, rmargin=1.0cm,
headsep=5mm, footskip=0cm}
\usepackage{Sweave}
\SweaveOpts{concordance=TRUE}
\begin{document}
<<echo=F>>=
alunos <- read.table('dados/alunos.txt',
header=TRUE,
fileEncoding='utf-8'
)
#alunos <- read.table('dados/alunos_ficti
#
header=TRUE,
fileEncoding='utf-8'
)
aluno <- alunos[6,
1]
#
# Matrícula
# 1 201620294 CLEONE JÚNIO LELIS SA
# 2 201420562 JOSÉ TALMON MELO JÚ
# 3 201320310 LUCAS DE SOUZA GA
# 4 201520229 MARCEL SILVA HENR
sd(x, na.rm=FALSE)

```

Print screen Tinn-R. Fonte.

Outros editores - GNU Emacs

- ▶ [https://www.gnu.org/software/emacs/.](https://www.gnu.org/software/emacs/)
- ▶ É um dos editores mais maduros e estáveis que existem, continua moderno, em constante evolução e é amplamente utilizado.
- ▶ O primeiro a ter suporte para o R.
- ▶ Suporta inúmeras linguagens de programação e frameworks.
- ▶ O pacote **ESS** (Emacs Speaks Statistics) dá suporte para R, S-Plus, SAS, Stata e Julia.
- ▶ Download do Emacs + ESS (opcional):
 - ▶ Usuários Windows:
<http://vgoulet.act.ulaval.ca/en/emacs/>.
 - ▶ Usuários Linux instalam por
`sudo apt-get install emacs`.
 - ▶ Ou podem instalar via `*.tar.gz`.
- ▶ Existem variantes como Spacemacs e Doom-Emacs.



The screenshot shows the Doom-Emacs interface with the following elements highlighted:

- File navigation:** A red box highlights the sidebar containing file icons and a tree view of directory contents.
- Bookmark:** A red box highlights a small icon in the sidebar.
- Highlighted keywords:** A red box highlights several words in blue in the R code area.
- Mini page navigation:** A red box highlights a small window in the bottom right corner showing a preview of the current buffer.
- Function arguments:** A red box highlights a tooltip or a callout pointing to the function arguments of a selected R command.

The R code in the buffer is:

```
bla.R - Doom Emacs
estbas_pe
# ATTENTION: Número de visualizações por hora em cada dia.
tb_hora <- tb %>%
  mutate(dia = floor_date(ts, "day"),
        hora = hour(ts)) %>%
  group_by(tit_wrap, ud, dia, hora) %>%
  group_by(tit_wrap, ud, dia) %>%
  summarise(n = max/views) - min/views)
# complete(dia, hora = 0:23, fill = list("n" = 0))

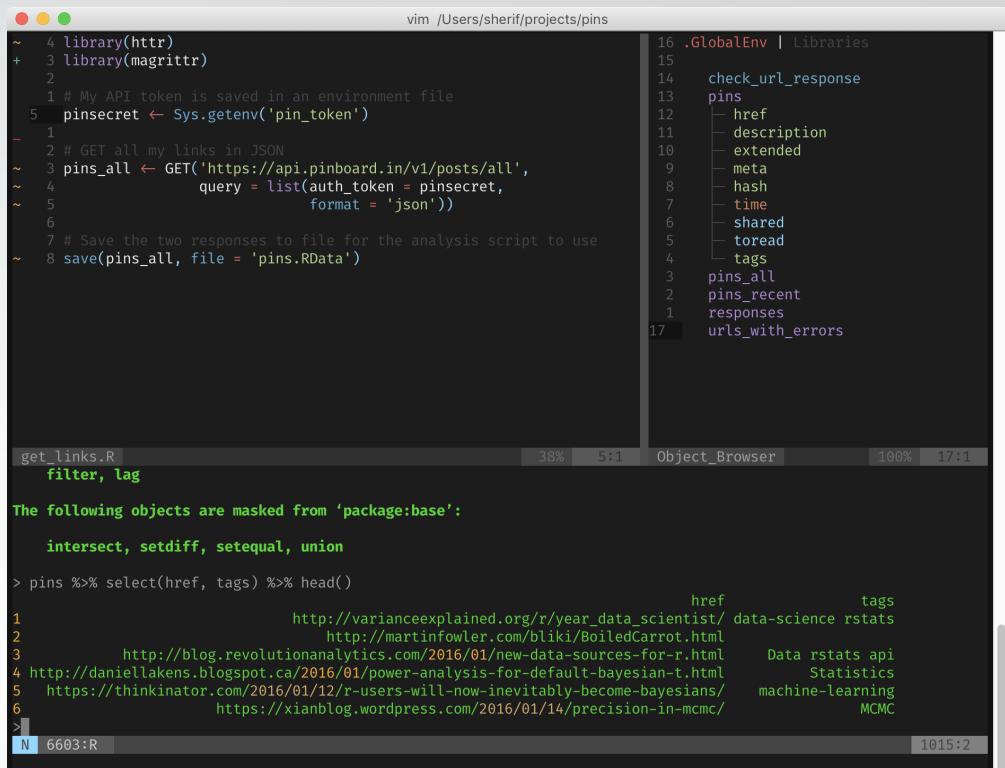
ggplot(data = subset(tb_hora, dia > min(dia) & ud == 6),
       mapping = aes(x = dia, y = n)) +
  +   y = "Número de visualizações",
  +   color = "Título") +
  +   theme(legend.position = c(0.025, 0.975),
  +         legend.justification = c(0, 1))
Error: Problem with `filter()` input `..1`.
✖ object 'ud' not found
ℹ Input `..1` is `ud == max(ud)`.

Run `rlang::last_error()` to see where the error occurred.
>
youtube-dest-ere-captions.$
youtube-rselenium-subtitle$ 5.4k *R:estbas_pe* 140:2 Bot
youtube-rselenium.R
floor_date: x=, unit=seconds, week_startgetOption("lubridate.week.start", 7) || date=
```

Print screen do Doom-Emacs com R. Fonte.

Outros editores - Vim

- ▶ [https://www.vim.org/.](https://www.vim.org/)
- ▶ Assim como Emacs, é um editor maduro e estável, amplamente usado para inúmeros propósitos.
- ▶ Suporte para R com o plugin **Nvim-R**:
<https://github.com/jalvesaq/Nvim-R>
<https://nbcgib.uesc.br/lec/software/editores/nvim-r>
https://www.vim.org/scripts/script.php?script_id=2628.
- ▶ Plugins desenvolvido por brasileiros:
 - ▶ Jakson Alves de Aquino.
 - ▶ José Cláudio Faria.



A screenshot of the Vim text editor interface. The main window displays R code for interacting with a pinboard API. The sidebar on the right shows a tree view of global environment variables, including 'pins' and its sub-objects like 'href', 'description', 'extended', etc. The bottom status bar shows the file name 'get_links.R', the current line '6603:R', and the time '10:15:2'. The bottom right corner of the status bar also shows the Vim version '17.1'.

```
vim /Users/sherif/projects/pins
+ 4 library(httr)
+ 3 library(magrittr)
+ 2
+ 1 # My APT token is saved in an environment file
+ 5 pinsecret ← Sys.getenv('pin_token')
+ 1
+ 2 # GET all my links in JSON
+ 3 pins_all ← GET('https://api.pinboard.in/v1/posts/all',
+ 4                     query = list(auth_token = pinsecret,
+ 5                               format = 'json'))
+ 6
+ 7 # Save the two responses to file for the analysis script to use
+ 8 save(pins_all, file = 'pins.RData')

. GlobalEnv | Libraries
15
14 check_url_response
13 pins
12   href
11   description
10   extended
9   meta
8   hash
7   time
6   shared
5   toread
4   tags
3 pins_all
2 pins_recent
1 responses
17 urls_with_errors

get_links.R 38% 5:1 Object_Browser 100% 17:1
filter, lag
The following objects are masked from 'package:base':
  intersect, setdiff, setequal, union
> pins %>% select(href, tags) %>% head()
1                               href          tags
2                               http://varianceexplained.org/r/year_data_scientist/ data-science rstats
3                               http://martinfowler.com/bliki/BoiledCarrot.html
4 http://daniellakens.blogspot.ca/2016/01/power-analysis-for-default-bayesian-t.html Data rstats api
5 https://thinkinator.com/2016/01/12/r-users-will-now-inevitably-become-bayesians/ Statistics
6                               https://xianblog.wordpress.com/2016/01/14/precision-in-mcmc/ machine-learning
MCMC
>
N 6603:R 10:15:2
```

Print screen do Vim com R. Fonte.

Outros editores - VS Code

- [https://code.visualstudio.com/.](https://code.visualstudio.com/)
- É um editor muito usado por programadores de aplicações para linguagens JavaScript, Python, etc.
- Suporte para linguagem R com plugins.
- Tutorial de configuração para R aqui: Produção Animal com R.

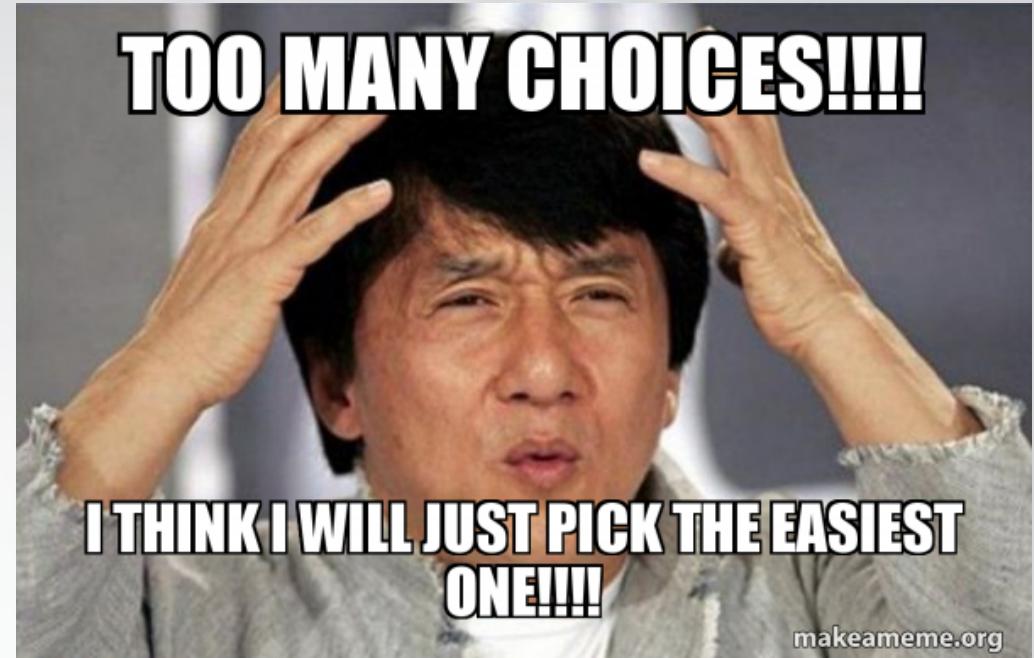
The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer:** Shows a tree view of files and folders. Visible items include: TANIA (with dist, Pespalm_trabalho2, producaoGAS, .jupyter_checkpoints, RData, History, chutesGompertz.ods, dadosGráfico.csv, dadosGráfico2.csv, Dendograma28-2.png, Dendograma28.png, Dendograma42.png, GAS.csv, GAS.xlsx, GASES.R, GASES.R~, GraficoGas.csv, Graficos.ipynb, GraficosGas.r, GraficosGas.r~, IdentidadeModelos.ipynb, Material e Métodos.docx, modelos.xlsx, modelos28dias.txt, modelos28diasconvergir..., and modelos28diasconvergir...); OUTLINE and TIMELINE.
- Code Editor:** Displays an R script named "GASES.R". The code includes library imports (lattice, car, manipulate, lattice, phia, fmsnisc, emmeans), data loading (dados <- read.csv), data transformation (dados <- transform(dados, BGP=factor(BGP), Colheita=factor(Colheita), Rep=factor(Rep))), and plotting (xyplot(Gases~tempo|Colheita, groups=BGP, auto.key=FALSE, type=c("smooth"), data=dados[,]), plot(Gases~tempo, subset(dados, BGP==144 & Colheita==28), main="PG PROPOLIS"))). It also defines a function "france" for fitting a model to the data.
- Terminal:** Shows the command "library(lattice)" followed by the response "R> library(lattice)".
- Status Bar:** Shows "Ln 2, Col 1", "Spaces: 7", "UTF-8", "LF", "R Language", and other icons.

Print screen do VS code com R. Fonte.

Outros editores

- ▶ Existem vários outros editores que podem ser habilitados para R.
 - ▶ Atom.
 - ▶ RGedit: Gedit + R.
 - ▶ RKWard IDE.
 - ▶ NppToR: R in Notepad++.
 - ▶ Etc.
- ▶ Mais editores de código:
 - ▶ Wikipedia - R programming language editors
 - ▶ Sciviews - Editors



Fonte.

Recomendações sobre editores

IDEs



Fonte.

- ▶ Recomendado para **iniciantes**.
- ▶ Vêm prontas para utilizar.
- ▶ Curva de apredizado fácil.
- ▶ Funcionalidades à plena vista.
- ▶ Pouca customização.
- ▶ São limitadas a linguagem.
- ▶ Nossa recomendação: **RStudio IDE**.

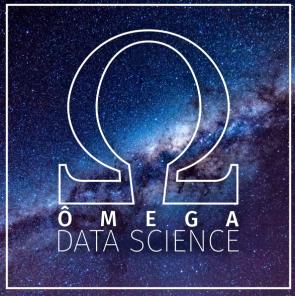
Editores



Fonte.

- ▶ Recomendado para experientes.
- ▶ E que precisam trabalhar com outras linguagens também.
- ▶ Requer habilitar o editor para a linguagem.
- ▶ Alto nível de customização.
- ▶ Pode-se trabalhar com várias linguagens e de forma similar.
- ▶ Nossa recomendação: **Doom-Emacs ou VS Code**.

Perguntas?



Primeiros passos com R

Instruções, área de trabalho, documentação e mais

Prof. Dr. **Walmes Zeviani**

2021-10-28

Instruções na linguagem R

O que é código e o que não é?

Usando o R a primeira vez



A sensação de abrir o R pela primeira vez.

- ▶ **CLI** - *Command Line Interface.*
- ▶ **REPL** - *Read, Eval, Print and Loop.*

Instruções na linguagem R

Modos de uso

► Modo REPL

- Script com instruções (receita).
- Instruções avaliadas no console (interpretador).
- Analista supervisoria o processo.
- Salva/duplica/modifica o script conforme necessidade.

► Modo Batch

- Script executa ser supervisão.
- Usado em ambientes de produção ou simulação computacional.

```
# Faz uma soma.  
2 + 2
```

```
## [1] 4
```

```
# Meu índice de massa corporal.  
83/1.85^2
```

```
## [1] 24.25128
```

```
# Quantos segundos tem um dia?  
24 * 60 * 60
```

```
## [1] 86400
```

Comentários no código

- Tudo que vem após # é comentário.
- Eles documentam o que o código faz.

Instruções e comentários

Faça comentários relevantes

- ▶ Evite comentários óbvios.
- ▶ Evite comentários ambiguos.

Instruções

- ▶ Podem ocupar uma única linha.
- ▶ Ocupar várias linhas.
- ▶ Uma linha ter várias instruções.

Recomendações

- ▶ Evite ultrapassar 72 ou 80 caracteres.
- ▶ Mantenha o código devidamente indentado (`ctrl + i`).
- ▶ Evite muitas instruções em uma linha.

```
# Uma única linha.  
2 + 2 + 7 + 5
```

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

```
# Em várias linhas.  
2 +  
  2 +  
  7 +  
  5
```

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

```
# Várias em uma linha.  
2 + 2; 7 + 5
```

```
## [1] 4
```

```
## [1] 12
```

?

Dúvidas, comentários, sugestões?

Área de trabalho

Onde estão as coisas que eu crio?

Área ou espaço de trabalho

Área de trabalho

- Ao fazer atribuições, são criados objetos na área de trabalho.
- Usa-se `<-` para atribuir algo para um objeto (`alt` + `-`)
- Objetos podem ser reusados para criar outros (essa é a intenção).
- Objetos podem ser sobrescritos.
- Objetos podem ser apagados.

```
peso <- 83      # Meu peso (kg).
altura <- 1.18 # Minha altura (m)
imc <- peso/altura^2
imc
## [1] 59.60931

# Lista objetos na área de trabalho.
ls()
## [1] "altura" "imc"     "peso"

# Apaga objetos.
rm(peso, altura)
ls()
## [1] "imc"
```

Espaços de trabalho

Onde o R procura por objetos?

- ▶ `ls()` mostra o conteúdo do `.GlobalEnv`, a área de trabalho.
- ▶ Mas existem objetos em outros ambientes ou espaços.
- ▶ Esses são os espaços dos pacotes.
- ▶ Quando não encontra no `.GlobalEnv`, ele vai para o próximo espaço de trabalho.
- ▶ `search()` retorna a lista de espaços de trabalho.
- ▶ Cada pacote tem um seu espaço (*namespace*).

```
# Mas `woman` não está no `GlobalEnv`.
# De onde veio?
woman

# Mostra o .GlobalEnv.
ls()

# Mostra os demais espaços.
search()

# Lista o conteúdo de um espaço.
ls("package:datasets")

# O que acontece se eu fizer?
woman <- c("Gertrude Mary Cox",
       "Florence Nightingale David")

woman           # Está no .GlobalEnv.
datasets::woman # Está no pacote.
```

?

Dúvidas, comentários, sugestões?

Acesso à documentação do R

Como aprender R sem sair do R

Documentação interna

A documentação do R

- ▶ Consiste de documentação de objetos e funções.
- ▶ Tutoriais chamados de vinhetas (*vignettes*).
- ▶ Existem funções específicas para a consulta destes.
- ▶ Pode-se procurar na web também.

```
# Duas formas iguais de chamar  
# a documentação.  
?woman  
help(woman)  
  
# Procura por ocorrências de `women`.  
help.search("woman")  
  
# Objetos que batem com o termo.  
apropos("tukey")  
  
# Exibe as vinhetas de um pacote.  
browseVignettes(package = "survival")  
  
# Procura pelo termo no  
# r-project.org.  
RSiteSearch("spider plot")
```

Campos da documentação

Cabeçalho

Indica o pacote.

Título

Título da função.

Description

Descrição do que o objeto é/faz.

Usage

Como usar ou fazer a chamada.

Arguments

Quais os argumentos formais da função.

Value

O que a função retorna.

Details

Detalhes adicionais de implementação.

Note

Notas adicionais sobre uso e afins.

See Also

Referências para documentação relacionada.

References

Referências bibliográficas.

Authors

Autores da função.

Examples

Exemplos de uso.

Guia de sobrevivência no R

1. ls()
2. apropos()
3. help()
4. help.search()
5. help.start()
6. RSiteSearch()
7. example()
8. demo()
9. browseVignettes()
10. vignette()
11. str()
12. args()
13. class()
14. methods()
15. find()



Kit de sobrevivência no R. Aprenda a usá-lo e tudo será mais fácil.

Arquivos da linguagem R

Customizando a experiência com a linguagem

Arquivos da linguagem R

Existem arquivos relacionados à linguagem R que servem para melhorar a experiência do usuário.

- ▶ **.Rhistory:**

Arquivo que contém as instruções enviadas pro console.

- ▶ **.RData:**

Arquivo binário que salva sua área de trabalho.

- ▶ Serve para restaurar a área de trabalho.
- ▶ Útil quando o processamento é demorado.

- ▶ **.Rproj:**

Arquivo que define configurações do projeto (para RStudio apenas).

- ▶ Eles são salvos no **diretório de trabalho**.
- ▶ Pode ser configurado em RStudio IDE > Tools > Global Options.

- ▶ **.Rprofile:**

Arquivo de configurações lido no início das seções.

- ▶ Carregar pacotes muito usados e configurar opções.
- ▶ Mensagem de boas-vindas e diagnóstico do sistema.
- ▶ Pode ser definido por projeto ou por usuário.

Diretório de trabalho

- ▶ Diretório de trabalho é local no sistema operacional para onde o R está apontando.
- ▶ Isto é, de onde ele lê e escreve arquivos por padrão.
- ▶ Você pode definir por comandos (recomendável) ou usando RStudio IDE > Session > Setting Working Directory.

```
# Mostra o atual diretório de trabalho.  
getwd()  
  
# Troca por outro endereço.  
setwd("~/Downloads")  
  
# Lista o conteúdo do diretório de trabalho.  
dir()
```

Pacotes

Instalação e gerenciamento

Instalação de pacotes

Pacotes

- ▶ Pacotes são coleções de funções e conjuntos de dados organizados e documentados.
- ▶ O pacote contém código R e eventualmente códigos de outras linguagens.
- ▶ Pacotes podem depender de *libs* do sistema operacional.

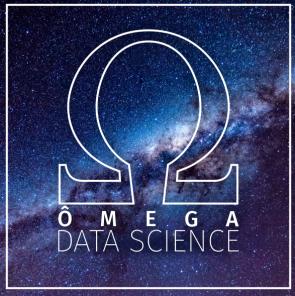
Formas de instalação

- ▶ Pacotes podem ser instalados de repositórios: CRAN, Bioconductor, MRAN, etc.
- ▶ De arquivos de instalação `*.tar.gz`.
- ▶ De repositórios Git: GitHub, GitLab, etc.
- ▶ Para mais, visite [r-packages-guide](#).

```
# Para instalar um pacote do repositório.  
install.package("tidyverse")  
  
# Para carregar o pacote e usá-lo.  
library(tidyverse)  
  
# Para ver o conteúdo dele.  
ls("package:tidyverse")  
  
# Documentação do pacote.  
help(package = "tidyverse")  
  
# Para ver onde foi instalado.  
system.file(package = "tidyverse")  
  
# Os caminhos para endereços de instalação.  
.libPaths()  
  
# Para remover o pacote da sessão.  
detach("package:tidyverse", unload = TRUE)  
  
# Funções relacionadas a pacotes.  
apropos("package")
```

?

Dúvidas, comentários, sugestões?



Aritmética básica com R

Operações matemáticas e lógicas

Prof. Dr. **Walmes Zeviani**

2021-10-28

Operações matemáticas e lógicas

R como uma calculadora científica

Operações matemáticas

As operações básicas

```
2 + 3      # Soma.  
2 - 3      # Subtração.  
2 * 3      # Multiplicação.  
2/3        # Divisão.  
2^3         # Potenciação.  
2^(1/3)     # Radiciação.  
10 %% 3    # Resto.  
10 %/% 3   # Parte inteira.
```

Trigonometrícias

```
sin(3) # Seno.  
cos(3) # Cosseno.  
tan(3) # Tangente.  
  
asin(0.5) # Arco seno.  
acos(0.5) # Arco cosseno.  
atan(0.5) # Arco tangente.
```

Logarítmo

```
exp(2)          # Exponencial neperiano.  
log(10)        # Neperiano.  
log10(10)      # Base 10.  
log2(10)       # Base 2.  
log(10, base = 5) # Base qualquer.
```

Estatísticas

```
y <- c(1.85, 1.78, 1.81, 1.69, 1.71)  
mean(y)      # Média.  
median(y)    # Mediana.  
sd(y)        # Desvio-padrão amostral.  
var(y)        # Variância amostral.  
max(y)       # Máximo.  
min(y)       # Mínimo.  
length(y)    # Número de valores.
```

Operadores lógicos

Comparações de valor

```
x <- 3
y <- 4

2 == 2 # Igualdade.
2 != 2 # Desigualdade.
x <= y # Outros operadores:
        # "<", ">", and ">=".

(2 < 5) & (7 >= 3) # Operador AND.
(2 < 5) | (7 >= 3) # Operador OR.
!(2 < 5)             # Operador NOT.

"a" == "b" # Compara strings.
"a" < "b" # Ordem alfanumérica.
"1" < "a"
```

Tipos especiais

- ▶ `NA`: para valores ausentes.
- ▶ `NULL`: para objetos vazios.
- ▶ `Inf` e `-Inf`: para infinitos.
- ▶ `NaN`: para resultados não razoavelmente definidos.

```
5 + NA           # O resultado é NA.
is.na(5 + NA)   # Verifica se é NA.

10 + NULL        # Retorna objeto vazio.
is.null(NULL)    # Verifica se é nulo.

5/0              # Infinito.
is.finite(5/0)   # Verifica se é finito.

0/0              # Valor indeterminado.
is.nan(0/0)      # Verifica se é not a number.
```

Cartões de referência do R

R Reference Card

R Reference Card · Statistics

R base

R base · PT

RStudio IDE

RStudio IDE · PT

?

Dúvidas, comentários, sugestões?

Manipulando vetores de dados

Como trabalhar com dados?

Vetores

- ▶ O vetor é uma estrutura que armazena vários valores.
- ▶ O tipo de valor é **homogêneo**, i.e. todos os elementos do mesmo tipo de valor.

```
v1 <- c(1, 5, 11, 33)      # Numérico.  
v2 <- c("hello", "world")  # De caracteres.  
v3 <- c(TRUE, TRUE, FALSE) # Lógico.  
  
# ATTENTION: Coerção para o tipo mais geral.  
v4 <- c(v1, v2, v3, "boo")
```

Importante

Regra da Reciclagem

- ▶ Vetores menores são reciclados até ter tamanho do maior para fazer a operação.

```
v1 <- c(2, 4, 6, 8, 10)  
v2 <- c(1, 3)  
length(v1)  
length(v2)
```

```
# Operações entre vetores.  
v1 * 10 # Vetor-escalar.  
v1/2    # Vetor-escalar.  
v1 + v2 # Vetor-vetor.
```

Tipos de valores e classes

Tipos de valor

Os tipos básicos são:

- ▶ Lógico: TRUE, FALSE.
- ▶ Inteiro: 0, 5, 10, 20, etc.
- ▶ Numérico (aka double, float): 3.14, 10.22, etc.
- ▶ Caractere (aka string): "Curitiba", "Campinas", etc.
- ▶ Fator: "Casado", "Solteiro", etc.
- ▶ Complexo: números complexos com parte imaginária.
- ▶ Datas: yyyy-mm-aa.
- ▶ Data tempo: yyyy-mm-aa hh:mm:ss.

Consulta do tipo de valor

```
# Funções que começam com `is.`.  
apropos("^is\\.")  
is.integer(1)  
is.numeric(1)  
  
is.integer(1L)  
is.numeric(1L)  
  
is.character("Curitiba")  
  
y <- factor(c("Solteiro", "Casado"))  
is.factor(y)  
is.character(y)  
  
is.logical(c(TRUE, FALSE))  
  
# Conversões de tipo.  
apropos("^as\\.")  
as.numeric("123")  
as.logical(c(0, 1))
```

Classe e atributos

- ▶ Algo importante no aprendizado de R é dominar a lógica de **classes e métodos**.
- ▶ As funções genéricas (métodos) atuam conforme a classe do objeto.
- ▶ Os atributos são metadados os objetos usados para várias situações.

```
# Um vetor numérico.  
x <- c(1, 2, 3)  
class(x)  
methods(class = "numeric")  
  
# Numérico mas com valores nomeados.  
notas <- c("João" = 7.8,  
          "Bianca" = 10,  
          "Eduarda" = 8.5)  
class(notas)  
attributes(notas)  
names(notas)  
  
# Uma tabela.  
class(women)  
attributes(women)
```

Sequências, repetições e números aleatórios

- Útil para aprender R, saber gerar tipos de objetos e preencher com valores.
- Vamos ver como gerar:
 - Sequências regulares: 1, 2, 3, 4, 5, e assim por diante.
 - Repetições: 1, 1, 1, 2, 2, 2, 3, 3, 3.
 - Amostras aleatórias e números aleatórios.

Sequências regulares.

1:7

```
seq(from = 1, to = 10, by = 2)
seq(from = 1, to = 20, length.out = 7)
seq(from = 1, by = 2, length.out = 7)
```

Repetições.

```
rep(0, 5)
rep(1:3, times = 2)
rep(1:3, each = 2)
```

Amostras aleatórias.

```
sample(1:20, size = 10,
       replace = FALSE)
sample(c("a", "b", "c"), size = 10,
       replace = TRUE)
```

Números aleatórios.

```
runif(n = 10, min = 0, max = 1)
rnorm(n = 10, mean = 1.80, sd = 0.1)
```

Seleção de elementos no vetor

Seleção por posição

- A seleção por posição/indexação considera a posição dos elementos.

```
# Numérico mas com valores nomeados.  
notas <- c("João" = 7.8,  
         "Bianca" = 10,  
         "Eduarda" = 8.5,  
         "Felipe" = 7.0,  
         "Márcia" = 6.5)
```

```
notas[1]      # A posição 1.  
notas[5]      # A posição 5.  
notas[1:2]    # Um intervalo.  
notas[c(1, 3)] # Um conjunto.  
notas[-1]     # Remove.
```

Seleção condicional

- A seleção condicional considera uma máscara lógica para selecionar.

```
mask <- notas > 7.0  
mask  
notas[mask]  
  
notas[notas > 9.0]
```

Seleção por nomes

- Usa o nome associado aos elementos.

```
# Seleciona valores pelo nome associado.  
notas["João"]  
notas[c("Márcia", "Eduarda")]
```

Seleção e modificação

Modificação do conteúdo

- ▶ Uma vez que valores podem ser selecionados, o conteúdo pode ser modificado.

```
# Atribui nota para um aluno.  
notas["João"] <- 0  
notas  
  
# Atribui nota "desconhecida" para aluno.  
notas["Felipe"] <- NA  
notas  
  
# Remove elemento do vetor.  
notas <- notas[-4]  
notas
```

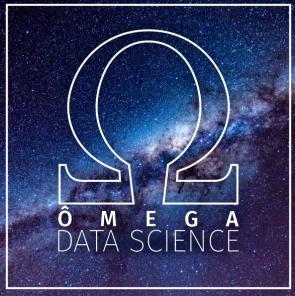
Adicionando mais valores

- ▶ O vetor pode ser incrementado ou concatenado com outros.

```
append(notas, value = c("Carlos" = 9.0))  
append(notas, value = c("Simone" = 7.2),  
      after = 0)  
  
novas_notas <- c(notas,  
                  c("Pedro" = 8.0,  
                    "Luana" = 8.3))  
novas_notas
```

?

Dúvidas, comentários, sugestões?



Conhecendo o Tidyverse

O framework de manipulação de dados do R

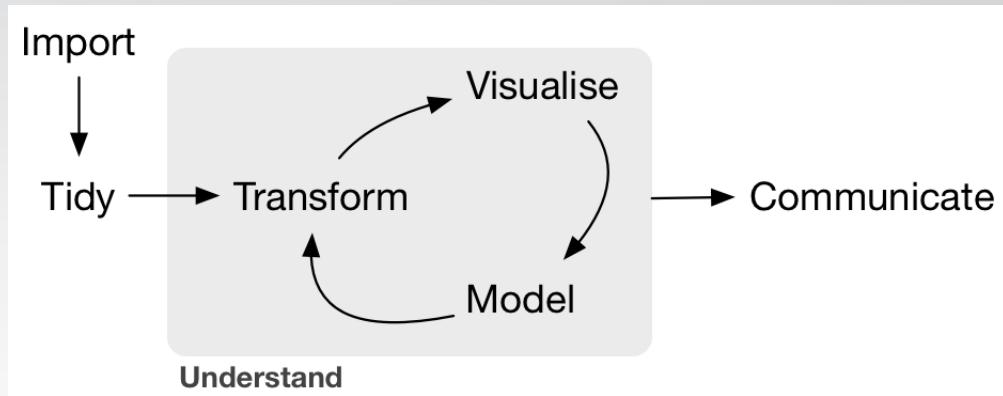
Prof. Dr. **Walmes Zeviani**

2021-10-28

Manipulação e visualização de dados

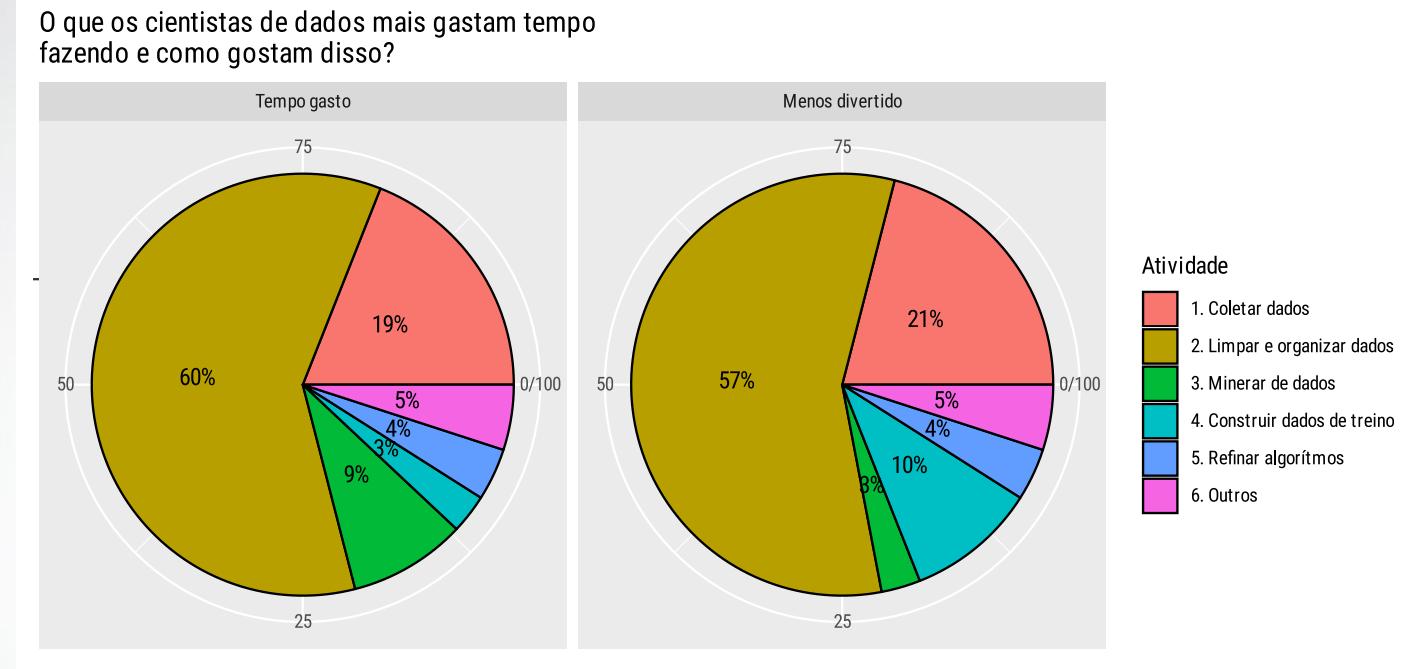
Manipulação e visualização de dados

- ▶ Manipular e visualizar dados (MVD) são atividades **obrigatórias** em Data Science (DS).
- ▶ A MVD **determina o sucesso** de uma série de etapas.
 - ▶ Entendimento dos dados.
 - ▶ Limpeza e conciliação de dados.
 - ▶ Engenharia de características.
 - ▶ Especificação de modelos.
 - ▶ Comunicação de resultados, etc.
- ▶ Fazer MVD de forma **eficiente** requer:
 - ▶ Conhecer o processo e suas etapas.
 - ▶ Dominar a tecnologia para execução.
- ▶ **Linguagens de programação** oferecem uma série de vantagens: reproduzível, extensível, escalonável, integrável, portável, etc.



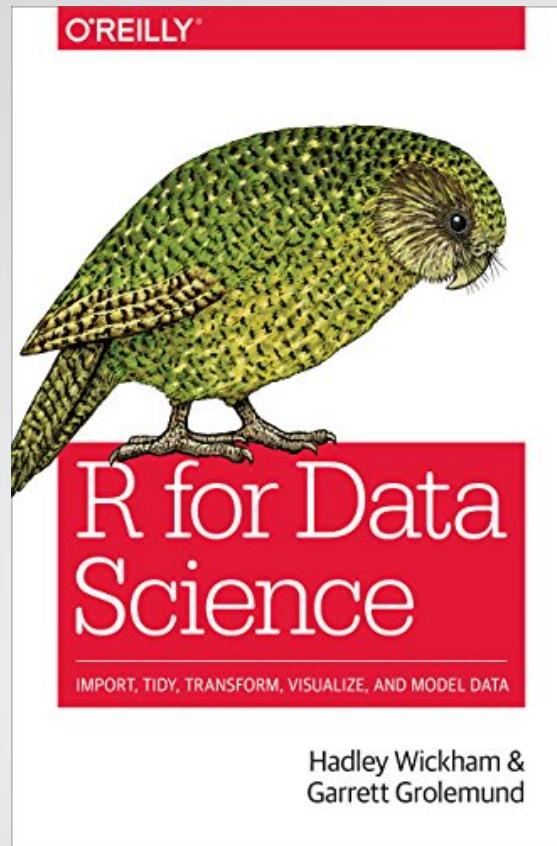
Ciclo de vida da ciência de dados. Fonte da imagem:
<https://bookdown.org/fjmcgrade/ismaykim/#intro-for-students>

O tempo gasto nas tarefas em DS

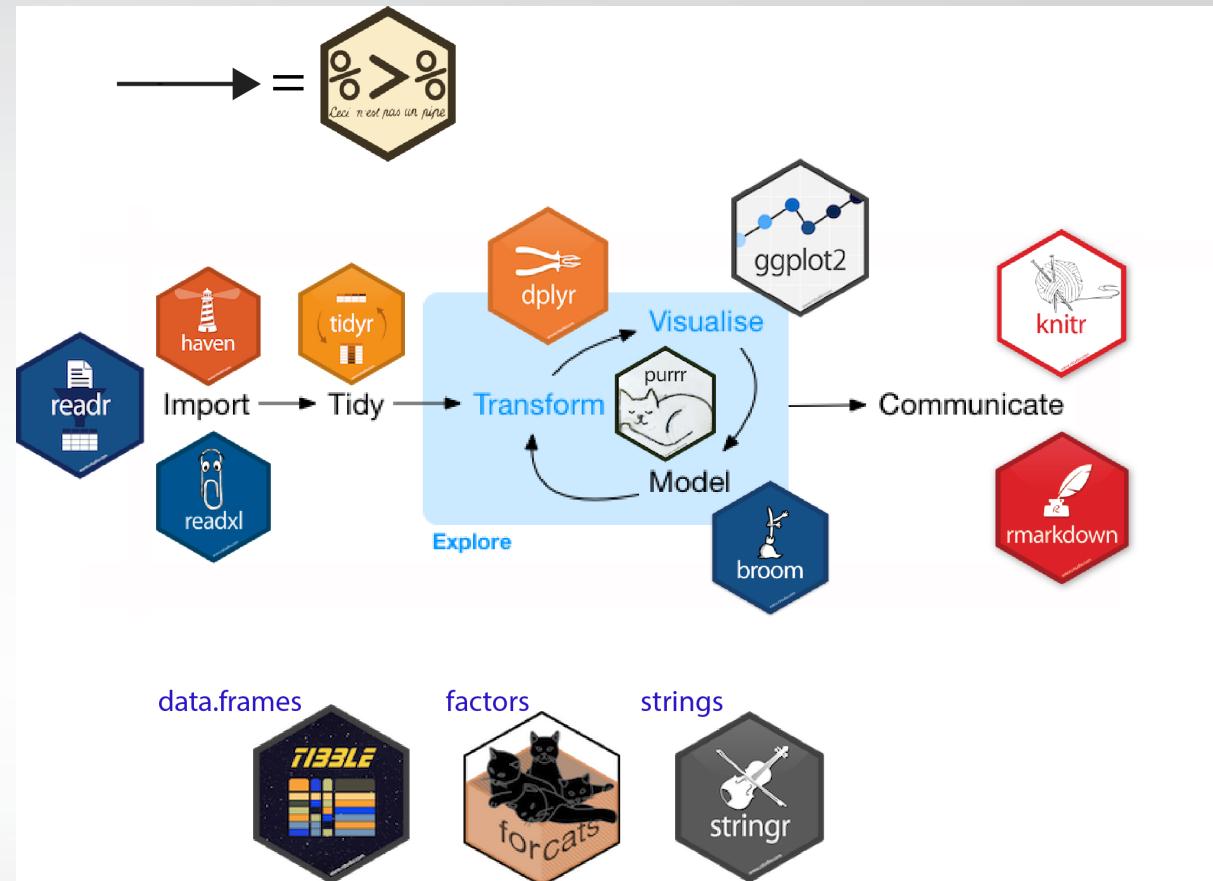


Tempo gasto e diversão em atividades. Fonte: Gil Press, 2016.

R for Data Science



R for Data Science, a principal referência sobre o emprego da linguagem R em ciência de dados.



Workflow de ciência de dados com o {tidyverse}. Fonte:
https://oliviergimenez.github.io/intro_tidyverse/#7

0 framework do {tidyverse}

O {tidyverse}

- ▶ Oferece uma **reimplementação e extensão** das funcionalidades para manipulação e visualização.
- ▶ É uma coleção **8 de pacotes R** que operam em harmonia.
- ▶ Eles foram planejados e construídos para trabalhar em conjunto.
- ▶ Possuem gramática, organização, filosofia e estruturas de dados mais clara.
- ▶ Maior facilidade de desenvolvimento de código e portabilidade.
- ▶ Outros pacotes acoplam muito bem com o {tidyverse}.
- ▶ Pacotes: <https://www.tidyverse.org/packages/>.
- ▶ R4DS: <https://r4ds.had.co.nz/>.
- ▶ Cookbook: <https://rstudio-education.github.io/tidyverse-cookbook/program.html>.

```
library(tidyverse)
tidyverse_packages()

## [ 1] "broom"          "cli"
## [ 3] "crayon"         "dbplyr"
## [ 5] "dplyr"          "dtplyr"
## [ 7] "forcats"        "googledrive"
## [ 9] "googlesheets4"  "ggplot2"
## [11] "haven"          "hms"
## [13] "httr"           "jsonlite"
## [15] "lubridate"      "magrittr"
## [17] "modelr"         "pillar"
## [19] "purrr"          "readr"
## [21] "readxl"         "reprex"
## [23] "rlang"          "rstudioapi"
## [25] "rvest"          "stringr"
## [27] "tibble"         "tidyverse"
## [29] "xml2"           "tidyverse"
```

Os pacotes do {tidyverse}



Pacotes que fazer parte do {tidyverse}.

A anatomia do {tidyverse}

{tibble}

- ▶ O `data.frame` é a estrutura nativa (primitiva) para representar tabelas de dados.
- ▶ O `tibble` é uma reimplementação da estrutura com melhorias.
 - ▶ Método `print` mais enxuto e informativo.
 - ▶ Mais consistente para seleção e modificação de conteúdo.
 - ▶ Mais fácil conversão de outros formatos para `tibble`.
 - ▶ Colunas/cédulas podem representar objetos mais complexos.
- ▶ Documentação:
 - ▶ <https://tibble.tidyverse.org/>.
 - ▶ <https://r4ds.had.co.nz/tibbles.html>.

{readr}

- ▶ O `readr` tem recursos para importação de dados retangulares na forma de texto pleno.
- ▶ 10x mais rápido que R básico e 1.2-2x mais lento de `data.table`.
- ▶ Leitura/escrita de dados tabulares: csv, tsv, fwf.
 - ▶ Funções de importação: `read_*`().
 - ▶ Funções de escrita: `write_*`().
 - ▶ Funções de *parsing*: `parse_*`.
- ▶ Recursos "inteligentes" que determinam tipo de variável, como importar campos de datas como datas!
- ▶ Documentação:
 - ▶ <https://readr.tidyverse.org/>.
 - ▶ <https://r4ds.had.co.nz/data-import.html>.

A anatomia do {tidyverse}

{tidyr}

- ▶ Suporte a criar de dados no formato `tidy` (tabular).
- ▶ Formato `tidy`:
 - ▶ Cada variável está em uma coluna.
 - ▶ Cada observação é uma linha.
 - ▶ Cada valor é uma cédula.
- ▶ Principais recursos:
 - ▶ Mudar disposição dos dados: long/empilhar
 \rightleftharpoons wide/esparramar.
 - ▶ Lidar com valores ausentes.
 - ▶ Partir/concatenar variáveis.
 - ▶ Aninhar/desaninhar listas.
- ▶ Documentação:
 - ▶ <https://tidyr.tidyverse.org/>.
 - ▶ <https://r4ds.had.co.nz/tidy-data.html>.

{dplyr}

- ▶ O `dplyr` é a **gramática** para manipulação de dados.
- ▶ Tem um conjunto **consistente** de verbos para atuar sobre tabelas.
 - ▶ Verbos: `mutate()`, `select()`, `filter()`, `arrange()`, `summarise()`, `slice()`, `rename()`, etc.
 - ▶ Sufixos: `_at()`, `_if()`, `_all()`, etc.
 - ▶ Extratificação: `group_by()` e `ungroup()`.
 - ▶ Junções: `inner_join()`, `full_join()`, `left_join()` e `right_join()`.
- ▶ Destaque são as operações de *split-apply-combine*.
- ▶ Documentação:
 - ▶ <https://dplyr.tidyverse.org/>.
 - ▶ <https://r4ds.had.co.nz/relational-data.html>.

A anatomia do {tidyverse}

{ggplot2}

- ▶ Criação de gráficos baseado no *The Grammar of Graphics* Wilkinson et. al (2013).
- ▶ Claro mapeamento das variáveis do BD em variáveis visuais e construção baseada em camadas.
- ▶ Documentação: <https://ggplot2.tidyverse.org/>.
- ▶ Wickham (2016): `ggplot2` - Elegant Graphics for Data Analysis.
- ▶ Teutonico (2015): `ggplot2` Essentials.

{purrr}

- ▶ O `purrr` fornece um conjunto **completo** e **consistente** para **programação funcional**.
- ▶ São uma sofisticação da *família apply*.
- ▶ Funções que aplicam funções em lote varrendo objetos: vetores, listas, etc.
- ▶ Várias função do tipo `map()` para cada tipo de input/output.
- ▶ Percorrem vetores, listas, colunas, linhas, etc.
- ▶ Permitem filtrar, concatenar, parear listas, etc.
- ▶ Além disso, permite:
 - ▶ Chamar funções de forma não tradicional.
 - ▶ Aplicar funções para tratamento de excessões.
 - ▶ Operar de forma a acumular e reduzir recursivamente.
 - ▶ Aninhar e aplinar objetos.
- ▶ Documentação: <https://purrr.tidyverse.org/>.

A anatomia do {tidyverse}

{forcats}

- ▶ Para manipulação de variáveis categóricas/fatores.
- ▶ As principais operações são:
 - ▶ Renomenar: manualmente, programaticamente (truncar, abreviar, etc.).
 - ▶ Reordenar: manualmente, por frequência, por alguma variável.
 - ▶ Aglutinar: combinar níveis menos frequentes, etc.
- ▶ Documentação:
 - ▶ <https://forcats.tidyverse.org/>.
 - ▶ <https://peerj.com/preprints/3163/>.

{stringr}

- ▶ Recursos coesos construídos para manipulação de *strings*.
- ▶ Feito sobre o pacote `stringi`.
- ▶ Praticamente tudo que envolva aplicação de expressões regulares.
 - ▶ Detectar.
 - ▶ Contar.
 - ▶ Partir.
 - ▶ Extrair.
 - ▶ Substituir.
 - ▶ etc.
- ▶ Documentação:
 - ▶ <https://stringr.tidyverse.org/>.

Adicionam funcionalidades

{lubridate} e {hms}

- ▶ Recursos para manipulação de dados *date*, *time* e *date-time*.
- ▶ Fácil decomposição de datas: dia, mês, semana, dia da semana, etc.
- ▶ Lida com fusos horários, horários de verão, etc.
- ▶ Extende para outras classes de dados baseados em *date-time*: duração, período, intervalos.
- ▶ Mas **não** é carregado junto com o `tidyverse`.

{magrittr}

- ▶ O operador permite expressar de forma mais direta as operações.
- ▶ É uma ideia inspirada no Shell e usada em várias linguagens.
- ▶ A lógica é bem simples:
 - ▶ `x %>% f` é o mesmo que `f(x)`.
 - ▶ `x %>% f(y)` é o mesmo que `f(x, y)`.
 - ▶ `x %>% f %>% g %>% h` é o mesmo que `h(g(f(x)))`.
- ▶ Existem outros operadores *pipe* para situações específicas.

Instalação o {tidyverse}

```
# Do CRAN.  
install.packages("tidyverse")  
  
# Do GitHub.  
# install.packages("devtools")  
devtools::install_github("hadley/tidyverse")  
  
# Atualizar caso já tenha instalado.  
tidyverse_update()
```

?

Dúvidas, comentários, sugestões?

Considerações finais

Sobre o {tidyverse}

- ▶ É um framework para manipulação de dados no R.
- ▶ Torna mais fácil, ágil e portável a escrita de código para manipulação de dados no R.
- ▶ Os pacotes foram feitos para trabalhar em harmonia.
- ▶ Não é a única forma de trabalhar com dados no R, mas é a mais popular em data science.

O que vem agora?

- ▶ Uma visão aprofundada de cada pacote do **tidyverse**.
- ▶ Exemplos didáticos seguidos de desafios práticos.
- ▶ *Happy coding.*

Referências

Teutonico, D. (2015). *ggplot2 Essentials*. Packt Publishing. ISBN: 9781785287558. URL: <https://books.google.com.br/books?id=zN0DCgAAQBAJ>.

Wickham, H. (2016). *ggplot2: Elegant Graphics for Data Analysis. Use R!* Springer International Publishing. ISBN: 9783319242774. URL: <https://books.google.com.br/books?id=XgFkDAAAQBAJ>.

Wilkinson, L., D. Wills, D. Rope, et al. (2013). *The Grammar of Graphics. Statistics and Computing*. Springer New York. ISBN: 9781475731002. URL: <https://books.google.com.br/books?id=ZiwLCAAAQBAJ>.



Importação de dados com Readr

Lendo dados de arquivos de texto

Prof. Dr. **Walmes Zeviani**

2021-10-28

Um overview do {readr}

- ▶ O processo de análise de dados começa com a importação dos dados para o ambiente de manipulação.
- ▶ Existem várias meios para armazenar dados.
 - ▶ Arquivos de texto pleno (tsv, txt, csv, etc).
 - ▶ Planilhas eletrônicas.
 - ▶ Bancos de dados relacionais.
 - ▶ Etc.
- ▶ O `readr` tem recursos para importação de dados retangulares na forma de texto pleno.
- ▶ Documentação:
 - ▶ <https://readr.tidyverse.org/>.
 - ▶ <https://r4ds.had.co.nz/data-import.html>.
 - ▶ <https://cran.r-project.org/package=readr>.

Cartão de referência de importação de dados com o {readr}

Clique no texto para abrir o arquivo

Funções para importação

```
library(tidyverse)

ls("package:readr") %>%
  str_subset("^read_")

## [1] "read_builtin"          "read_csv"
## [3] "read_csv_chunked"      "read_csv2"
## [5] "read_csv2_chunked"     "read_delim"
## [7] "read_delim_chunked"    "read_file"
## [9] "read_file_raw"         "read_fwf"
## [11] "read_lines"             "read_lines_chunked"
## [13] "read_lines_raw"         "read_lines_raw_chunked"
## [15] "read_log"               "read_rds"
## [17] "read_table"              "read_table2"
## [19] "read_tsv"               "read_tsv_chunked"
```

Principais argumentos para importação de dados

```
# Argumentos da `read_csv()`.  
args(read_csv)
```

```
## function (file, col_names = TRUE, col_types = NULL, locale = default_locale(),  
##           na = c("", "NA"), quoted_na = TRUE, quote = "\"", comment = "",  
##           trim_ws = TRUE, skip = 0, n_max = Inf, guess_max = min(1000,  
##                         n_max), progress = show_progress(), skip_empty_rows = TRUE)  
## NULL
```

```
# Argumentos da `locale()`.  
args(locale)
```

```
## function (date_names = "en", date_format = "%AD", time_format = "%AT",  
##           decimal_mark = ".", grouping_mark = ",", tz = "UTC", encoding = "UTF-8",  
##           asciify = FALSE)  
## NULL
```

As funções de escrita

```
ls("package:readr") %>%
  str_subset("^write_")

## [1] "write_csv"      "write_csv2"       "write_delim"
## [4] "write_excel_csv" "write_excel_csv2" "write_file"
## [7] "write_lines"     "write_rds"        "write_tsv"

args(write_delim)

## function (x, file, delim = " ", na = "NA", append = FALSE, col_names = !append,
##   quote_escape = "double", eol = "\n", path = deprecated())
## NULL
```

As funções de parsing

- ▶ Uma das maiores vantagens do `readr` é a maior flexibilidade para determinar o tipo de valor dos campos.
- ▶ As funções para o *parsing* (exame) são usadas para atribuir o tipo de valor apropriado durante a importação.
- ▶ Isso é economia de tempo.

```
ls("package:readr") %>%
  str_subset("^parse_")

## [1] "parse_character"  "parse_date"
## [3] "parse_datetime"   "parse_double"
## [5] "parse_factor"     "parse_guess"
## [7] "parse_integer"    "parse_logical"
## [9] "parse_number"      "parse_time"
## [11] "parse_vector"
```

```
# Conversão para data e data-tempo.
parse_date("2018/12/25")
parse_datetime("20181225")
parse_datetime("2018-12-25T12:10:00")

# Número com separador de milhar.
guess_parser("1,234,566")
parse_guess("1,234,566")

# Datas.
guess_parser(c("2010-10-10"))
parse_guess(c("2010-10-10"))
```

Importação de dados com `readr`

- ▶ O argumento obrigatório é o caminho para o arquivo.
- ▶ Argumentos opcionais existem pra um controle detalhado das opções de importação.

```
url <- "http://leg.ufpr.br/~walmes/data/anovareg.txt"

# Default.
tb <- read_tsv(file = url)

# Tipo de valores para cada variável.
tb <- read_tsv(file = url, col_types = "cicd")

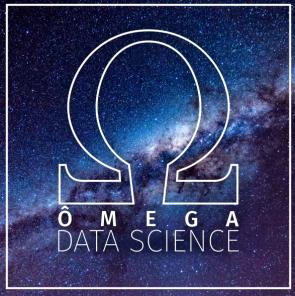
# Renomeia os campos.
tb <- read_tsv(file = url, col_names = c("clt", "ntr", "blc", "index"))
```

Exportação de dados com `readr`

```
# Funções para a escrita de tabelas em disco.  
ls("package:readr") %>%  
  str_subset("^write_")  
  
## [1] "write_csv"  
## [2] "write_csv2"  
## [3] "write_delim"  
## [4] "write_excel_csv"  
## [5] "write_excel_csv2"  
## [6] "write_file"  
## [7] "write_lines"  
## [8] "write_rds"  
## [9] "write_tsv"  
  
# Salvando tabela em disco.  
write_csv(iris, path = "iris_dataset.csv")
```

- ▶ Funções da `readr` produzem `tibbles`.
- ▶ Não importam string como fator, apenas por especificação.

```
# Mostra a estrutura assim como `str()`.  
attr(tb, "spec") <- NULL  
glimpse(tb)  
  
## #> #> #> #> #>  
## #> #> #> #> #>  
## #> #> #> #> #>  
## #> #> #> #> #>  
## #> #> #> #> #>
```



Arrumação de dados com Tidyr

Limpeza e organização

Prof. Dr. **Walmes Zeviani**

2021-10-28

Um overview do {tidyr}

- ▶ A maior parte das etapas de análise de dados assume que os dados estão arrumados:
 - ▶ Cada **coluna** é uma variável/atributo/campo.
 - ▶ Cada **linha** é uma observação/caso/instância/tupla.
 - ▶ Cada **cédula** é o registro de uma variável de uma observação.
- ▶ Situações que fogem a regra:
 - ▶ Disposição no **formato** longo ou amplo.
 - ▶ Colunas com valores **concatenados/separados**.
 - ▶ Registros com valores **ausentes**.
- ▶ O **tidyr** contém recursos para arrumação dos dados.
 - ▶ Mudança de disposição dos dados.
 - ▶ Substituição de missings.
 - ▶ Separação e união de campos.
- ▶ Documentação:
 - ▶ <https://tidyr.tidyverse.org/>.
 - ▶ <https://r4ds.had.co.nz/tidy-data.html>.
 - ▶ <https://cran.r-project.org/package=tidyr>

Cartão de referência de importação de dados com o {tidyverse}

Clique no texto para abrir o arquivo

Funções do pacote

```
library(tidyverse)
ls("package:tidyverse")

## [1] "%>%"
## [5] "build_wider_spec"
## [9] "construction"
##[13] "drop_na"
##[17] "expand"
##[21] "extract_"
##[25] "fish_encounters"
##[29] "hoist"
##[33] "nest_"
##[37] "num_range"
##[41] "pivot_longer_spec"
##[45] "relig_income"
##[49] "separate_rows"
##[53] "spread_"
##[57] "table3"
##[61] "tibble"
##[65] "uncount"
##[69] "unnest_"
##[73] "unnest_wider"
##[77] "world_bank_pop"
"as_tibble"
"chop"
"contains"
"drop_na_"
"expand_"
"extract_numeric"
"full_seq"
"last_col"
"nest_legacy"
"one_of"
"pivot_wider"
"replace_na"
"separate_rows_"
"starts_with"
"table4a"
"tidy_legacy"
"unite"
"unnest_auto"
"unpack"
"billboard"
"complete"
"crossing"
"ends_with"
"expand_grid"
"fill"
"gather"
"matches"
"nesting"
"pack"
"pivot_wider_spec"
"separate"
"smiths"
"table1"
"table4b"
"tribble"
"unite_"
"unnest_legacy"
"us_rent_income"
"build_longer_spec"
"complete_"
"crossing_"
"everything"
"extract"
"fill_"
"gather_"
"nest"
"nesting_"
"pivot_longer"
"population"
"separate_"
"spread"
"table2"
"table5"
"unchop"
"unnest"
"unnest_longer"
"who"
```

Empilhar variáveis

Situação comum quando:

- são feitas medidas ao longo do tempo.
- dados de painel e/ou questionário.

```
n <- 3
tb1 <- tibble("trat" = LETTERS[1:n],
             aval1 = rpois(n, 4),
             aval2 = rpois(n, 4),
             aval3 = rpois(n, 4))
tb1
```

```
## # A tibble: 3 × 4
##   trat  aval1  aval2  aval3
##   <chr> <int> <int> <int>
## 1 A        6      4      4
## 2 B        3      6      5
## 3 C        4      2      3
```

```
tb2 <- tb1 %>%
      pivot_longer(names_to = "aval",
                   values_to = "insetos",
                   cols = aval1:aval3)
tb2
```

```
## # A tibble: 9 × 3
##   trat  aval  insetos
##   <chr> <chr>  <int>
## 1 A     aval1     6
## 2 A     aval2     4
## 3 A     aval3     4
## 4 B     aval1     3
## 5 B     aval2     6
## 6 B     aval3     5
## 7 C     aval1     4
## 8 C     aval2     2
## 9 C     aval3     3
```

Desempilhar variável

- ▶ É a operação inversa de empilhar.
- ▶ Dados nessa disposição são menos comuns.

```
tb2 %>%
  pivot_wider(names_from = "aval",
              values_from = "insetos")

## # A tibble: 3 × 4
##   trat  aval1  aval2  aval3
##   <chr> <int> <int> <int>
## 1 A        6     4     4
## 2 B        3     6     5
## 3 C        4     2     3
```

Separar variável

- Muito comum quando um campo de texto é a união de várias informações.
- Ex: datas, horas, endereços, etc.

```
tb <- tibble(veiculo = c("Celta", "Gol", "Uno"),
             ano_mod = c("2011/2012", "2012/2012", "2015/2016"),
             local = c("Curitiba-PR", "Santos-SP", "Viçosa-MG"))

tb %>%
  separate(col = "ano_mod",
          into = c("ano", "modelo"),
          sep = "/",
          convert = TRUE) %>%
  separate(col = "local",
          into = c("cidade", "estado"),
          sep = "-")

## # A tibble: 3 × 5
##   veiculo  ano modelo cidade estado
##   <chr>    <int>  <int> <chr>   <chr>
## 1 Celta     2011    2012 Curitiba PR
## 2 Gol       2012    2012 Santos   SP
## 3 Uno       2015    2016 Viçosa   MG
```

Unir variáveis

- Quando vários campos precisam ser combinados para gerar uma informação.
- Ex: datas, horas, endereços, nomes.

```
tb <- tibble(dia = c(1, 5, 23, 16),
             mes = c(3, 6, 2, 9),
             ano = 2018)
tb %>%
  unite(col = "data", ano, mes, dia, sep = "-", remove = FALSE) %>%
  mutate(data = parse_date(data, format = "%Y-%m-%d"))

## # A tibble: 4 × 4
##   data      dia   mes   ano
##   <date>    <dbl> <dbl> <dbl>
## 1 2018-03-01     1     3  2018
## 2 2018-06-05     5     6  2018
## 3 2018-02-23    23     2  2018
## 4 2018-09-16    16     9  2018
```

Manuseio de valores ausentes

```
tb <- tibble(jogador = 1:5,
             jogos = c(0, 1, 3, 1, 2),
             gols = c(NA, 0, 0, 2, 1),
             faltas = c(NA, 1, 1, 0, 0))

# tb %>%
#     drop_na()

tb %>%
    replace_na(list(gols = 0, faltas = 0))

## # A tibble: 5 × 4
##   jogador jogos  gols faltas
##     <int> <dbl> <dbl> <dbl>
## 1       1      0      0      0
## 2       2      1      0      1
## 3       3      3      0      1
## 4       4      1      2      0
## 5       5      2      1      0
```

Exercícios para usar o tidyverse

- ▶ Futebol.
 1. Ler os dados em http://leg.ufpr.br/~walmes/data/euro_football_players.txt.
 2. Substituir os missings em `goal`, `red` e `yel` por 0.
- ▶ Avaliação de veículos.
 1. Ler os dados em http://leg.ufpr.br/~walmes/data/aval_carros_nota.txt.
 2. Desempilhar na variável `item` os valores de `nota`.
- ▶ Carros à venda.
 1. Ler os dados em http://leg.ufpr.br/~walmes/data/duster_venda_260314.txt.
 2. Separar os campos ano e modelo na variável `ano`. Ex: `2012/2013`.
 3. Substituir o `NA` em `km` percorrido pela média de km percorrido: `mean(..., na.rm = TRUE)`.



Transformação de dados com Dplyr

Fazendo operações de transformação e análise de dados

Prof. Dr. **Walmes Zeviani**

2021-10-28

Um overview do {dplyr}

- ▶ Depois dos dados arrumados, é a hora começar conhecê-los!
- ▶ Começa a fase de **análise exploratória de dados** (AED).
- ▶ Os dados são explorados para:
 - ▶ Conhecer as (propriedades das) variáveis.
 - ▶ Determinar medidas descritivas.
 - ▶ Comparar grupos.
 - ▶ Quantificar relações entre variáveis.
 - ▶ Extrair padrões.
 - ▶ Detectar ameaças e corrigir problemas.
- ▶ AED envolve inúmeras operações.
- ▶ É preciso conhecê-las e ser criativo para aplicar da melhor forma.

Detalhes do dplyr

- ▶ O `dplyr` é a **gramática** para manipulação de dados.
- ▶ Tem um conjunto **consistente** de verbos para atuar sobre tabelas.
 - ▶ Verbos: `mutate()`, `select()`, `filter()`, `arrange()`, `summarise()`, `slice()`, `rename()`, etc.
 - ▶ Sufixos: `_at()`, `_if()`, `_all()`, etc.
 - ▶ Agrupamento: `group_by()` e `ungroup()`.
 - ▶ Junções: `inner_join()`, `full_join()`, `left_join()` e `right_join()`.
 - ▶ Funções resumo: `n()`, `n_distinct()`, `first()`, `last()`, `nth()`, etc.
 - ▶ E muito mais no cartão de referência: <https://github.com/rstudio/cheatsheets/raw/master/data-transformation.pdf>.
- ▶ Documentação:
 - ▶ <https://dplyr.tidyverse.org/>.
 - ▶ <https://r4ds.had.co.nz relational-data.html>.
 - ▶ <https://cran.r-project.org/package=dplyr>

Cartão de referência de importação de dados com o {dplyr}

Clique no texto para abrir o arquivo

Criação de um data.frame

	matrícula	nome	curso	prova1	prova2	prova3	faltas
1	256	João	Mat	80	90	80	4
2	487	Vanessa	Mat	75	75	75	4
3	965	Tiago	Est	95	80	75	0
4	125	Luana	Est	70	85	50	8
5	458	Gisele	Est	45	50		16
6	874	Pedro	Mat	55	75	90	0
7	963	André	Est	30		30	20

Uma tabela com dados fictícios sobre alunos e seus desempenhos.

Criação de um tibble

Criação por colunas

```
library(tidyverse)
```

```
# Tabela com alunos do curso de
# Matemática e de Estatística.
df1 <- tibble(
  matricula = c(256, 487, 965,
                125, 458, 874, 963),
  nome = c("João", "Vanessa", "Tiago",
          "Luana", "Gisele", "Pedro",
          "André"),
  curso = c("Mat", "Mat", "Est", "Est",
            "Est", "Mat", "Est"),
  prova1 = c(80, 75, 95, 70, 45, 55, 30),
  prova2 = c(90, 75, 80, 85, 50, 75, NA),
  prova3 = c(80, 75, 75, 50, NA, 90, 30),
  faltas = c(4, 4, 0, 8, 16, 0, 20))
```

```
df1
```

```
## # A tibble: 7 × 7
##   matricula nome    curso prova1 prova2
##       <dbl> <chr>   <chr>   <dbl>   <dbl>
## 1      256 João    Mat      80      90
## 2      487 Vanessa Mat      75      75
## 3      965 Tiago   Est      95      80
## 4      125 Luana   Est      70      85
## 5      458 Gisele  Est      45      50
## 6      874 Pedro   Mat      55      75
## 7      963 André   Est      30      NA
## # ... with 2 more variables:
## #   prova3 <dbl>, faltas <dbl>
```

Criação de um tibble

Criação por colunas

```
# Tabela com alunos de Engenharia.  
df2 <- tibble(  
  matricula = c(505, 658, 713),  
  nome = c("Bia", "Carlos", "Cris"),  
  curso = c("Eng", "Eng", "Eng"),  
  prova1 = c(65, 75, 75),  
  prova2 = c(85, 80, 90),  
  faltas = c(0, 0, 2))
```

```
df2
```

```
## # A tibble: 3 × 6  
##   matricula nome  curso prova1 prova2  
##       <dbl> <chr> <chr>    <dbl>    <dbl>  
## 1      505  Bia   Eng        65      85  
## 2      658  Carlos Eng        75      80  
## 3      713  Cris   Eng        75      90  
## # ... with 1 more variable: faltas <dbl>
```

Criação de um tibble

Criação por linhas

```
# Informações de cadastro dos alunos  
# em outra base de dados.  
df_extra <- tribble(  
  ~mat,      ~nome, ~idade, ~bolsista,  
  256, 'João' , 18,      "S",  
  965, 'Tiago' , 18,      "N",  
  285, 'Tiago' , 22,      "N",  
  125, 'Luana' , 21,      "S",  
  874, 'Pedro' , 19,      "N",  
  321, 'Mia'   , 18,      "N",  
  669, 'Luana' , 19,      "S",  
  967, 'André' , 20,      "N",  
)
```

```
df_extra
```

```
## # A tibble: 8 × 4  
##   mat nome  idade bolsista  
##   <dbl> <chr> <dbl> <chr>  
## 1 256 João    18 S  
## 2 965 Tiago   18 N  
## 3 285 Tiago   22 N  
## 4 125 Luana   21 S  
## 5 874 Pedro   19 N  
## 6 321 Mia     18 N  
## 7 669 Luana   19 S  
## 8 967 André   20 N
```

Ordenação

The diagram illustrates the sorting of student records in two tables. It shows two initial tables on the left, followed by arrows indicating the sorting process, and two final sorted tables on the right.

Initial Table 1:

	mat.	nome	curso	p1	p2	p3	fl
1	256	João	Mat	80	90	80	4
2	487	Vanessa	Mat	75	75	75	4
3	965	Tiago	Est	95	80	75	0
4	125	Luana	Est	70	85	50	8
5	458	Gisele	Est	45	50		16
6	874	Pedro	Mat	55	75	90	0
7	963	André	Est	30	30	20	

Initial Table 2:

	mat.	nome	curso	p1	p2	p3	fl
4	125	Luana	Est	70	85	50	8
1	256	João	Mat	80	90	80	4
5	458	Gisele	Est	45	50		16
2	487	Vanessa	Mat	75	75	75	4
6	874	Pedro	Mat	55	75	90	0
7	963	André	Est	30	30	20	
3	965	Tiago	Est	95	80	75	0

Sorting Process:

The first sorting process (indicated by a single downward arrow) sorts the records by the 'mat' column. The result is shown in the second table on the right.

Final Table 1 (After First Sort):

	mat.	nome	curso	p1	p2	p3	fl
1	256	João	Mat	80	90	80	4
2	487	Vanessa	Mat	75	75	75	4
3	965	Tiago	Est	95	80	75	0
4	125	Luana	Est	70	85	50	8
5	458	Gisele	Est	45	50		16
6	874	Pedro	Mat	55	75	90	0
7	963	André	Est	30	30	20	

Sorting Process:

The second sorting process (indicated by two downward arrows) sorts the records by the 'nome' column. The result is shown in the third table on the right.

Final Table 2 (After Second Sort):

	mat.	nome	curso	p1	p2	p3	fl
7	963	André	Est	30	30	20	
5	458	Gisele	Est	45	50		16
4	125	Luana	Est	70	85	50	8
3	965	Tiago	Est	95	80	75	0
6	874	Pedro	Mat	55	75	90	0
2	487	Vanessa	Mat	75	75	75	4
1	256	João	Mat	80	90	80	4

Ordenação dos registros de uma tabela.

Ordenação

Por uma variável

```
df1 %>%  
  arrange(matricula)  
  
## # A tibble: 7 × 7  
##   matricula nome  curso prova1 prova2  
##       <dbl> <chr> <chr>  <dbl>  <dbl>  
## 1       125 Luana  Est     70     85  
## 2       256 João   Mat     80     90  
## 3       458 Gisele Est     45     50  
## 4       487 Vanessa Mat     75     75  
## 5       874 Pedro  Mat     55     75  
## 6       963 André  Est     30     NA  
## 7       965 Tiago  Est     95     80  
## # ... with 2 more variables:  
## #   prova3 <dbl>, faltas <dbl>
```

Por duas ou mais variáveis

```
df1 %>%  
  arrange(curso, desc(prova1))  
  
## # A tibble: 7 × 7  
##   matricula nome  curso prova1 prova2  
##       <dbl> <chr> <chr>  <dbl>  <dbl>  
## 1       965 Tiago  Est     95     80  
## 2       125 Luana  Est     70     85  
## 3       458 Gisele Est     45     50  
## 4       963 André  Est     30     NA  
## 5       256 João   Mat     80     90  
## 6       487 Vanessa Mat     75     75  
## 7       874 Pedro  Mat     55     75  
## # ... with 2 more variables:  
## #   prova3 <dbl>, faltas <dbl>
```

`desc()`: ordenação de descente da variável.

Seleção das variáveis

Seleção pelos nomes

```
df1 %>%  
  select(nome, prova1, prova2, prova3)
```

```
## # A tibble: 7 × 4  
##   nome    prova1 prova2 prova3  
##   <chr>    <dbl>   <dbl>   <dbl>  
## 1 João      80     90     80  
## 2 Vanessa   75     75     75  
## 3 Tiago     95     80     75  
## 4 Luana     70     85     50  
## 5 Gisele    45     50     NA  
## 6 Pedro     55     75     90  
## 7 André     30     NA     30
```

```
df1 %>% select(c("nome", "prova1"))  
df1 %>% select(-nome, -faltas)  
df1 %>% select(prova1:prova3)
```

Seleção pela posição

```
df1 %>%  
  select(1:3)
```

```
## # A tibble: 7 × 3  
##   matricula nome   curso  
##   <dbl>     <chr>  <chr>  
## 1 256       João   Mat  
## 2 487       Vanessa Mat  
## 3 965       Tiago  Est  
## 4 125       Luana  Est  
## 5 458       Gisele Est  
## 6 874       Pedro  Mat  
## 7 963       André  Est
```

```
df1 %>% select(1, 4)  
df1 %>% select(-1, -4)
```

Seleção das variáveis

Seleção de variáveis por condição

```
df1 %>%  
  select_if(is.numeric)  
  
## # A tibble: 7 × 5  
##   matricula prova1 prova2 prova3 faltas  
##   <dbl>     <dbl>    <dbl>    <dbl>    <dbl>  
## 1      256      80      90      80      4  
## 2      487      75      75      75      4  
## 3      965      95      80      75      0  
## 4      125      70      85      50      8  
## 5      458      45      50      NA     16  
## 6      874      55      75      90      0  
## 7      963      30      NA     30     20
```

```
df1 %>%  
  select_if(negate(is.numeric))
```

Seleção por expressão regular

```
df1 %>%  
  select(matches("^prova"))  
  
## # A tibble: 7 × 3  
##   prova1 prova2 prova3  
##   <dbl>    <dbl>    <dbl>  
## 1      80      90      80  
## 2      75      75      75  
## 3      95      80      75  
## 4      70      85      50  
## 5      45      50      NA  
## 6      55      75      90  
## 7      30      NA     30
```

```
df1 %>%  
  select(matches("\d$"))  
df1 %>%  
  select(matches("^.{6}$"))
```

Filtros

The diagram illustrates the process of filtering data from a primary table through three intermediate steps:

- curso igual a Est**: The first filter selects rows where the 'curso' column is 'Est'. This results in a table with 4 rows: 3, 4, 5, and 7.
- condição igual a Apr**: The second filter selects rows where the 'condição' column is 'Apr'. This results in a table with 3 rows: 1, 2, and 6.
- média menor que 70**: The third filter selects rows where the 'média' column is less than 70. This results in a table with 3 rows: 4, 5, and 7.

The final result table contains the intersection of all filters, which is the row for student 7 (André).

nome	curso	fl	média	condição
1 João	Mat	4	83,3	Apr
2 Vanessa	Mat	4	75,0	Apr
3 Tiago	Est	0	83,3	Apr
4 Luana	Est	8	68,3	Exa
5 Gisele	Est	16	31,7	Rep
6 Pedro	Mat	0	73,3	Apr
7 André	Est	20	20,0	Rep

nome	curso	fl	média	condição
3 Tiago	Est	0	83,3	Apr
4 Luana	Est	8	68,3	Exa
5 Gisele	Est	16	31,7	Rep
7 André	Est	20	20,0	Rep

nome	curso	fl	média	condição
1 João	Mat	4	83,3	Apr
2 Vanessa	Mat	4	75,0	Apr
6 Pedro	Mat	0	73,3	Apr

nome	curso	fl	média	condição
4 Luana	Est	8	68,3	Exa
5 Gisele	Est	16	31,7	Rep
7 André	Est	20	20,0	Rep

Filtro dos registros de uma tabela.

Filtros

Usando uma variável

```
df1 %>%  
  filter(curso == "Est")  
  
## # A tibble: 4 × 7  
##   matricula nome  curso prova1 prova2  
##       <dbl> <chr> <chr>  <dbl>  <dbl>  
## 1       965 Tiago Est      95     80  
## 2       125 Luana Est      70     85  
## 3       458 Gisele Est     45     50  
## 4       963 André Est     30     NA  
## # ... with 2 more variables:  
## #   prova3 <dbl>, faltas <dbl>
```

```
df1 %>%  
  filter(faltas == 0)  
df1 %>%  
  filter(faltas != 0)  
df1 %>%  
  filter(faltas %in% c("Aline", "Vanessa"))
```

Usando duas ou mais

```
df1 %>%  
  select(curso, prova1:prova3) %>%  
  filter(curso == "Est",  
         (prova1 + prova2 + prova3) > 180)  
  
## # A tibble: 2 × 4  
##   curso prova1 prova2 prova3  
##   <chr>  <dbl>  <dbl>  <dbl>  
## 1 Est      95     80     75  
## 2 Est      70     85     50
```

Transformações

The diagram illustrates two data transformations on a student database:

Top Transformation: An arrow labeled "criar" points from the original table to a modified table. The original table has columns nome, curso, p1, p2, p3, and fl. The modified table adds columns média and condição. The fl column is highlighted in yellow, and the new columns are highlighted in green.

	nome	curso	p1	p2	p3	fl	média	condição
1	João	Mat	80	90	80	4	83,3	Apr
2	Vanessa	Mat	75	75	75	4	75,0	Apr
3	Tiago	Est	95	80	75	0	83,3	Apr
4	Luana	Est	70	85	50	8	68,3	Exa
5	Gisele	Est	45	50		16	31,7	Rep
6	Pedro	Mat	55	75	90	0	73,3	Apr
7	André	Est	30		30	20	20,0	Rep

Bottom Transformation: An arrow labeled "deletar" points from the original table to a simplified table. The original table has columns nome, curso, p1, p2, p3, and fl. The modified table only includes columns nome, p1, p2, and p3.

	nome	p1	p2	p3
1	João	80	90	80
2	Vanessa	75	75	75
3	Tiago	95	80	75
4	Luana	70	85	50
5	Gisele	45	50	
6	Pedro	55	75	90
7	André	30		30

Criação e deleção de variáveis em uma tabela.

Transformações

As operações podem modificar a tabela com a:

1. **Criação** de novas variáveis.
2. **Remoção** de variáveis.
3. **Transformação** de variáveis.

As operações de criação/transformação podem ser:

1. **Matemáticas**: aritméticas, potência, logarítmicas, trigonométricas, etc.
2. **Compartimentação** (*binning*): agrupar em classes.
3. **Conversão** de tipo de valor: i.e. de `int` → `str`.
4. **Substituição**: i.e. preencher um valor ausente.

As transformações podem ser:

1. Uma → uma:

$$y = \log(x).$$

2. Várias → uma:

$$z = x/y^2.$$

3. Várias → várias:

$$y_1, \dots, y_k = f_1(x_1, \dots, x_m), \dots, f_k(x_1, \dots, x_m).$$

Transformações

Criação de variável

```
df1 %>%
  mutate(media = (prova1 + prova2 + prova3/3)) %>%
  select(nome, curso, media)
```

```
## # A tibble: 7 × 3
##   nome   curso  media
##   <chr>  <chr> <dbl>
## 1 João    Mat    197.
## 2 Vanessa Mat    175
## 3 Tiago   Est    200
## 4 Luana   Est    172.
## 5 Gisele  Est    NA
## 6 Pedro   Mat    160
## 7 André   Est    NA
```

Transformações em várias variáveis

```
df1 %>%
  mutate_at(vars(prova1:prova3),
            ~replace_na(., 0)) %>%
  select(prova1:prova3)
```

```
## # A tibble: 7 × 3
##   prova1 prova2 prova3
##   <dbl>  <dbl>  <dbl>
## 1     80     90     80
## 2     75     75     75
## 3     95     80     75
## 4     70     85     50
## 5     45     50      0
## 6     55     75     90
## 7     30      0     30
```

Transformações

Transformações dado uma condição

```
# Passa para caixa alta.  
df1 %>%  
  mutate_if(is.character, str_to_upper)  
  
## # A tibble: 7 × 7  
##   matricula nome    curso prova1 prova2  
##       <dbl> <chr>   <chr>  <dbl>  <dbl>  
## 1       256 JOÃO    MAT      80     90  
## 2       487 VANESSA MAT      75     75  
## 3       965 TIAGO   EST      95     80  
## 4       125 LUANA   EST      70     85  
## 5       458 GISELE  EST      45     50  
## 6       874 PEDRO   MAT      55     75  
## 7       963 ANDRÉ  EST      30     NA  
## # ... with 2 more variables:  
## #   prova3 <dbl>, faltas <dbl>  
  
df1 %>% mutate_if(is.numeric, sqrt)  
df1 %>% mutate_if(is.numeric, log)  
df1 %>% mutate_if(is.character, as.factor)  
df1 %>% mutate_if(is.numeric, as.integer)
```

Transformações dado uma expressão regular

```
# Divide a nota por 10.  
df1 %>%  
  mutate_at(vars(starts_with("prova")),  
           ~./10)  
  
## # A tibble: 7 × 7  
##   matricula nome    curso prova1 prova2  
##       <dbl> <chr>   <chr>  <dbl>  <dbl>  
## 1       256 João    Mat      8       9  
## 2       487 Vanessa Mat     7.5    7.5  
## 3       965 Tiago   Est     9.5    8  
## 4       125 Luana   Est      7     8.5  
## 5       458 Gisele  Est     4.5    5  
## 6       874 Pedro   Mat     5.5    7.5  
## 7       963 André   Est      3     NA  
## # ... with 2 more variables:  
## #   prova3 <dbl>, faltas <dbl>
```

Transformações

Cortar valores em classe

```
# Intervalos para corte e rótulos.
inter <- c(-Inf, 40, 70, Inf)
condi <- c("reprovado", "exame", "aprovado")

tb_final <- df1 %>%
  mutate_at(vars(matches("prova")),
            ~replace_na(., 0)) %>%
  mutate(media = (prova1 + prova2 + prova3)/3,
         result= cut(media,
                     breaks = inter,
                     labels = condi,
                     right = FALSE,
                     include.lowest = TRUE))
```

```
tb_final %>%
  select(nome, curso,
         media, result)

## # A tibble: 7 × 4
##   nome    curso  media result
##   <chr>   <chr> <dbl> <fct>
## 1 João    Mat    83.3 aprovado
## 2 Vanessa Mat    75     aprovado
## 3 Tiago   Est    83.3 aprovado
## 4 Luana   Est    68.3 exame
## 5 Gisele  Est    31.7 reprovado
## 6 Pedro   Mat    73.3 aprovado
## 7 André   Est    20     reprovado
```

Transformações

Substituição de valores ausentes

```
df1 %>%
  mutate_at(vars(matches("prova")),
            ~replace_na(., 0)) %>%
  mutate(faltas =
         replace_na(faltas, 0)) %>%
  select(prova1:prova3, faltas)
```

```
## # A tibble: 7 × 4
##   prova1 prova2 prova3 faltas
##     <dbl>   <dbl>   <dbl>   <dbl>
## 1     80     90     80      4
## 2     75     75     75      4
## 3     95     80     75      0
## 4     70     85     50      8
## 5     45     50      0     16
## 6     55     75     90      0
## 7     30      0     30     20
```

```
df1 %>%
  replace_na(replace =
              list(prova1 = 0,
                    prova2 = 0,
                    prova3 = 0,
                    faltas = 60)) %>%
  select(prova1:prova3, faltas)
```

```
## # A tibble: 7 × 4
##   prova1 prova2 prova3 faltas
##     <dbl>   <dbl>   <dbl>   <dbl>
## 1     80     90     80      4
## 2     75     75     75      4
## 3     95     80     75      0
## 4     70     85     50      8
## 5     45     50      0     16
## 6     55     75     90      0
## 7     30      0     30     20
```

Medidas resumo

- ▶ Operações para determinar estatísticas descritivas.
 - ▶ Soma, média, mediana, quartis, quantis, etc.
 - ▶ Variância, desvio-padrão, amplitude, desvio absoluto da mediana, coeficiente de variação, etc.
 - ▶ Número de níveis distintos, frequências absolutas/relativas, etc.
- ▶ Elas podem ser marginais ou considerar a estratificação conforme uma ou mais variáveis categóricas.
- ▶ Podem ser aplicadas em todas as variáveis de um mesmo tipo (homegêneo).

nome	curso	p1	p2	p3	f _l	média	desvio
1 João	Mat	80	90	80	4	83,3	5,8
2 Vanessa	Mat	75	75	75	4	75,0	0,0
3 Tiago	Est	95	80	75	0	83,3	10,4
4 Luana	Est	70	85	50	8	68,3	17,6
5 Gisele	Est	45	50		16	47,5	3,5
6 Pedro	Mat	55	75	90	0	73,3	17,6
7 André	Est	30		30	20	30,0	0,0

média	p1	p2	p3
64,3	64,3	75,8	66,7
desvio	22,3	13,9	22,3

Cálculo de medidas resumo.

Medidas resumo

Uma estatística

```
# Medidas resumo de uma estatística.  
df1 %>%  
  summarise(sum(prova1),  
            mean(prova1),  
            max(prova1),  
            min(prova1),  
            median(prova1),  
            sd(prova1),  
            var(prova1),  
            length(prova1)) %>%  
  t()
```

```
## [1]  
## sum(prova1)    450.00000  
## mean(prova1)   64.28571  
## max(prova1)    95.00000  
## min(prova1)    30.00000  
## median(prova1) 70.00000  
## sd(prova1)     22.25395  
## var(prova1)    495.23810  
## length(prova1) 7.00000
```

Um vetor de estatísticas

```
# Medidas resumo de um vetor de estatísticas.  
quantile(df1$prova1, probs = c(0.25, 0.75))  
## 25% 75%  
## 50.0 77.5  
  
table(df1$curso)  
  
##  
## Est Mat  
## 4     3
```

Medidas resumo

Estatísticas definidas pelo usuário

```
df1 %>%
  summarise(CV = 100 *
            sd(prova1)/mean(prova1))

# Define uma função para facilitar.
CV <- function(x, ...) {
  100 * sd(x, ...)/mean(x, ...)
}

df1 %>%
  summarise_if(.predicate = is.numeric,
               .funs = CV,
               na.rm = TRUE)

## # A tibble: 1 × 5
##   matricula prova1 prova2 prova3 faltas
##       <dbl>    <dbl>    <dbl>    <dbl>   <dbl>
## 1      58.6    34.6    18.4    33.4    105.
```

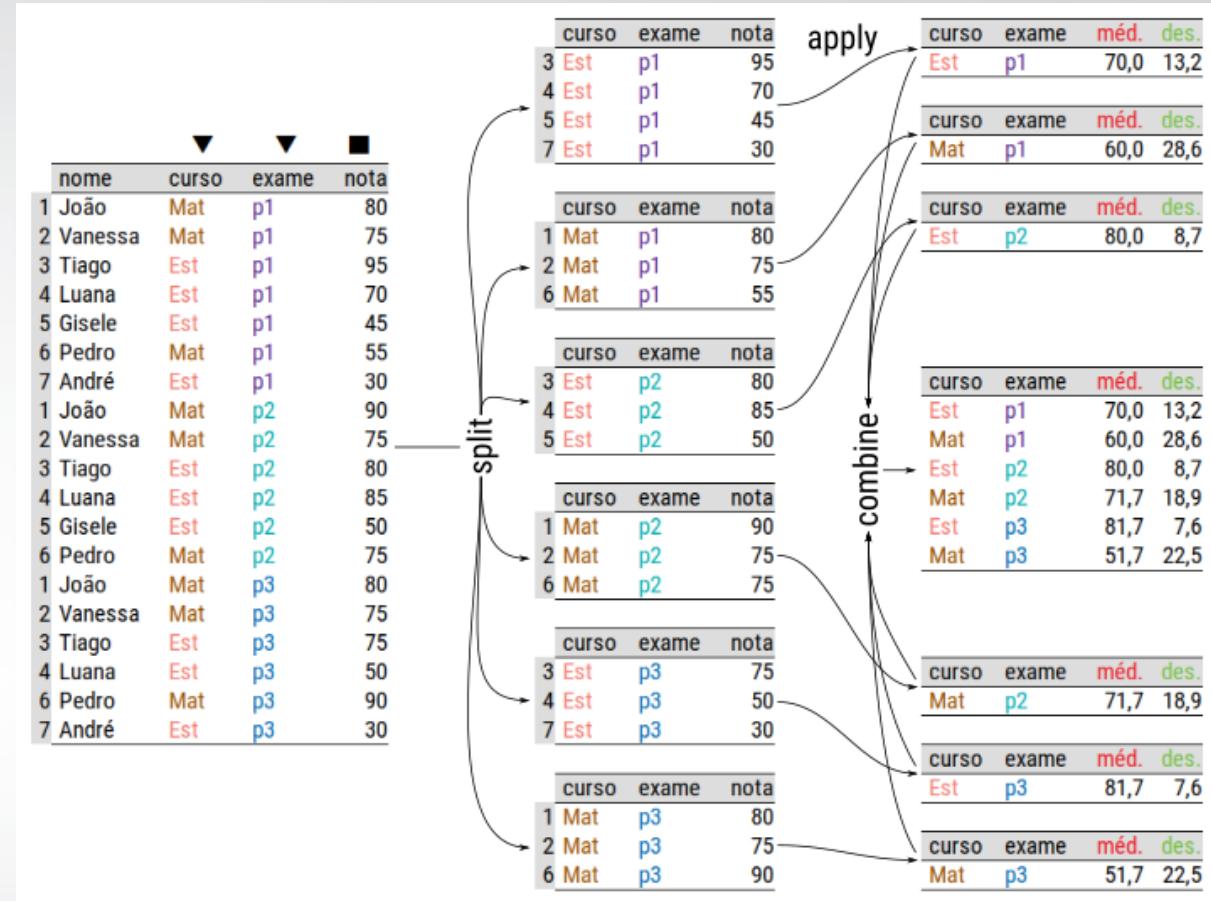
Várias estatísticas para várias variáveis

```
df1 %>%
  summarise_at(vars(prova1:prova3, faltas),
               .funs = c("mean",
                         "sd",
                         "CV"),
               na.rm = TRUE) %>%
  glimpse()

## # Rows: 1
## # Columns: 12
## # $ prova1_mean <dbl> 64.28571
## # $ prova2_mean <dbl> 75.83333
## # $ prova3_mean <dbl> 66.66667
## # $ faltas_mean <dbl> 7.428571
## # $ prova1_sd   <dbl> 22.25395
## # $ prova2_sd   <dbl> 13.93437
## # $ prova3_sd   <dbl> 22.28602
## # $ faltas_sd   <dbl> 7.807201
## # $ prova1_CV   <dbl> 34.61725
## # $ prova2_CV   <dbl> 18.37499
## # $ prova3_CV   <dbl> 33.42903
## # $ faltas_CV   <dbl> 105.0969
```

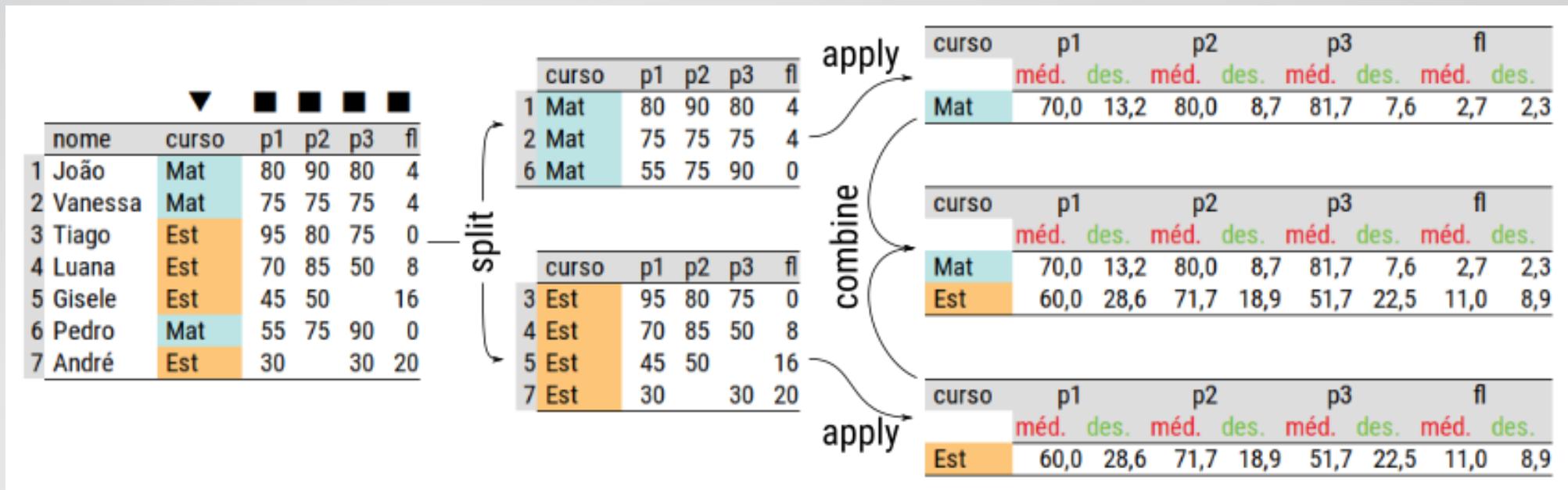
Agregação

- ▶ Consiste em aplicar estatísticas em variáveis fazendo a extratificação por outras variáveis.
- ▶ São tarefas conhecidas como *split-apply-combine*.
- ▶ Ou também chamadas de *GROUP BY*.



Agregação de uma tabela.

Agregação



Agregação de uma tabela.

Agregação

Cálculo de frequências

```
# Registros por curso.  
df1 %>%  
  count(curso)  
  
## # A tibble: 2 × 2  
##   curso     n  
##   <chr> <int>  
## 1 Est        4  
## 2 Mat        3
```

```
# Média final por curso.  
df1 %>%  
  mutate_at(vars(prova1:prova3),  
            ~replace_na(., 0)) %>%  
  mutate(media = (prova1 + prova2 +  
                  prova3)/3,  
         condic = ifelse(media >= 70,  
                           "Aprov",  
                           "Reprov")) %>%  
  count(curso, condic)
```

```
## # A tibble: 3 × 3  
##   curso condic     n  
##   <chr> <chr> <int>  
## 1 Est    Aprov      1  
## 2 Est    Reprov     3  
## 3 Mat    Aprov      3
```

Agregação

Medidas descritivas resumo

```
# Nota média por curso em cada avaliação.  
df1 %>%  
  group_by(curso) %>%  
  summarise_at(vars(prova1:prova3),  
               "mean",  
               na.rm = TRUE)
```

```
## # A tibble: 2 × 4  
##   curso prova1 prova2 prova3  
##   <chr>  <dbl>  <dbl>  <dbl>  
## 1 Est      60    71.7   51.7  
## 2 Mat      70    80     81.7
```

```
# Média final por curso.  
df1 %>%  
  mutate_at(vars(prova1:prova3),  
           ~replace_na(., 0)) %>%  
  mutate(media = (prova1 + prova2 +  
                  prova3)/3) %>%  
  group_by(curso) %>%  
  summarise_at("media",  
               c("mean", "sd",  
                 "min", "max"),  
               na.rm = TRUE)
```

```
## # A tibble: 2 × 5  
##   curso  mean    sd    min    max  
##   <chr>  <dbl>  <dbl>  <dbl>  <dbl>  
## 1 Est    50.8  29.9   20    83.3  
## 2 Mat    77.2  5.36  73.3  83.3
```

Agregação

- ▶ Uma vez que uma tabela é agrupada, várias informações e métodos estão disponíveis.
- ▶ Isso permite criar uma lista de tabelas para operar com programação funcional.

```
# Cria um tabela de dados agrupados.  
u <- df1 %>%  
  group_by(curso)  
  
# Inspecciona os métodos disponíveis.  
class(u)  
methods(class = "grouped_df")  
  
# Usa alguns dos métodos.  
n_groups(u)  
group_vars(u)  
group_size(u)  
group_indices(u)
```

```
# Cria uma lista de tabelas.  
u <- df1 %>%  
  group_split(curso)  
# str(u, vec.len = 2,  
#       max.level = 2, give.attr = FALSE)  
glimpse(u)  
  
## # list<tibble[,7]> [1:2]  
## $ : tibble [4 × 7] (S3:tbl_df/tbl/data.frame)  
## $ : tibble [3 × 7] (S3:tbl_df/tbl/data.frame)  
## @ ptype: tibble [0 × 7] (S3:tbl_df/tbl/data.frame)
```

Concatenação

- A concatenação permite adicionar novas observações a uma tabela ou novas variáveis.
- Seja por linha ou colunas, entradas com `NA` são criadas para os índices que não foram especificados.

	mat. nome	p1	p2	p3	fl
1	256 João	80	90	80	4
2	487 Vanessa	75	75	75	4
3	965 Tiago	95	80	75	0
4	125 Luana	70	85	50	8
5	458 Gisele	45	50	16	
6	874 Pedro	55	75	90	0
7	963 André	30	30	20	

	mat. nome	p1	p2	fl
1	505 Bia	65	85	0
2	658 Carlos	75	80	2
3	713 Cris	75	90	2

Concatenação de duas tabelas.

Concatenação

De linhas (vertical)

```
# Concatenação na vertical (pilha).  
bind_rows(df1[1:3, c(1, 3, 5)],  
          df1[5:7, c(1, 3, 5, 4)],  
          df1[4, c(1, 5, 4)])
```

```
## # A tibble: 7 × 4  
##   matricula curso prova2 prova1  
##       <dbl> <chr>  <dbl>  <dbl>  
## 1      256 Mat      90     NA  
## 2      487 Mat      75     NA  
## 3      965 Est      80     NA  
## 4      458 Est      50     45  
## 5      874 Mat      75     55  
## 6      963 Est     NA      30  
## 7     125 <NA>     85     70
```

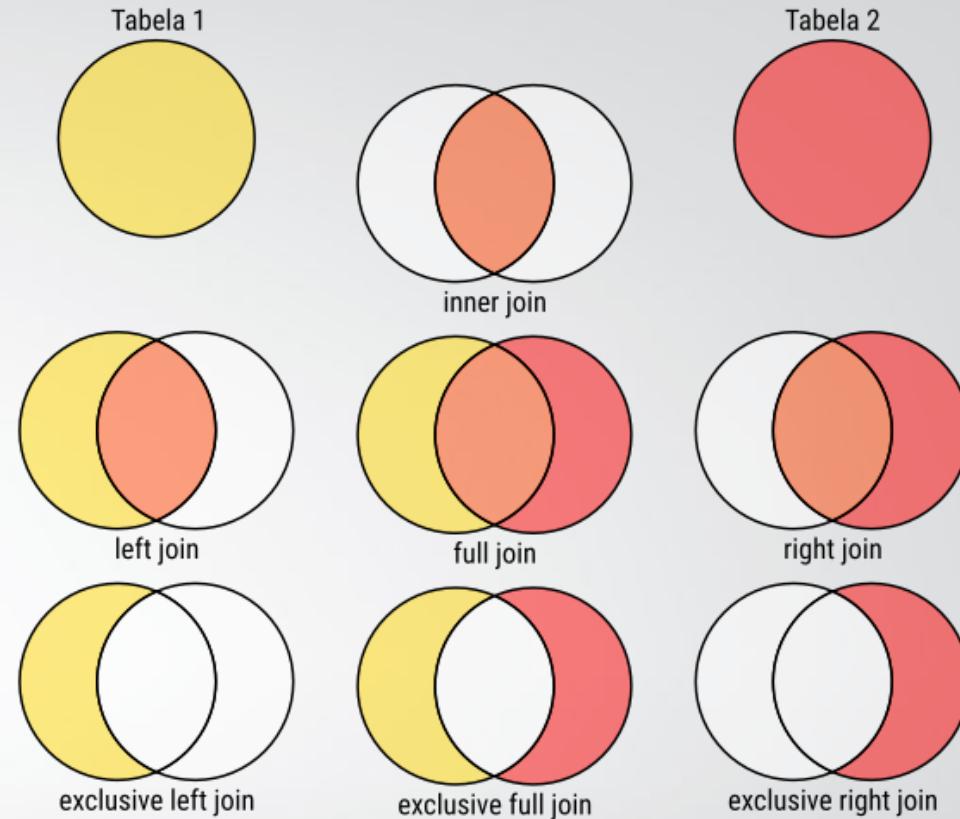
De colunas (horizontal)

```
# Concatenação na horizontal (fila).  
bind_cols(df1[, c(1:3)],  
          df1[, c(6:7)])
```

```
## # A tibble: 7 × 5  
##   matricula nome  curso prova3 faltas  
##       <dbl> <chr> <chr>  <dbl>  <dbl>  
## 1      256 João  Mat      80     4  
## 2      487 Vanessa Mat      75     4  
## 3      965 Tiago Est      75     0  
## 4      125 Luana Est      50     8  
## 5      458 Gisele Est     NA    16  
## 6      874 Pedro Mat      90     0  
## 7      963 André Est      30    20
```

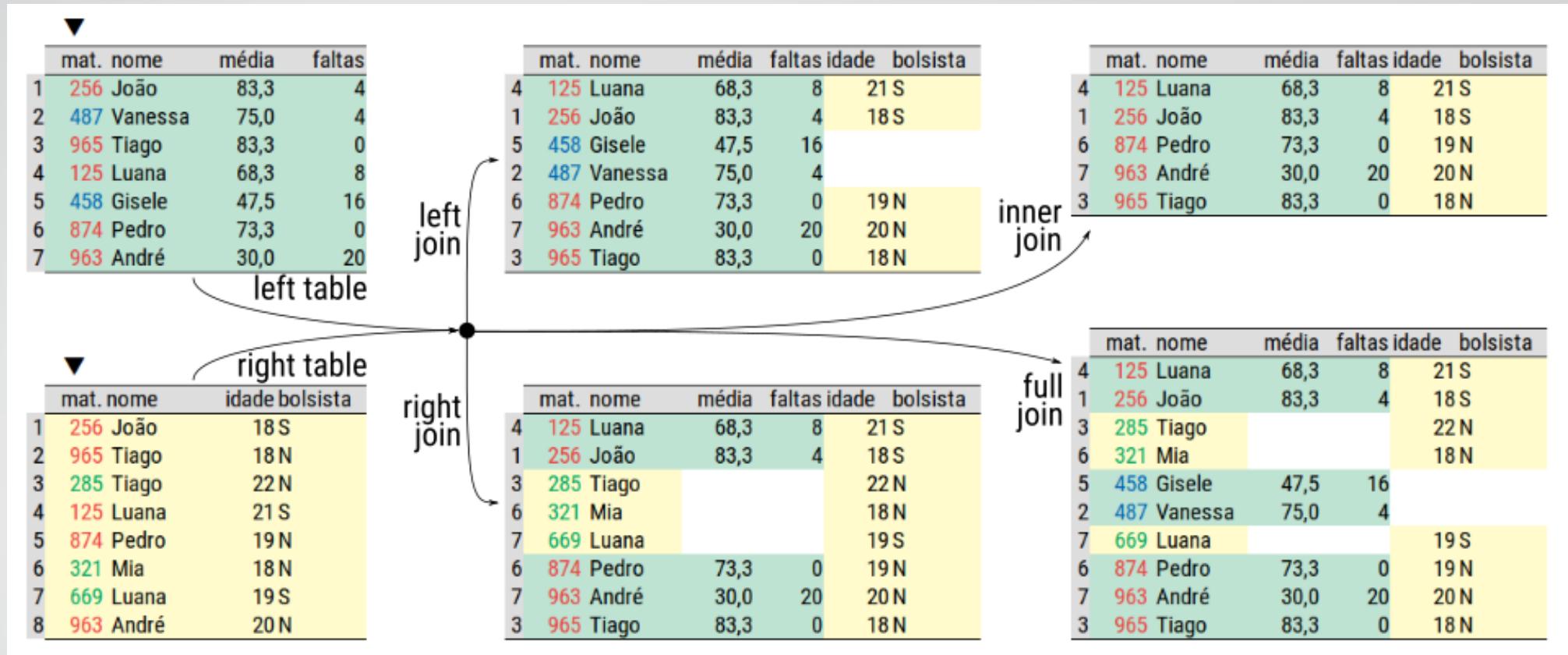
Junções

- ▶ Junções permitem parear dados de tabelas separadas quando elas possuem uma chave (ou chave primária).
- ▶ As operações de junção podem ser inicialmente de 4 tipos:
 - ▶ Junção por interseção (*inner join*).
 - ▶ Junção por união (*full join*).
 - ▶ Junção à esquerda (*left join*).
 - ▶ Junção à direita (*right join*).
 - ▶ Existe também os *exclusive joins*.



Tipos de junções de tabelas ilustrado com diagramas de Veen.

Junções



Junções de tabelas do tipo inclusivas.

Junções

```
# Full join = união.  
full_join(df1, df_extra,  
          by = c("matricula" = "mat",  
                 "nome"))  
  
# Inner join = intersecção.  
inner_join(df1,  
           df_extra,  
           by = c("matricula" = "mat",  
                  "nome"))  
  
## # A tibble: 4 × 9  
##   matricula nome  curso prova1 prova2  
##       <dbl> <chr> <chr>  <dbl>  <dbl>  
## 1       256 João  Mat      80      90  
## 2       965 Tiago Est      95      80  
## 3       125 Luana Est      70      85  
## 4       874 Pedro Mat      55      75  
## # ... with 4 more variables:  
## #   prova3 <dbl>, faltas <dbl>,  
## #   idade <dbl>, bolsista <chr>
```

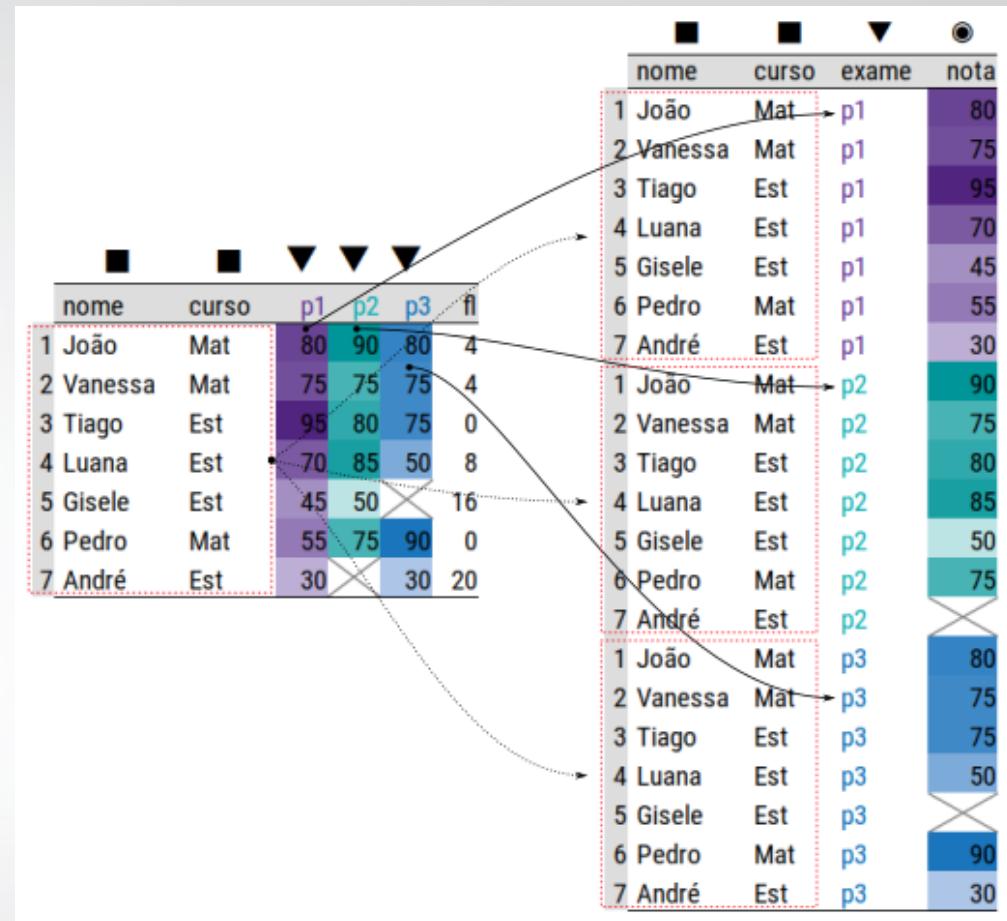
```
# Todos os que estão na 1º tabela  
left_join(df1, df_extra,  
          by = c("matricula" = "mat",  
                 "nome"))  
  
# Todos os que estão na 2º tabela  
right_join(df1, df_extra,  
           by = c("matricula" = "mat",  
                  "nome"))
```

```
# Os da 2º que não aparecem na 1º.  
anti_join(df1, df_extra,  
          by = c("matricula" = "mat",  
                 "nome"))
```

```
## # A tibble: 3 × 7  
##   matricula nome    curso prova1 prova2  
##       <dbl> <chr> <chr>  <dbl>  <dbl>  
## 1       487 Vanessa Mat      75      75  
## 2       458 Gisele  Est      45      50  
## 3       963 André   Est      30     NA  
## # ... with 2 more variables:  
## #   prova3 <dbl>, faltas <dbl>
```

Rearranjo

- São operações de *reshaping* da tabela.
- Modificam a disposição dos registros.
 - Empilhar ou amontoar um conjunto de variáveis.
 - Desempilhar ou esparramar os níveis de uma variável.

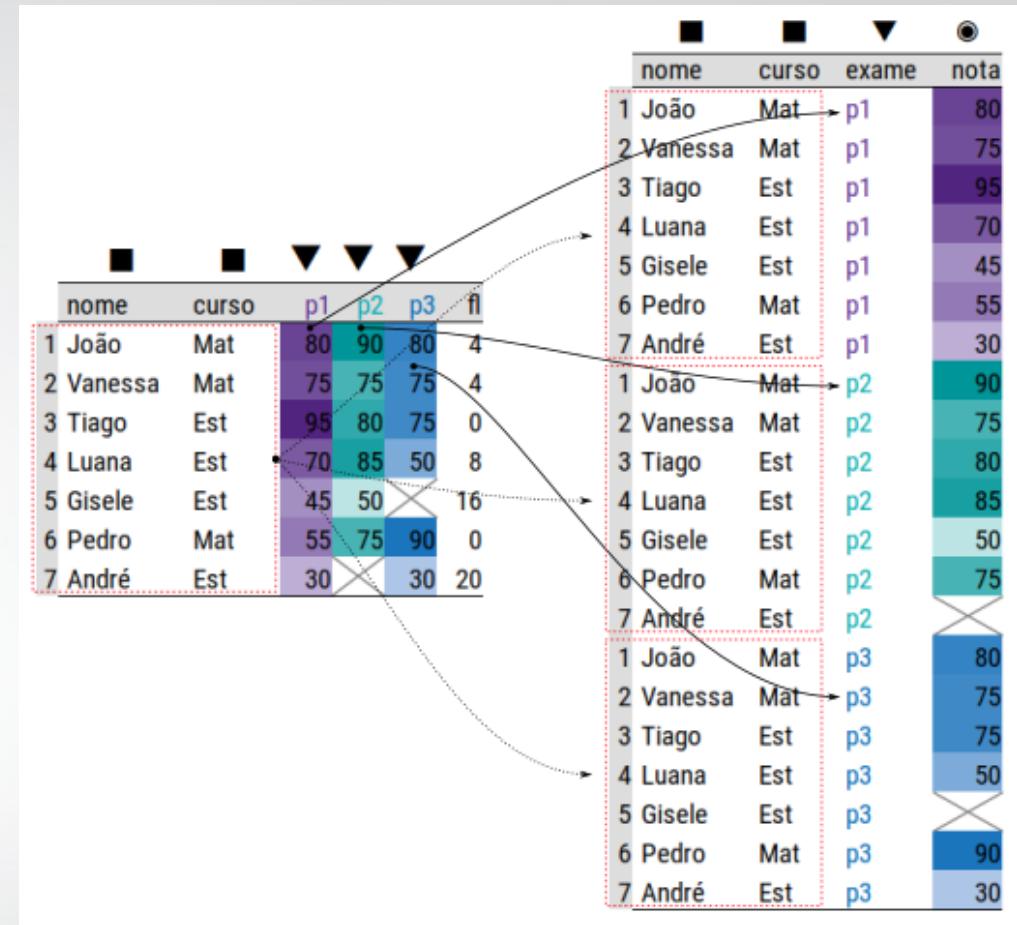


Modificação da disposição com empilhamento.

Rearranjo · Empilhar

```
# Empilhar.  
u <- df1 %>%  
  pivot_longer(names_to = "exame",  
              values_to = "nota",  
              prova1:prova3)  
head(u) %>%  
  select(nome, curso, exame, nota)
```

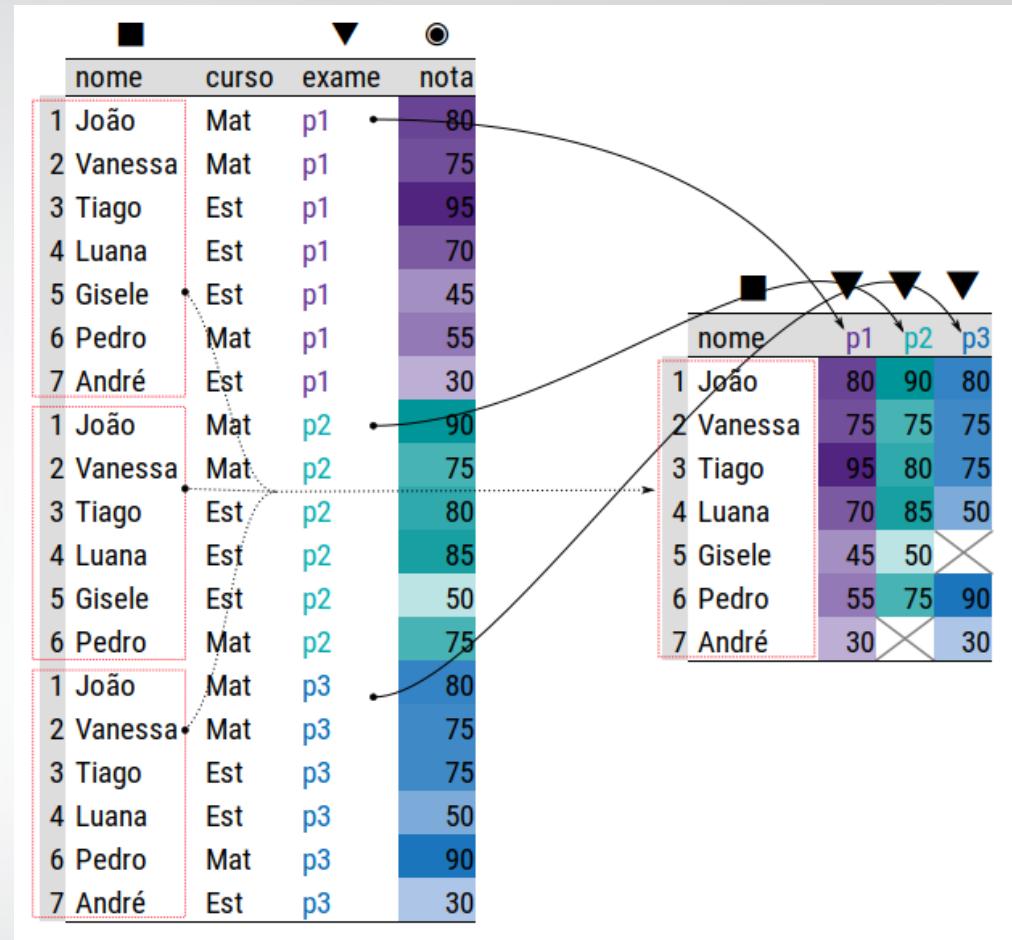
```
## # A tibble: 6 × 4  
##   nome    curso exame  nota  
##   <chr>   <chr> <chr>  <dbl>  
## 1 João     Mat    prova1    80  
## 2 João     Mat    prova2    90  
## 3 João     Mat    prova3    80  
## 4 Vanessa  Mat    prova1    75  
## 5 Vanessa  Mat    prova2    75  
## 6 Vanessa  Mat    prova3    75
```



Modificação da disposição com empilhamento.

Rearranjo · Esparramar

```
# Spread = esparramar.  
v <- u %>%  
  pivot_wider(names_from = "exame",  
             values_from = "nota")  
head(v)  
  
## # A tibble: 6 × 7  
##   matricula nome  curso faltas prova1  
##       <dbl> <chr> <chr>    <dbl>  <dbl>  
## 1        256 João   Mat      4     80  
## 2        487 Vanessa Mat      4     75  
## 3        965 Tiago  Est      0     95  
## 4        125 Luana  Est      8     70  
## 5        458 Gisele Est     16     45  
## 6        874 Pedro  Mat      0     55  
## # ... with 2 more variables:  
## #   prova2 <dbl>, prova3 <dbl>
```



Modificação da disposição com desempilhamento.

Considerações finais

- ▶ Foram vistas as principais operações com dados tabulares.
- ▶ Para mais detalhes consulte a documentação e livros dedicados.
- ▶ Exercícios serão aplicados para fixação do conteúdo.
- ▶ Links para guias de referência e tutoriais serão dados no final.
- ▶ Conexão com banco de dados será visto em detalhe em outra disciplina.

Exercícios para usar o dplyr

- ▶ Ler os dados em
<http://leg.ufpr.br/~walmes/data/ninfas.txt>.
- ▶ Ordenação.
 1. Ordenar pelo valor do terço `superior`.
 2. Ordenar pelo valor do terço `medio` de forma descrescente.
 3. Ordenar pelas datas > variedade > bloco.
- ▶ Filtros.
 1. Filtrar só para a variedade BRS 245 RR.
 2. Filtrar só para a variedade BRS 245 RR e EMBRAPA 48.
 3. Filtrar só para variedades diferentes de EMBRAPA 48.
 4. Filtrar quando `superior` com mais de 30 e inferior com mais de 20.
 5. Filtrar para `medio` entre 20 e 50.
 6. Filtrar para avaliações entre 2009-12-24 e 2010-01-11.
 7. Filtrar para a soma dos terços maior que 100.

- Seleção de variáveis.
 1. Selecionar apenas os terços.
 2. Remover a variável bloco.
 3. Mudar a ordem das colunas finais para **inferior, medio e superior**.
 4. Manter as variáveis com nome terminado em **rior**.
- Modificação/criação de variáveis.
 1. Criar a variável total somando os terços.
 2. Criar a diferença entre o terço superior e inferior.
 3. Converter bloco e variedade para fator.
 4. Criar a raiz quadrada do número de ninfas em cada terço.
- Medidas descritivas gerais.
 1. Total de ninfas no terço superior.
 2. Total de ninfas em cada um dos terços.
 3. Média e desvio-padrão de ninfas em cada terço.
 4. Correlação do número de ninfas entre os terços (duas a duas).
- Medidas descritivas por extrato.
 1. Total de registros por variedade.
 2. Total de registros por data.
 3. Total de registros por variedade e data.
 4. Total de ninfas no terço superior por data.
 5. Total de ninfas nos 3 terços juntos por data.
 6. Total de ninfas nos 3 terços juntos por variedade, ordene no final.
 7. Total de ninfas nos 3 terços juntos por data e variedade. Guardar em objeto para usar a seguir.
 8. A variedade com mais ninfas em cada data.
 9. A data com mais ninfas em cada variedade.