

Laboratório de Biologia Computacional e Molecular

Centro de Biotecnologia da UFRGS  
Universidade Federal do Rio Grande do Sul



# R para Ciências da Vida (BCM13065) Aula 8

PPGBCM - UFRGS

Diego Bonatto  
2024/2

# Gráficos e representações de dados

- Os gráficos são ferramentas essenciais para analisar, comunicar e compreender dados biológicos.

- **Facilitar a Visualização de Dados Complexos**
  - Importância: Dados biológicos frequentemente envolvem grandes quantidades de informações, como sequências genômicas, redes metabólicas e padrões de expressão gênica. Gráficos tornam esses dados compreensíveis, transformando números em representações visuais.
  - Exemplo: Um gráfico de calor (heatmap) pode mostrar padrões de expressão gênica em diferentes condições experimentais.
- **Identificação de Padrões e Relações**
  - Importância: Gráficos ajudam a identificar tendências, correlações e outliers que podem não ser óbvios em tabelas de dados.
  - Exemplo:
    - Gráficos de dispersão (scatter plots) mostram correlações entre variáveis biológicas, como concentração de proteínas versus atividade enzimática.
    - Gráficos de linha exibem tendências temporais, como a variação de uma população de células ao longo do tempo.

# Gráficos e representações de dados

- **Suporte à Tomada de Decisões**
- **Importância:** Representações visuais ajudam os pesquisadores e profissionais a tomar decisões informadas com base em dados concretos.
- **Exemplo:**
  - Um boxplot comparando diferentes tratamentos pode indicar a eficácia de um novo medicamento.
  - Um gráfico de sobrevivência (Kaplan-Meier) pode auxiliar na avaliação da eficácia de terapias em estudos clínicos.
- **Comunicação Clara de Resultados**
- **Importância:** Gráficos são uma forma universal de apresentar resultados de forma intuitiva, acessível até mesmo para pessoas de fora do campo específico.
- **Exemplo:**
  - Em um artigo científico, gráficos de barras ou torta ajudam a sintetizar dados complexos, como a distribuição de espécies em um ecossistema.
  - Redes metabólicas gráficas ajudam a descrever interações entre proteínas ou genes.

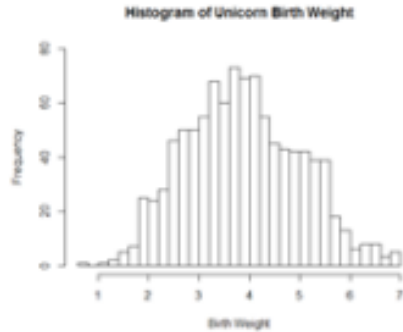
# Gráficos e representações de dados

- **Comunicação Clara de Resultados**
- **Importância:** Gráficos são uma forma universal de apresentar resultados de forma intuitiva, acessível até mesmo para pessoas de fora do campo específico.
- **Exemplo:**
  - Em um artigo científico, gráficos de barras ou torta ajudam a sintetizar dados complexos, como a distribuição de espécies em um ecossistema.
  - Redes metabólicas gráficas ajudam a descrever interações entre proteínas ou genes.

# Gráficos e representações de dados

- **Diferentes representações para diferentes perguntas**
- **Cada tipo de gráfico é mais adequado para um tipo de dado ou pergunta específica:**
  - Gráficos de Dispersão (Scatter Plots): Correlações ou distribuição de variáveis contínuas.
  - Boxplots e Violin Plots: Distribuições estatísticas e variabilidade de dados.
  - Gráficos de Linhas (Line Plots): Tendências temporais ou dependências de variáveis contínuas.
  - Heatmaps: Análise de grandes matrizes de dados, como expressão gênica.
  - Diagramas de Redes (Network Graphs): Interações biológicas complexas (e.g., redes metabólicas ou de sinalização celular).

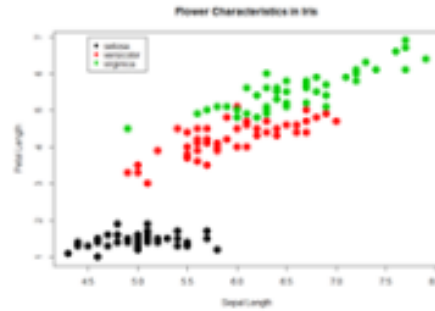
# Gráficos – R base



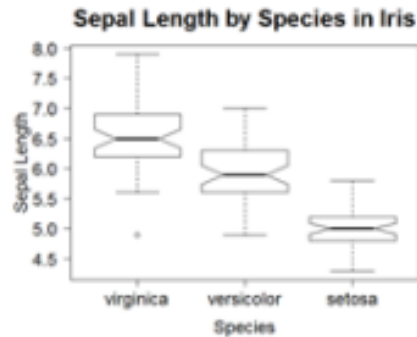
1. Basic Histogram



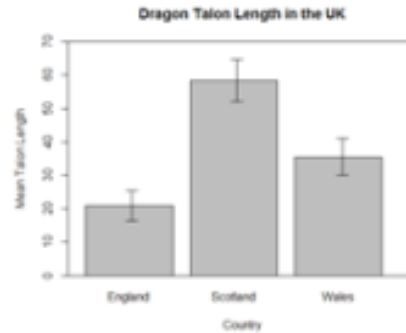
2. Line Graph with Regression



3. Scatterplot with Legend



4. Boxplot with reordered/  
formatted axes



5. Boxplot with Error Bars

# Gráficos – R base - parâmetros

**Opções:**

**lwd → line width**

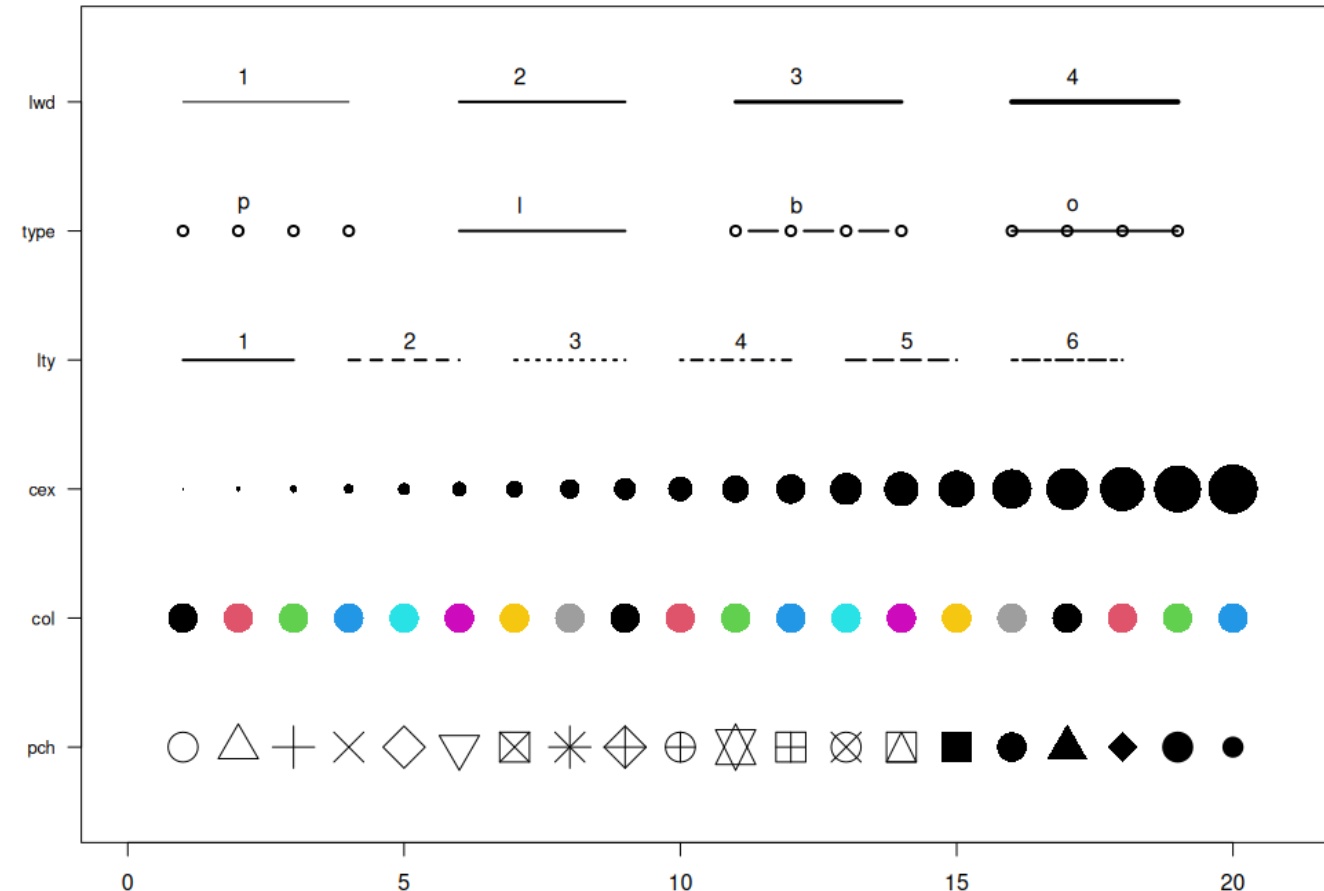
**type → link between dots**

**lty → line type**

**cex → shape size**

**col → control colors**

**pch → marker shape**



Categoria	Parâmetro	Descrição	Exemplo
Elementos Básicos	main	Título principal do gráfico.	main = "Meu Gráfico"
Elementos Básicos	sub	Subtítulo do gráfico.	sub = "Subtítulo"
Elementos Básicos	xlab, ylab	Rótulos dos eixos X e Y.	xlab = "Eixo X", ylab = "Eixo Y"
Eixos	xlim, ylim	Intervalos dos eixos X e Y.	xlim = c(0, 10)
Eixos	xaxt, yaxt	Controle da exibição dos eixos ("n" para suprimir).	xaxt = "n"
Eixos	las	Orientação dos rótulos dos eixos (0 a 3).	las = 1
Eixos	tck, tcl	Tamanho das marcas dos eixos (ticks).	tck = 0.02
Pontos e Linhas	pch	Estilo dos pontos.	pch = 19
Pontos e Linhas	lty	Tipo de linha (1: contínua, 2: tracejada, etc.).	lty = 2
Pontos e Linhas	lwd	Espessura da linha.	lwd = 2
Pontos e Linhas	col	Cor dos pontos, linhas ou elementos.	col = "blue"
Cores	col.axis	Cor dos textos dos eixos.	col.axis = "red"
Cores	col.lab	Cor dos rótulos dos eixos.	col.lab = "green"
Cores	col.main	Cor do título principal.	col.main = "purple"
Cores	bg	Cor do fundo do gráfico.	bg = "lightgray"
Cores	fg	Cor dos elementos do primeiro plano.	fg = "black"
Texto e Fontes	cex	Escala geral do tamanho do texto.	cex = 1.5
Texto e Fontes	cex.axis	Escala do texto dos eixos.	cex.axis = 1.2
Texto e Fontes	font	Estilo da fonte (1: normal, 2: negrito, 3: itálico, 4: negrito e itálico).	font = 2
Legendas	legend	Adiciona legenda (usado com a função legend()).	legend("topright", ...)
Legendas	bty	Tipo de borda (e.g., "o", "n" para nenhuma).	bty = "n"
Legendas	inset	Posição da legenda relativa ao gráfico.	inset = 0.05
Disposição dos Dados	type	Tipo de plotagem ("p": pontos, "l": linhas, "b": ambos, etc.).	type = "b"
Disposição dos Dados	asp	Taxa de aspecto do gráfico.	asp = 1
Outros Parâmetros	mar	Margens do gráfico (vetor com 4 valores).	mar = c(5, 4, 4, 2)
Outros Parâmetros	oma	Margens externas para múltiplos gráficos.	oma = c(1, 1, 1, 1)
Outros Parâmetros	adj	Ajuste horizontal do texto (0: esquerda, 1: direita).	adj = 0.5
Outros Parâmetros	xaxs, yaxs	Estilo dos limites dos eixos ("r" para arredondamento).	xaxs = "i"



# R Base Graphics Cheatsheet

## SET GRAPHICAL PARAMETERS

the following can *only* be set with `par()`

`par(...)`

<i>multiple plots</i>	<code>mfcol = c(nrow,ncol)</code>	<i>plot margins (outer)</i>	<code>oma = c(bottom, left, top, right) default: c(0, 0, 0, 0) lines</code>
<i>plot margins</i>	<code>mar = c(bottom, left, top, right) default: c(5.1, 4.1, 4.1, 2.1) lines</code>	<i>query x &amp; y limits</i>	<code>par("usr")</code>

## CREATE A NEW PLOT

<b>Bar charts</b>	<code>barplot(height, ...)</code>	<b>Histograms</b>	<code>hist(x, ...)</code>
<i>bar labels</i>	<code>names.arg =</code>	<i>breakpts</i>	<code>breaks =</code>
<i>border</i>	<code>border =</code>		
<i>fill color</i>	<code>col =</code>		
<i>horizontal</i>	<code>horiz = TRUE</code>		
		<b>Line charts</b>	<code>plot(x, type = "l")</code>
		<i>line type</i>	<code>lty = "blank" 0 "solid" 1 "dashed" 2 "dotted" 3</code>
		<i>line width</i>	<code>lwd =</code>
<b>Box plots</b>	<code>boxplot(x, ...)</code>		
<i>horizontal</i>	<code>horizontal = TRUE</code>		
<i>box labels</i>	<code>names =</code>		
<b>Dot plots</b>	<code>dotchart(x, ...)</code>	<b>Scatterplots</b>	<code>plot(x, ...)</code>
<i>dot labels</i>	<code>labels =</code>	<i>symbol</i>	<code>pch =</code>

## REMOVE

<i>axis labels</i>	<code>ann = FALSE</code>
<i>axis, tickmarks, and labels</i>	<code>xaxt = "n" yaxt = "n"</code>
<i>plot box</i>	<code>bty = "n"</code>

NOTE: Many of the parameters here can be also be set in `par()`. See R help for more options.

## ADJUST

<i>allow plotting out of plot region</i>	<code>xpd = TRUE</code>
<i>aspect ratio</i>	<code>asp =</code>
<i>axis limits</i>	<code>xlim =, ylim =</code>
<i>axis lines to match axis limits</i>	<code>xaxs = "i", yaxs = "i" (internal axis calculation)</code>

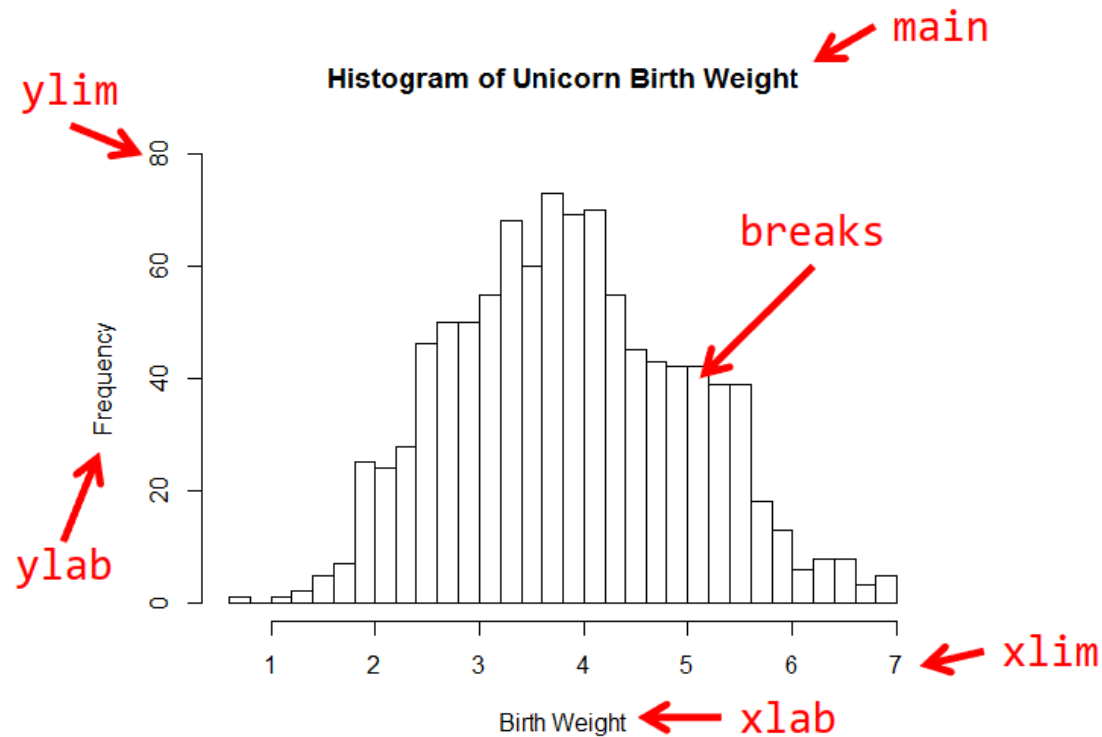
## ADD TEXT

<b>location</b>	<code>xlab =, ylab =</code>	<b>size</b> (magnification factor)	<code>cex =</code>
<i>axis labels</i>	<code>sub =</code>	<i>all elements</i>	<code>cex.lab =</code>
<i>subtitle</i>	<code>main =</code>	<i>axis labels</i>	<code>cex.sub =</code>
<i>title</i>		<i>tick mark labels</i>	<code>cex.axis =</code>
		<i>title</i>	<code>cex.main =</code>
<b>style</b>	<code>font = 1 (plain) 2 (bold) 3 (italic) 4 (bold italic)</code>	<b>position</b>	
<i>font face</i>		<i>text direction</i>	<code>las = 1 (horizontal)</code>
<i>font family</i>	<code>family = "serif" "sans" "mono"</code>	<i>justification</i>	<code>adj = 0 .5 1 (left, center, right)</code>

## ADD TO AN EXISTING PLOT

<b>Add new plot</b>	<code>[any plot function] (... , add = TRUE)</code>	<b>Lines</b>	<code>lines(x, ...)</code>
ex. <code>barplot(x, add = TRUE)</code>		<i>line style</i>	<code>lty =</code>
		<i>line width</i>	<code>lwd =</code>
		<i>color</i>	<code>col =</code>
<b>Axes</b>	<code>axis(side, ...)</code>	<b>Points</b>	<code>points(x, ...)</code>
<i>location</i>	<code>side = 1 2 3 4 (bottom, left, top, right)</code>	<i>symbol</i>	<code>pch =</code>
<i>tick mark:</i>			
<i>labels</i>	<code>labels =</code>		
<i>location</i>	<code>at =</code>		
<i>remove</i>	<code>tick = FALSE</code>		
<i>rotate text</i>	<code>las = 1 (horizontal)</code>	<i>color</i>	<code>col =</code>
		<i>fill color</i>	<code>bg = (pch: 21-25 only)</code>
<b>Axis labels</b>	<code>mtext(text, ...)</code>	<b>Text</b>	<code>text(x, y, text, ...)</code>
<i>location</i>	<code>side = 1 2 3 4 (bottom, left, top, right)</code>	<i>position (rel. to x,y)</i>	<code>pos = 1 2 3 4 (below, left, above, right) (default=center)</code>
<i>lines to skip</i>	<code>line = (from plot region, default=0)</code>	<b>Title</b>	<code>title(main, ...)</code>
<i>position</i>	<code>at = x or y-coord (depending on side)</code>	<i>axis labels</i>	<code>xlab =, ylab =</code>
<i>justification</i>	<code>adj = 0 .5 1 (left, center, right)</code>	<i>subtitle</i>	<code>sub =</code>
		<i>title</i>	<code>main =</code>

Joyce Robbins, joycerobbins1@gmail.com

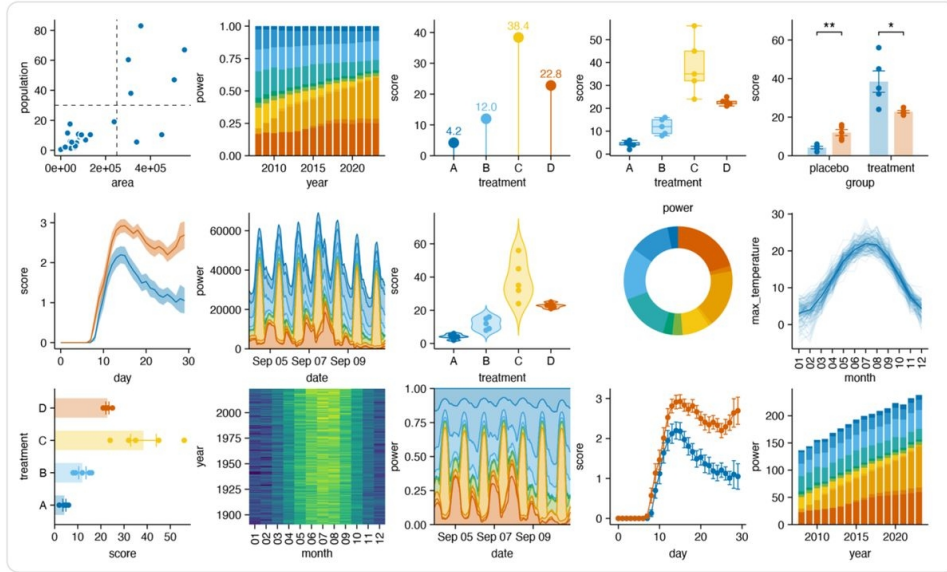


```
99 #~ FINAL PLOT:
100
101 hist(unicorns$birthweight,           # x value
102      breaks = 40,                   # number of cells
103      xlab = "Birth weight",          # x-axis label
104      main = "Histogram of Unicorn Birth weight", # plot title
105      ylim = c(0,80))                # limits of the y axis (min,max)
106
```

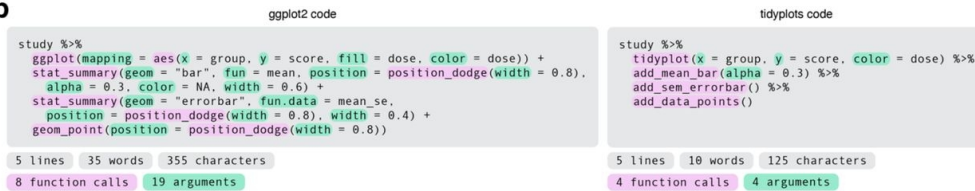
Aspecto	Vantagens	Desvantagens
<b>Simplicidade</b>	Interface simples e fácil de usar para gráficos básicos.	Pode ser limitada para gráficos mais complexos.
<b>Personalização</b>	Permite personalizar quase todos os aspectos dos gráficos.	Personalizações avançadas podem ser complexas e requerer muitos parâmetros.
<b>Performance</b>	Muito eficiente para gráficos simples, mesmo em grandes conjuntos de dados.	Performance pode diminuir ao criar gráficos complexos ou detalhados.
<b>Compatibilidade</b>	Integrado nativamente no R, sem necessidade de pacotes adicionais.	Gráficos podem não ter o mesmo apelo visual que alternativas modernas como ggplot2.
<b>Documentação</b>	Amplamente documentado e suportado pela comunidade.	Nem sempre intuitivo para iniciantes, especialmente para personalizações avançadas.
<b>Flexibilidade</b>	Suporte a uma ampla gama de tipos de gráficos (linha, barra, dispersão etc.).	Gráficos interativos não são suportados diretamente.

# Gráficos - ggplot2

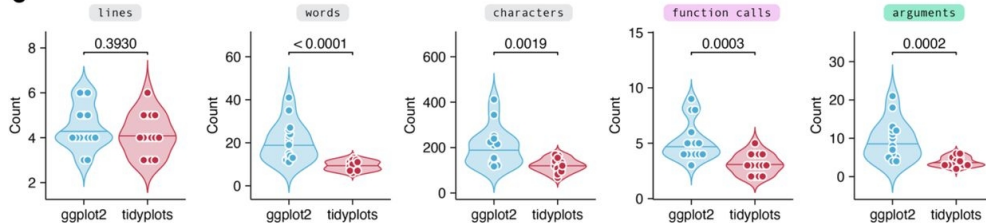
a



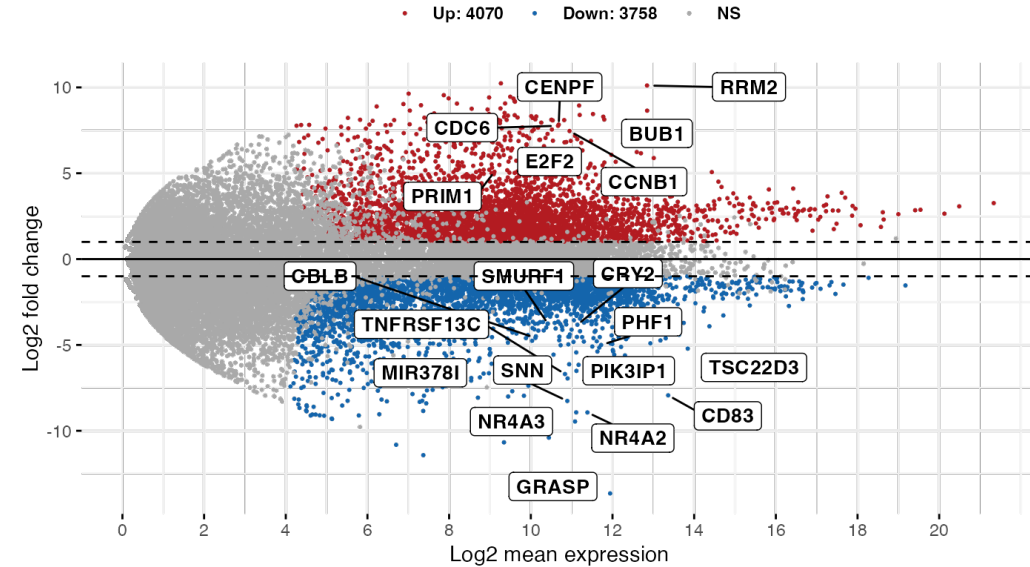
b



c



Group 1 → Group 2



# Gráficos - ggplot2

## Basics

**ggplot2** is based on the **grammar of graphics**, the idea that you can build every graph from the same components: a **data** set, a **coordinate system**, and **geoms**—visual marks that represent data points.



To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **x** and **y** locations.



Complete the template below to build a graph.

```
ggplot (data = <DATA>) +  
  <GEOM_FUNCTION> (mapping = aes(<MAPPINGS>),  
    stat = <STAT>, position = <POSITION>) +  
  <COORDINATE_FUNCTION> +  
  <FACET_FUNCTION> +  
  <SCALE_FUNCTION> +  
  <THEME_FUNCTION>
```

required

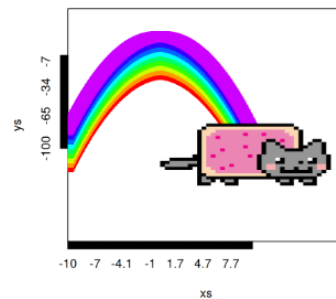
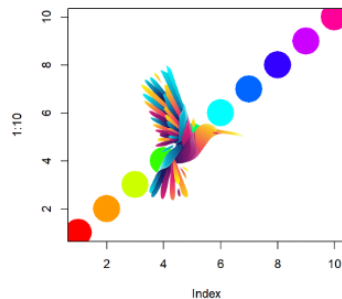
Not required, sensible defaults supplied

**ggplot**(data = mpg, aes(x = cty, y = hwy)) Begins a plot that you finish by adding layers to. Add one geom function per layer.

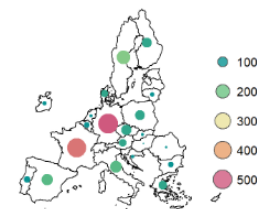
**last\_plot()** Returns the last plot.

**ggsave**("plot.png", width = 5, height = 5) Saves last plot as 5' x 5' file named "plot.png" in working directory. Matches file type to file extension.

Aspecto	Vantagens	Desvantagens
<b>Estilo Visual</b>	Gráficos modernos, esteticamente agradáveis e altamente personalizáveis.	Pode ser mais complexo para ajustar configurações simples em comparação à base R.
<b>Consistência</b>	Usa a gramática de gráficos, tornando o código estruturado e legível.	Exige aprendizado inicial para entender a gramática de gráficos.
<b>Flexibilidade</b>	Suporte integrado para gráficos avançados e combinações complexas de dados.	Gráficos básicos podem exigir mais código do que na base R.
<b>Extensibilidade</b>	Suporte para temas, camadas e extensões por meio de pacotes adicionais.	Muitos pacotes podem ser necessários para gráficos especializados.
<b>Comunidade e Suporte</b>	Extensa comunidade, tutoriais e exemplos disponíveis.	Pode ser difícil localizar soluções para problemas muito específicos.
<b>Interatividade</b>	Compatível com pacotes como <code>plotly</code> para gráficos interativos.	Gráficos interativos dependem de pacotes externos e maior conhecimento técnico.

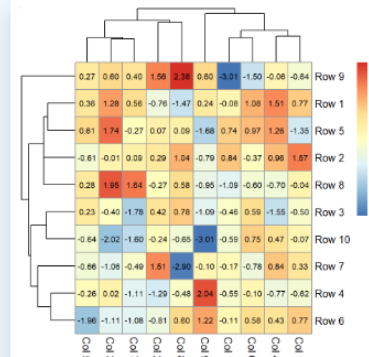
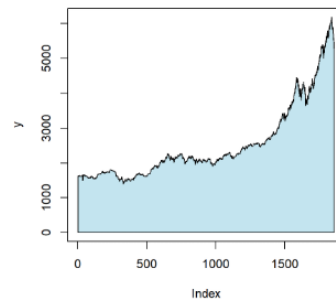
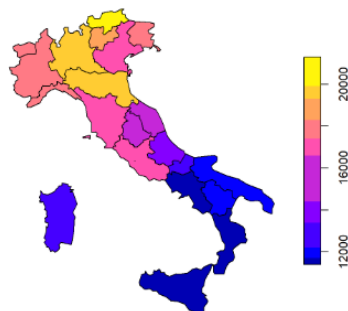


Airports by Country (2013)



© EuroGeographics for the administrative boundaries

Choropleth map





# Pacotes especializados - ggplot2

## 1. Pacotes para personalização e estilo

- **ggthemes**: Oferece temas adicionais para gráficos, como o estilo "Economist" e "Wall Street Journal".
- **cowplot**: Simplifica a combinação de múltiplos gráficos e aprimora layouts.
- **ggpubr**: Facilita a criação de gráficos de publicação, oferecendo opções pré-formatadas e estatísticas embutidas.
- **patchwork**: Combina múltiplos gráficos ggplot2 em um único layout com sintaxe simples.
- **hrbrthemes**: Disponibiliza temas focados em gráficos modernos e legíveis.

ggpubr 0.6.0 Reference Changelog

### ggpubr: 'ggplot2' Based Publication Ready Plots

**ggplot2**, by Hadley Wickham, is an excellent and flexible package for elegant data visualization in R. However the default generated plots requires some formatting before we can send them for publication. Furthermore, to customize a ggplot, the syntax is opaque and this raises the level of difficulty for researchers with no advanced R programming skills.

The 'ggpubr' package provides some easy-to-use functions for creating and customizing 'ggplot2'-based publication ready plots.

Find out more at <https://rpkgs.datanovia.com/ggpubr/>.

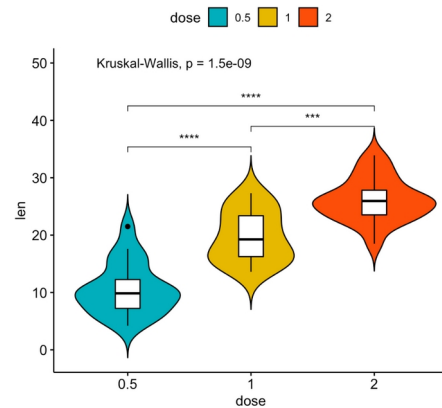
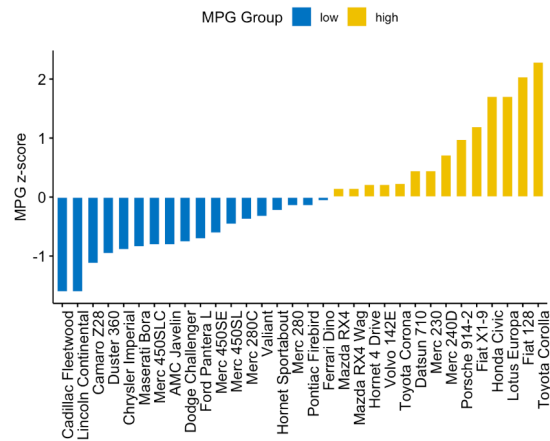
### Installation and loading

- Install from [CRAN](#) as follow:

```
install.packages("ggpubr")
```

- Or, install the latest version from [GitHub](#) as follow:

```
# Install
if(!require(devtools)) install.packages("devtools")
devtools::install_github("kassambara/ggpubr")
```



### Links

[View on CRAN](#)

[Browse source code](#)

[Report a bug](#)

### License

GPL (>= 2)

### Citation

[Citing ggpubr](#)

### Developers

Alboukadel Kassambara

Author, maintainer

### Dev status

[R-CMD-check](#)

CRAN 0.6.0

downloads 220K/month

downloads 15.4M



# Pacotes especializados - ggplot2

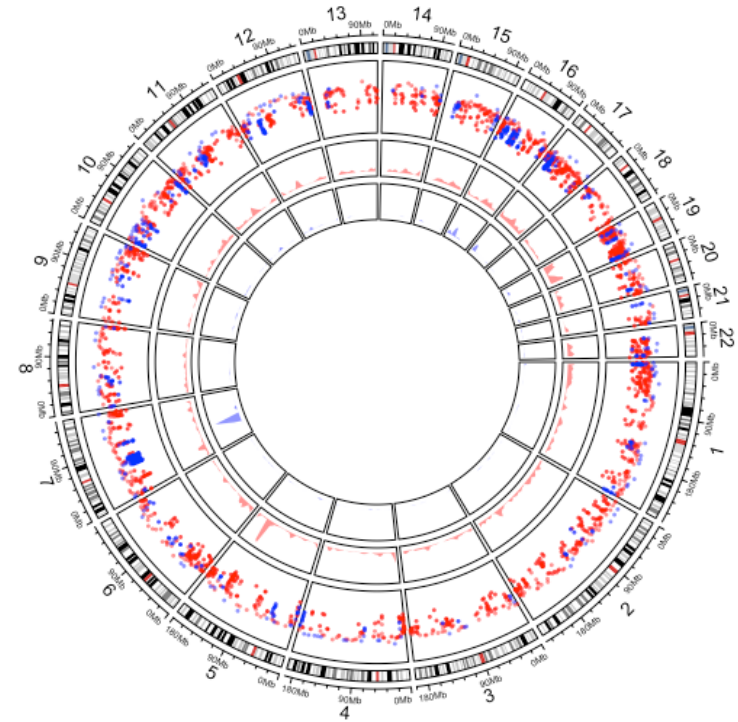
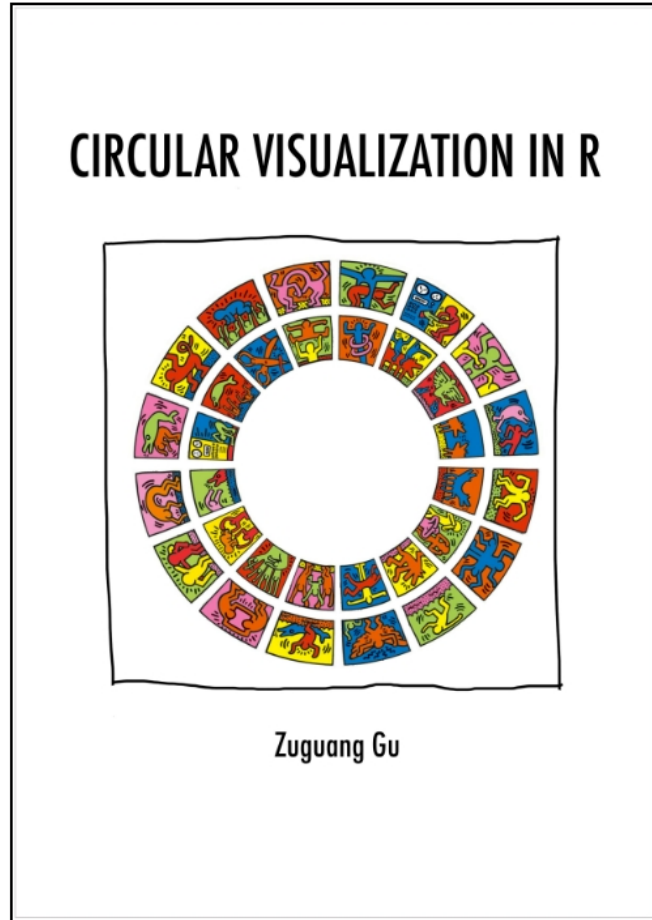



Figure 12.5: Genomic rainfall plot and densities.

# Pacotes especializados - ggplot2



**ggtree**  
GuangchuangYu/ggtree

[Download](#)
★ STARS 845

[ggtree](#)  
[Documentation](#)  
[FAQ](#)  
[Featured Articles](#)  
[Gallery](#)  
[Thanks](#)

The author

@guangchuangyu on Twitter

## ggtree: visualization and annotation of phylogenetic trees

release version [1.14.6](#)
devel version [1.15.5](#)
download [42242/total](#)
download [1656/month](#)

The [ggtree](#) package extending the [ggplot2](#) package. It based on grammar of graphics and takes all the good parts of [ggplot2](#). [ggtree](#) is designed for not only viewing phylogenetic tree but also displaying annotation data on the tree. [ggtree](#) is released within the [Bioconductor](#) project and the source code is hosted on [GitHub](#).


### Authors

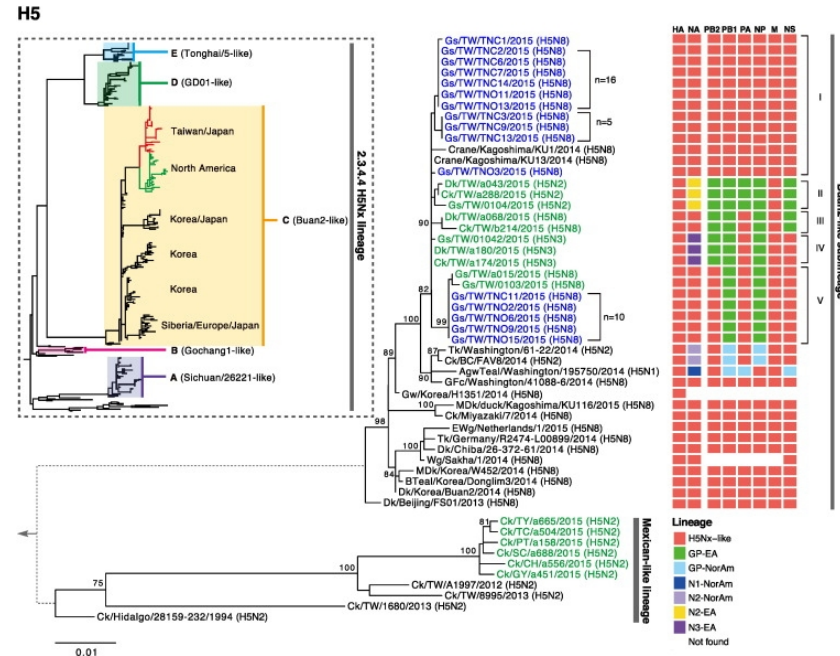
Guangchuang Yu, School of Basic Medical Sciences, Southern Medical University.

say [thanks](#)
follow me on [微信](#)
[打赏](#) 支付宝/微信

### Citation

Please cite the following article when using [ggtree](#) :






# Pacotes especializados - ggplot2

[README](#) [Code of conduct](#) [License](#)

## Make Complex Heatmaps

 R-CMD-check failing coverage 67% rank 49 / 2268 in Bioc 9.5 years



Complex heatmaps are efficient to visualize associations between different sources of data sets and reveal potential patterns. Here the **ComplexHeatmap** package provides a highly flexible way to arrange multiple heatmaps and supports various annotation graphics.

The [InteractiveComplexHeatmap](#) package can directly export static complex heatmaps into an interactive Shiny app. Have a try!

### Citation

Zuguang Gu, et al., [Complex heatmaps reveal patterns and correlations in multidimensional genomic data](#), Bioinformatics, 2016.

Zuguang Gu. [Complex Heatmap Visualization](#), iMeta, 2022.

### Install

ComplexHeatmap is available on [Bioconductor](#), you can install it by:

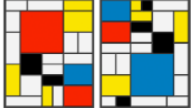
```
if (!requireNamespace("BiocManager", quietly=TRUE))
  install.packages("BiocManager")
BiocManager::install("ComplexHeatmap")
```

If you want the latest version, install it directly from GitHub:

```
library(devtools)
install_github("jokergoo/ComplexHeatmap")
```

ComplexHeatmap

Complete Reference



Zuguang Gu

[README](#)

## GOSemSim: GO semantic similarity measurement

in Bioc 15.5 years repo status Active platforms all codecov unknown

release version 2.30.0 devel version 2.31.1 download 817806/total download 17918/month

The semantic comparisons of Gene Ontology (GO) annotations provide quantitative ways to compute similarities between genes and gene groups, and have become important basis for many bioinformatics analysis approaches. GOSemSim is an R package for semantic similarity computation among GO terms, sets of GO terms, gene products and gene clusters. GOSemSim implemented five methods proposed by Resnik, Schlicker, Jiang, Lin and Wang respectively.

### Authors

Guangchuang YU <https://yulab-smu.top>

School of Basic Medical Sciences, Southern Medical University

Learn more at <https://yulab-smu.top/contribution-knowledge-mining/>.

If you use GOSemSim in published research, please cite:

- Yu G. Gene Ontology Semantic Similarity Analysis Using GOSemSim. In: Kidder B. (eds) Stem Cell Transcriptional Networks. *Methods in Molecular Biology*, 2020, 2117:207-215. Humana, New York, NY.
- Yu G\*, Li F\*, Qin Y, Bo X\*, Wu Y and Wang S\*. GOSemSim: an R package for measuring semantic similarity among GO terms and gene products. *Bioinformatics*. 2010, 26(7):976-978.

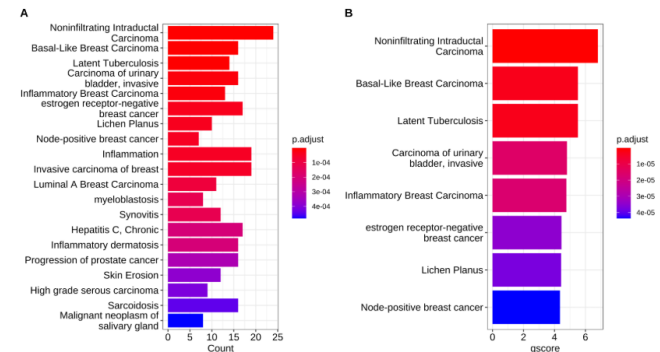


Figure 15.1: Bar plot of enriched terms.